НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Залєська- Зарічна Л.М

English for Computing

Навчальний посібник з англійської мови
для студентів НаУКМА факультету інформаційних технологій

2026

Навчальний посібник «ENGLISH FOR COMPUTING» призначений для студентів факультету інформаційних технологій, які прослуховують курс «Англійська мова за професійним спрямуванням», але може бути корисним для усіх бажаючих, які прагнуть вдосконалити свої знання англійської мови у галузі інформаційних технологій.

Посібник складається з дев'яти розділів (Application software . Computer languages, Types of High-level languages, Kinds of application software, Operating systems , Comparing operating systems, Programing, Databases, Network ,Cybersecurity, VR,AI), які охоплюють основні теми професійного спілкування в галузі інформаційних технологій та відповідають тематичному плануванню курсу «Англійська мова за професійним спрямуванням».

Посібник має на меті допомогти студентам розвивати та покращувати їхні комунікативні навички (а саме, читання, говоріння та письмове мовлення), а також поглибити їхні знання лексики (як спеціалізованої, так і загального спрямування). Відповідно кожен розділ включає автентичні тексти професійного спрямування, які доповнюються завданнями з говоріння, читання, письмового висловлення, а також лексичні вправи. Посібник розрахований як для аудиторної роботи зі студентами, так і для самостійної роботи студентів.

Отже, посібник надає можливість студентам удосконалити свої навички спілкування, що обумовлює його актуальність

CONTENTS

I. APPLICATION SOFTWARE. COMPUTER LANGUAGES 4
II. TYPES OF HIGH-LEVEL LANGUAGES. SYSTEM SOFTWARE 13
III. KINDS OF APPLICATION SOFTWARE 19
IV. OPERATING SYSTEMS. COMPARING OPERATING SYSTEMS 27
V. PROGRAMMING. CREATING COMPUTER PROGRAMS 38
VI. DATABASES AND DATABASE MANAGEMENT SYSTEMS 50
VII NETWORK STRUCTURES 56
VIII. CYBERSECURITY 64
IX. VIRTUAL REALITY. ARTIFICIAL INTELLIGENCE 72
X. COMPUTER TERMS GLOSSARY 82

I. APPLICATION SOFTWARE. COMPUTER LANGUAGES

1.Read and translate the text.

A digital computer cannot work without software, i.e. a program which controls different hardware components of the computer in a sequence specified by the user. By simply changing the program, the computer can be made to work in a different manner. In other words, software provides a digital computer with general-purpose computing capability. One should keep in mind that software and hardware are complementary to each other. Both have to work together to produce a meaningful result. The software, that is becoming an ever-larger part of the computer system, is growing more and more complicated, requiring teams of programmers and years of effort to develop. As a consequence, a new subdiscipline, software engineering, has arisen. The development of a large piece of software is perceived as an engineering task, to be approached with the same care as the construction of a skyscraper, for example, and with the same attention to cost, reliability, and maintainability of the final product.

Computer software is usually classified into two broad categories: application software and system software. Application software is a set of programs to carry out operations for a specific application. Application software may consist of one-off programs written by the user or alternatively of packages written by commercial software houses and bought by the user to accomplish specific tasks. For example, payroll is an application software for an organization to produce pay slips as an output. Application software is useful for word processing, billing system, accounting, producing statistical reports, analysis of numerous data in research, weather forecasting, etc. Such systems as MS Word, Lotus 1-2-3, d BASE III Plus are all application software.

Another example of application software is programming languages. Programming languages are the languages in which a programmer writes the instructions that the computer will ultimately execute. There are two major types of programming languages. These are low-level languages and high-level languages. Low-level languages are further divided into Machine language and Assembly language. The term low-level means closeness to the way in which the machine has been built. Low-level languages are machine oriented and require extensive knowledge of computer hardware and its configuration.

Machine language is the only language that is directly understood by the computer. It does not need any translator program. Machine language instructions, which are also called machine code, are written in binary form — i.e., strings of 1s and 0s (e.g. 01011010) — because electronic circuits inside computers, including memories, work only with the signals 0 and 1. When this sequence of codes is fed to the computer, it recognizes the codes and converts it into electrical signals needed to run it. For example, a program instruction may look like this: 10110010110100. It is not an easy language for the user to learn. It is also difficult to debug programs written in this language. Thus, programmers use languages that are readable for them but not for computers. For example, the mnemonic opcode LOAD in a human-readable language is far easier to understand than 0101 in machine code. Such a human readable language must be translated into machine language by special purpose programs called language processors. Machine language is considered to be the first generation language.

Assembly language was the first step to improve the programming structure. You know that computer can handle numbers and letters. Therefore some combinations of letters can be used to substitute for numbers of machine codes. The set of symbols and letters forms the assembly language and a translator program is required to translate this language to machine language.

This program is called Assembler. The symbolic programming of this language is easier to understand and saves a lot of time and effort of the programmer. It is also easier to correct errors and modify program instructions. Assembly language has the same efficiency of execution as the machine level language because there is a one-to-one translator between an assembly language program and its corresponding machine language program. Assembly language is considered to be a second-generation language. So, you now understand that programming in either assembly language or machine language is not an easy business requiring deep knowledge of computer hardware.

To make the programmer's work easier more convenient, high-level languages were soon (beginning in the mid-1950s) invented. High-level languages are problem oriented because the instructions are suitable for solving a particular problem. For example the language FORTRAN (FORmula TRANslation) was invented for solving mathematical problems. It was originally much like assembly language; however it allowed programmers to write algebraic expressions instead of coded instructions for arithmetic operations. COBOL (COmmon Business Oriented Language) was developed to handle records and files and the operations necessary for simple business application. The trend since then has been toward developing increasingly abstract languages, allowing the programmer to think and communicate with the machine at a level ever more remote from machine code.

2. Say whether the following sentences are true or false. Correct the false ones.

1. Software provides a digital computer with general-purpose capabilities.

2. Computer software and hardware are interchangeable.

3. As a separate discipline, software engineering appeared in the early days of computer science.
4. Application software may consist of one-off programs written by the user or alternatively of packages written by commercial software houses and bought by the user to accomplish specific tasks.

5. Application software is useful for word processing, billing system, accounting whereas system software is designed for solving research problems.

6. A programming language is a special coding system designed for writing instructions for a computer.

7. Low-level languages are machine-oriented, which means that they are used for solving a narrow circle of problems.

8. Low-level languages require no translator program.

9. High-level languages are used if it is not possible to solve a problem using a low-level language program.

10. High-level languages are termed so because they require high-skilled programmers being very difficult for man to understand and demanding deep knowledge of computer hardware.

3. Answer the questions.

1. How can a computer be made to work in a different manner?

2. What is computer software?

3. What does software provide a computer with?

4. How are computer software and hardware related? Explain this relationship.

5. How is the development of a large piece of software perceived?

6. What categories is computer software usually classified into?

7. What is application software? What may it consist of? Give some examples of application software.

8. What is a programming language? What two major types of programming languages exist?
9. What are the peculiarities of low-level languages? What does the term low-level mean?

10. What are the characteristic features of machine language? Why does machine language need no translator program?

11. What are the characteristic features of assembly language? In what way does it differ from machine language? What is the assembler?

12. What is a mnemonic opcode?

13. What were the reasons for the invention of high-level languages? What is the radical difference between high-level and low-level languages?

14. What were the languages FORTRAN and COBOL developed for?

15. What trend has been in the development of high-level languages since their introduction?

4. Complete the sentences translating their Ukrainian parts into English.

1. Software controls the different hardware components of the computer (у визначеній користувачем послідовності).

2. (Іншими словами, програмне забезпечення надає комп'ютеру) general-purpose computing capability.

3. Software and hardware have to work together (щоб дати значущий результат).

4. (Програмне забезпечення, що стає дедалі більшою частиною комп'ютерної системи, стає складнішим і складнішим, потребуючи) teams of programmers and years to develop.

5. (Як наслідок,) a new subdiscipline, software engineering, has arisen.

6. The development of a large piece of software is perceived as an engineering task, (до якої слід підходити з такою самою увагою, як, наприклад, до зведення хмарочоса,) and with the same attention to (вартості, надійності та зручності експлуатації кінцевого продукту).

7. (Прикладне програмне забезпечення може складатися зі спеціальних програм, що їх пише користувач, або з пакетів, які пишуть фірми) and bought by the user to accomplish specific tasks.

8. (Ще одним прикладом прикладного програмного забезпечення) is programming languages.

9. Such a human readable language (треба транслювати в машинну мову за допомогою спеціальної програми, яка називається «транслятор»).

10. The symbolic programming of assembly language (легше для розуміння та економить програмісту багато часу та зусиль).

11. (Аби полегшити програмісту роботу) more convenient, high-level languages were soon invented.

12. (Відтоді існує тенденція, спрямована на розвиток мов дедалі більш абстрактних,) allowing the programmer to think and communicate with the machine (на рівні дедалі більш несхожому на машинний код).

5. Give derivatives of the following words and explain their meanings.

Control, sequence, change, complement, mean, complicate, develop, engine, perceive, care, rely, maintain, apply, analyze, instruct, execute, ultimate, direct, consider, express, increase.

6. Give the opposites of the following words taken from text and using them make up sentences of your own.

Meaningful; arise; care; construction; final; analysis; numerous; improve; save (time, effort); one-to-one; deep; instead of; increasingly; remote from.

6. Find in the text the equivalents for:

emerge (appear), preassigned (predefined), in another way, order, to put it another way, enter, categorize (divide into, break down into), realize (consider, understand), profound, eventually, part (element), perform, supply (give), main (principal, basic), of this kind, collection.

7. Match up a word with its definition.

1) interface 2) firmware 3) virus 4) backup 5) driver 6) word processor 7) utility.

a) a copy of information held on a computer that is stored separately from the computer;

b) a computer program or part of a computer program that can make copies of itself and is intended to prevent the computer from working normally;

c) a set of instructions that form part of an electronic device and allow it to communicate with a computer or with other electronic devices;

d) a connection between two pieces of electronic equipment, or between a person and a computer;

e) a program used for preparing documents and letters, or a computer for doing this;

f) a computer program that makes it possible for a computer to use other pieces of equipment such as a printer;

g) a small program that provides an addition to the capabilities provided by the operating system.

8. Fill in the gaps in the sentences.

a)engaged	***b) layer***	***c) embedded***	***d) categories***	***e) mobiles;***
f) hardware	***g) tasks***	***h) programs***	***i) includes***	***j) running;***

Software refers to a set of …1… which is capable of performing some specific …2… on a computer system. They can be broadly classified into two …3… – System software and Application software. System Software is usually …4… in background processes. This software sync the work of …5… and other types of programs. It acts as a middle …6… between hardware and user applications. System software …7… programs like operating system. It is a well-known example of system software. This software interacts with the hardware and provides the

capability for …8… various types of programs. Desktop uses operating systems like Windows, Linux and MacOS, whereas Android and Windows 135 are commonly used operating systems for …9… . There are different types of operating systems like real time, distribute, …10…, etc

9. Study the vocabulary to the text .

control — керувати specify — задавати, визначати

meaningful — значущий

ever-larger — постійно зростаючий

arise (arose, arisen) — з'явитися, постати

perceive — 1) розуміти, усвідомлювати; 2) сприймати

care — ретельність, уважність, обережність

maintainability — експлуатаційна надійність; зручність експлуатації

application software — прикладне програмне забезпечення

system software — системне програмне забезпечення

one-off — одноразовий, (тут спеціальний)

alternatively — або software house — фірма з розробки та реалізації програмних засобів payroll — платіжна відомість (тут назва відповідної прикладної програми)

pay slip — бланк платіжної відомості

word processing — оброблення (редагування) тексту

ultimately — зрештою, в кінцевому підсумку

machine language — машинна мова

assembly language — мова асемблера

closeness — близькість, наближеність

translator program — програма-транслятор

run — виконувати (програму)

debug — налагоджувати (програму)

opcode — код операції

language processor — транслятор або інтерпретатор (програма, яка транслює або інтерпретує програму, написану мовою програмування)

handle — оперувати, працювати з чимось

therefore — тому, внаслідок чого

one-to-one — однозначний

trend — тенденція

remote — 1) віддалений; 2) несхожий

ADDITIONAL READING

The Diverse Functions of Application Software

Application software is designed with the primary goal of serving the specific needs of end users, from individuals to businesses. With its diversity and flexibility, application software offers numerous outstanding features, helping optimize work efficiency, automate processes, and enhance user experience. Below are the main functions of application software:

1. Information and Data Management

Managing emails, messages, audio, and video: Send, receive, store, and organize messages, audio, or video files conveniently.

Managing documents and data information: Efficiently store, search, and access information, supporting scientific and user-friendly data organization.

2. Business Administration

Customer Relationship Management (CRM): Integrates tools to track and manage customer information, improve interactions, and boost sales effectiveness.

Enterprise Resource Planning (ERP): Assists in managing resources from finance and HR to production and inventory, ensuring smooth business operations.

Project Management: Tracks progress, resources, and task allocation to achieve project goals effectively.

Process Management: Establishes, optimizes, and monitors workflows within an organization, ensuring high performance and consistency.

3. Finance and Human Resource Support

Accounting and Financial Management: Handles accounting tasks, manages cash flow, financial reporting, and cost optimization.

Human Resource Management: Supports managing employee information, payroll, attendance, and work schedules.

4. Content Creation and Presentation

Image and Video Development: Provides tools for designing, editing, and professionally presenting images and videos.

Presentation and Demonstration: Helps create visually appealing and engaging presentations.

5. Education and Training

Educational Software and E-learning: Supports teaching and learning through online platforms, enabling content management and facilitating anytime, anywhere learning.

6. Healthcare Support

Healthcare Applications: Manage medical information, monitor health indicators, and provide personalized health care advice.

7. Development and Programming

Website Development: Provides tools and platforms for designing and managing websites.

Programming Applications: Assists developers in writing code, testing, and deploying software.

Common Types of Application Software Today

Application software is categorized based on its purpose and field of application. Below are the most common types of application software, each tailored to meet specific user needs in personal life and work.

1. Personal Application Software

This type of software serves individual needs, from learning and entertainment to daily communication.

Word Processing and Document Editing Software: Microsoft Word, Google Docs.

Personal Management Applications: Google Calendar, Notion.

Image and Video Editing Software: Adobe Photoshop, Canva, iMovie.

Entertainment Applications: Spotify, Netflix, YouTube.

Learning Software: Duolingo, Quizlet, Coursera.

Personal application software helps users handle daily tasks quickly and efficiently while enhancing their quality of life.

2. Business Application Software

This type of software supports businesses in managing, operating, and optimizing workflows.

Customer Relationship Management (CRM): Salesforce, HubSpot.

Enterprise Resource Planning (ERP): SAP, Oracle NetSuite.

Human Resource Management Software: BambooHR, Workday, TEAMHUB HRM.

Accounting and Financial Software: QuickBooks, Xero.

Project Management Software: Trello, Asana, Jira.

These tools help businesses save time, increase productivity, and improve decision-making.

3. Mobile Application Software

Mobile application software refers to programs designed for smartphones and tablets.

Social Media Apps: Facebook, Instagram, TikTok.

Transportation and Navigation Apps: Google Maps, Grab, Uber.

E-commerce Apps: Shopee, Lazada, Amazon.

Personal Finance Apps: Momo, ViettelPay, PayPal.

Health Apps: MyFitnessPal, Headspace.

These applications focus on providing convenient and flexible solutions to meet users' needs anytime, anywhere.

4. Web-Based Application Software

Web applications run directly on web browsers without requiring installation on devices.

Online Work Tools: Google Workspace, Microsoft 365.

E-commerce Platforms: Shopify, Magento.

Online Design and Editing Software: Canva, Figma.

Online Learning Tools: Zoom, Microsoft Teams, Khan Academy.

Web-based software allows users to access tools from any internet-connected device while reducing storage burdens on personal devices.

5. Gaming Software

Gaming software is designed to entertain users through computer or mobile games.

Strategy Games: Age of Empires, Civilization.

Sports Games: FIFA, PES.

Action Games: Call of Duty, PUBG.

Mobile Games: Candy Crush, Clash of Clans.

Gaming software not only provides entertainment but also helps develop thinking skills and reflexes.

6. Specialized Software

Specialized software is designed for specific fields, meeting the demands of different industries.

Graphic Design Software: AutoCAD, Adobe Illustrator.

Medical Software: Hospital Information Systems (HIS), Medisoft.

Engineering Software: MATLAB, SolidWorks.

Educational Software: Moodle, Blackboard.

These tools offer optimized solutions for specific industries, helping professionals work more effectively.

7. Open-Source Applications

Open-source software allows users to freely use, modify, and distribute the source code.

Office Software: LibreOffice.

Database Management Systems: MySQL, PostgreSQL.

Content Management Systems (CMS): WordPress, Joomla.

Open-source software is popular for its flexibility, low cost, and high customizability.

Application software plays a crucial role in enhancing work efficiency, improving personal experiences, and driving business development in the digital age. With its diverse functions and types, application software not only automates processes, manages data efficiently, and facilitates communication but also supports collaboration and personalizes user experiences.

Understanding the various types of application software and selecting the right tools for your needs will provide long-term benefits, optimizing resources and effectively solving challenges. Whether you are an individual looking for solutions to support daily life or a business seeking tools to scale operations, application software is an indispensable choice. Stay tuned to Tokyo Tech Lab for more valuable insights into digital technology and modern innovation trends!

II. TYPES OF HIGH-LEVEL LANGUAGES. SYSTEM SOFTWARE

1.Read and translate the text.

Types of high-level languages. COBOL, FORTRAN, and their descendants such as Pascal and C are known as imperative languages, since they specify as a sequence of explicit commands how the machine is to go about solving the problem at hand; this is not very different from what takes place at the machine level. Other languages are functional, in the sense that programming is done by calling (i.e. invoking) functions or procedures, which are sections of code executed within a program. The best known language of this type is LISP (from List Processing), in which all computation is expressed as an application of a function to one or more "objects". Since LISP objects may be other functions as well as individual data items (variables, in mathematical terminology) or data structures, a programmer can create functions at the appropriate level of abstractions to solve the problem at hand. This feature has made LISP a popular language for artificial intelligence applications, although it has been somewhat superseded by logic programming languages as Prolog (from Programming in Logic). These are termed nonprocedural, or declarative, languages in the sense that the programmer specifies what goals are to be accomplished but not how specific methods are to be applied to attain those goals. Prolog is based on the concepts of resolution (akin to logical deduction) and unification (similar to pattern matching). Programs in such languages are written as a sequence of goals.

High-level languages have a major advantage over machine and assembly languages that they are easy to learn and use. It is because they are similar to the languages we use in our everyday life. Their disadvantage is that they require special software for translating high-level language programs into machine level programs. System software exists to supervise the operation of the machine and to facilitate easy access to the various hardware resources. System software are general programs designed for performing tasks such as controlling all operations required to move data into and out of the computer. It communicates with printers, card readers, disks, tapes, etc., monitors the use of various hardware like memory, CPU etc.

Also, system software is essential for the development of applications software. System software allows application packages to be run on the computer with less time and effort. Remember that it is not possible to run application software without system software. Development of system software is a complex task and it requires extensive knowledge of computer technology. Due to its complexity it is not developed in the house. Computer manufacturers build and supply this system software with the computer system. DOS, Unix, Windows are some of the widely used system software. Out of these Unix is a multi-user operating system whereas DOS and Windows are PC-based. System software can be broken down into several categories:

1. The operating system: programs which supervise the operation of the entire system including controlling the execution of all other software and diagnosing faults.

2. Language processors: programs which translate application software written in high-level languages (e.g. Pascal) into the low-level binary machine instructions which the processor executes. There are three main types: compilers, assemblers, and interpreters.

3. Library and utility programs: software to help the application programmer. It consists of standard commonly used software routines which the user may invoke (e.g. for copying files of data). An operating system is a set of programs to control and coordinate the operation of hardware and software in a computer system. It works analogously to a traffic police officer on a

congested street: it provides simple, efficient, and reliable means for utilizing the many different parts of the computer.

An operating system has a complex mixture of diversified functions. Its major functions are as follows: − management of processing; − memory management; − input/output and file management; − management of communications; − security; − user interface. An assembler translates a program written in an assembly language into machine code. Each instruction in the assembly language program has almost a one-to-one correspondence to a machine instruction in the machine code. In other words, the opcode and operand that constitute each assembly instruction are expressed with a readable name (i.e., a mnemonic opcode) and a decimal number, respectively, instead of the binary representation of the corresponding machine instruction. For example, LOAD 8 is an assembly instruction corresponding to the machine instruction 010100001000. Compared with other language processors, assemblers have relatively simple structures.

A compiler is a program translator that translates instructions of a high-level language to machine language. It is called compiler because it compiles machine language instructions for every program instruction of a high-level language. Thus, a compiler is a program translator like the assembler but more sophisticated. It scans the entire program first and then translates it into machine code.

An interpreter is another type of program translator used for translating high-level language programs into machine language programs. It takes one statement of a high-level language, translates it into machine language and immediately executes it. Translation and execution are carried out for each statement. It differs from a compiler, which translates the entire source program into machine code. Besides, a compiler is involved in the execution of a program. A program library is a collection of available computer programs and routines. Utility programs are specialized programs performing frequently required everyday tasks. Examples include sorting, report generation, file updating, file dump and backup, etc. Those programs are usually supplied by the manufacturer of the equipment.

2. Say whether the following sentences are true or false. Correct the false ones.

1. Pascal and C are predecessors of such languages as FORTRAN and COBOL.

2. LISP is a well-known example of declarative languages.

3. LISP and Prolog were designed for artificial intelligence applications while FORTRAN - for solving mathematical problems.

4. Programs in declarative languages are written as a sequence of explicit commands.

5. System software is supplied by computer manufacturers separately from computer systems.

6. A compiler is involved in the execution of a program.

7. Among other things, an operating system is responsible for finding spelling errors during word processing as well as for fighting computer viruses.

8. System software is essential for the development of computer components.

9. An interpreter translates the entire source program into machine code.

10. Utility programs are specialized programs performing frequently required operations.

3. Answer the questions.

1. What computer languages are called imperative? Give some examples.

2. What languages are referred to as functional?

3. How is computation in the language LISP expressed?

4. What may objects be in LISP?

5. What has made LISP a popular language for artificial intelligence applications?

6. What are the peculiarities of declarative languages?

7. What can be regarded as a disadvantage of high-level languages?

8. Why are high-level languages easier to learn and use as compared to machine and assembly languages?

9. What are the functions of system software? What sort of relationship is there between application software and system software?

10. Into what categories can system software be broken down? What are the functions of an operating system?

11. What is the purpose of language processors? Name their main types.

12. What are the peculiarities of an assembler?

13. What is the difference between a compiler and an interpreter?

14. What is the purpose of a program library and that of utility programs?

4. Complete the sentences translating their Ukrainian parts into English.

1. COBOL, FORTRAN, and their descendants such as Pascal and C are known as imperative languages, (позаяк вони вказують у вигляді явних команд, як машина має взятися за розв'язання наявної задачі).

2. (Оскільки об'єктами LISP можуть бути інші функції, а також окремі елементи даних або структури даних), a programmer can create functions at the appropriate level of abstractions.

3. These languages are termed nonprocedural or declarative (в тому розумінні, що програміст вказує, які цілі мають бути досягнені, а не те, як слід застосовувати конкретні методи для досягнення цих цілей).

4. System software are general programs (які призначені для керування всіма операціями, необхідними для введення даних у комп'ютер та виведення з нього).

5. High-level languages have a major advantage over machine and assembly languages (в тому, що їх легко вчити і застосовувати).

6. System software communicates with printers, card readers, disks, tapes, etc., (контролює використання різного апаратного забезпечення, такого як пам'ять, процесор тощо).

7. Development of system software is a complex task (і воно вимагає широких знань обчислювальної техніки).

8. The operating system consists of programs which (контролюють роботу всієї системи, включаючи керування виконанням усіх інших програм і діагностику несправностей).

9. (Операційна система забезпечує прості, ефективні та надійні засоби) for utilizing the many different parts of the computer.

10. Each instruction in the assembly language program (має майже однозначну відповідність машинній команді у машинному коді).

11. (Інтерпретатор — це ще один вид транслятора) used for translating high-level language programs into machine language programs.

12. Utility programs are specialized programs (що виконують повсякденні завдання, в яких часто виникає потреба).

5. Complete the sentences with a computer language from the text.

1._____ allows us to create our own tags to describe our data better. We aren't constrainer by a pre-defined set of tags the way we are with HTML.

2. IBM developed in the l 950s. It was the first high-level language in data processing.

3._____ applets are small programs that run automatically on web pages and let you watch animated characters, play games, etc.

4._____ is the HTML of the voice web. Instead of using a web browser and a keyboard, interact with a voice browser by listening to prerecorded audio output and sending, audio input through a telephone.

5. This language is widely used in the business community. For example, the statement ADDVAT NET-PRICE could be used in a _____program.

6. Complete the sentences with the words in italics.

Program programmers programming programmable

1. _____ is the process of venting a program using.

2.A computer _____is a set of instructions that tells the computer how to do a specific task.

3. Most computer _____make a plan of the program before they write it.

4. A _____keyboard allows the user to configure the layout and meaning of the keys

Compile compiler compilation

5. Programs written in a high-level language require _____ that is, translation into machine cone, the language understood by the processor.

6. A source program is converted into machine code by software called a _____.

7. Programmers usually_____ then programs to generate an object program and diagnose possible errors.

Bug debug debugger debugging

8. Any error or malfunction of a computer program is known as a _____.

9. A_____ is a program used to test and_____ other programs.

10. The process of going through the code to identify the cause of errors and fixing them is called_____.

7. Give derivatives of the following words and explain their meanings.

Sense, type, structure, solve, advantage, major, supervise, access, allow, time, supply, efficient, diversify, compare, entire, differ.

8. Give the opposites of the following words taken from the text and using them make up sentences of your own.

Descendant, explicit, within, individual, variable, artificial, facilitate, essential, extensive, standard, congested, diversified, supply.

9. Find in the text the equivalents for:

aim (purpose), take part, routine task, whole (complete), close to (like), program translator, characteristic (quality, peculiarity), direct, unique, right (suitable), drawback (shortcoming), interact, to some extent, to the effect that (in that), maker, because (as, for), construct (produce, create), invoke, make up, wide, merit (strength)

10. Study the vocabulary to the text .

descendant — нащадок

imperative language — імперативна мова

functional language — функціональна мова

declarative language — декларативна мова

explicit — явний

to invoke — викликати, активувати (програму, процедуру)

data item — елемент даних

somewhat — почасти, деякою мірою

to supersede — витісняти

resolution — рішення

akin to — подібно до

pattern matching — зіставлення зі зразком; ототожнення

card reader — пристрій для читання карт

fault — несправність

compiler — компілятор

interpreter — інтерпретатор

utility program — утиліта

congested — перевантажений

diversified — різноманітний

management — керування

to compile — 1) компілювати; 2) укладати, упорядковувати

source program — початкова програма

routine — 1) підпрограма; 2) стандартна програма

sorting — сортування

report generation — генерація звітів

file updating — внесення змін до файла

file dump — роздрукування файла

file backup — створення резервної копії файла

III. KINDS OF APPLICATION SOFTWARE

1.Read and translate the text.

Application software directs the computer to execute commands given by the user and includes any program that processes data for a user. Application software is generally divided into two groups: horizontal and vertical. A horizontal program can cut across many application areas. Vertical application programs are tailored to specific tasks. Application software includes word processors, spreadsheets, database management, inventory and payroll programs, design and manufacture applications, and many other kinds of "applications". Popular software is usually distributed on magnetic disks and CDs. Modern software and hardware can handle text, graphics, sound, and movies. MS WORD, Lotus 1-2-3 and dBASE III Plus are examples of application software.

Text-editing programs. Letters, reports, and documents can be easily prepared on personal computers, because the user can see on the monitor what the text will look like when printed (abbreviated as WYSIWYG, which stands for "what you see is what you get" and is pronounced "wizzy-wig"). Programs are available that check spelling and grammar before printing. Because of progress in laser printer technology, text printed by a laser printer is of typeset quality. A wide range of fonts is now available, and their number is far greater than that available with conventional typeset printing. All word processors offer facilities for document formatting, such as font changes, page layout, paragraph indention, and the like. Some word processors can also check spelling, find synonyms, incorporate graphics created with another program, correctly align mathematical formulas, create and print form letters.

Desktop publishing programs. Desktop publishing is the creation of pages for publication using personal computers. Utilizing programs for this purpose, one can easily draw pictures using a mouse as a pointer on the monitor of the computer. Text with arbitrary fonts can be easily mixed with the pictures, tables, and equations placed at desired locations.

Word processing is the most common application for a personal computer. Most word processing software programs allow us to create, 120 edit, and save documents, along with changing the position of the text in a document, inserting new information in the middle of the text, or removing words and sections no longer needed. With a typewriter, you would have to re-type the entire document after a few major changes. Given a computer, a document can be stored electronically and retrieved at any time for modification. Examples of word processing programs include Word Perfect, MS-Word, MultiMate, WordStar, DisplayWrite, Word for Windows, and Word Perfect for Windows.

Spreadsheet programs. Spreadsheet programs are the most popular programs for business calculations done on personal computers. They allow the user to enter columns and rows of numbers in a ledgerlike format on a monitor. The user can define a formula relating these columns and rows, and then the results are automatically calculated as numbers are entered. When changes in some entries are made, the program recalculates the results based on this formula. For example, at midyear a company can estimate year-end profits by projecting different sales growth rates for its products. Today's spreadsheets for PC's offer hundreds of functions. In addition, numerous spreadsheets representing different aspects of a business can be linked together. Changes in one cell of one spreadsheet are reflected in related cells in all linked spreadsheets.

Database management programs enable users to store large bodies of information and to search these databases in several ways.

Computer graphics programs enable computer users to create, change, and display pictures. The term computer graphics is also used to mean the pictures produced with these programs. The computer operator can create the original image on the computer or can use a previously created photograph or other picture that has been digitized. A digitized photo can be changed in a variety of ways. The user can change its dimensions or its colours, for example, or eliminate a part of it. Images in a photo can be moved or copied. Some computer graphics software works with motion pictures. Computer graphics plays a vital role in the publishing industry.

Presentation software is another major type of graphics program. This software enables users to create graphics to project onto motion picture screens at meetings. Most presentation software can also produce special effects, such as images that fade away or are transformed into other images, and even sound effects. Games software combines graphics, animation, sound, and music with clever design to produce exciting adventures and puzzles. Computer games are played on PC's or dedicated computers called video game units.

Virtual reality software uses graphics, sound, and other tools to create an artificial world through which a user can seem to move. Virtual worlds are filled with objects that can be "handled" by users wearing special sensor-lined gloves. Computer-aided design (CAD) programs are essential to many professions. Scientific visualization software is used in virtually every branch of science. One use of this software is development and testing of theories.

2. According to the text, are the following sentences TRUE or FALSE? If they are false, say why.

1) One of the primary functions of the first laptop computers was to store and calculate volumes of financial data for banks and large businesses.

2) End-user programs enable the user to perform tasks, such as creating documents, spreadsheets, databases and publications, doing online research, sending email, designing graphics, running businesses, and even playing games.

3) Application software is common to the task it is designed for.

4) When you begin creating a document, the word processing software will need to set the margins, font style and size, and the line spacing for you.

5) The word processor application allows you to add color, headings, and pictures or delete, copy, move, and change the document's appearance to suit your needs.

6) A software suite is a group of software applications with related functionality.

7) Sony Audio Master Suite is used for video production.

8) A Web browser or simply browser is an application specifically designed to locate, retrieve, and display content found on the Internet.

9) There are numerous application software for researchers and students, including gaming software such as World of Warcraft for the avid gamer, and iTunes for music lovers.

10) Word processing is the most specific application for a personal computer.

3. Find the words in the text which mean the following.

1) the programs used to direct the operation of a computer, as well as documentation giving instructions on how to use them;

2) a computer data file;

3) a type of software that offers the user a visual display of a simulated multicolumn worksheet and the means of using it especially for financial plans and budgets;

4) a comprehensive collection of related data organized for convenient access, generally in a computer;

5) a software program that allows the user to find and read encoded documents in a form suitable for display, especially such a program for use on the World Wide Web;

6) substantive information or creative material viewed in contrast to its actual or potential manner of presentation;

7) an ardent devotee; fan, enthusiast;

8) the act or process of calculating; computation;

9) the act of solving a problem, question, etc.

10) diligent and systematic inquiry or investigation into a subject in order to discover or revise facts, theories, applications, etc.

4. Translate into English paying special attention to the italicized words.

1. На виставці комп'ютерних технологій можна було побачити різні види сучасного прикладного програмного забезпечення, такого як програми керування базами даних, програми віртуальної реальності, ігрове програмне забезпечення, програми автоматизованого проектування і тому подібне.

2. Сучасні текстові редактори у поєднанні зі струменевими або лазерними принтерами забезпечують друк друкарської якості.

3. Електронні таблиці, де стовпці й рядки зв'язані певними формулами, дуже зручні для різних розрахунків: математичних, економічних, статистичних тощо.

4. Програма Photoshop дозволяє радикально змінювати попередньо створені фотозображення.

5. Розумна цінова і маркетингова політика дозволила фірмі збільшити темпи зростання продажів.

6. Світ віртуальної реальності створюється за допомогою спеціальної гарнітури (eyephones), тривимірних окулярів (3D goggles) і начинених датчиками рукавиць, що дозволяють маніпулювати об'єктами, які бачить людина.

7. Складання прогнозів погоди на тривалий період потребує обробки великих обсягів інформації, тому метеослужби (meteorological services) багатьох країн використовують для цього суперкомп'ютери.

8. Застарілі текстові редактори не можуть мати в собі графіку, створену іншою програмою.

9. У наш час засоби автоматизованого проектування відіграють важливу роль у таких галузях техніки, як архітектура, машино-, автомобіле- та авіабудування тощо.

10. Під час автоматичного виготовлення друкованих плат (printed circuit boards) виводи елементів схеми вставляються в спеціально призначені отвори, розташовані в потрібних місцях.

5. Translate the sentences paying attention to the meaning of the words in bold type.

1. Most **application** programs are now written in high-level languages.

2. The director received twenty **applications** for the position.

3. When you **apply** for a job, you must fill in an application form.

4. This liquid is for external **application** only.

5. **Computer-aided** design refers to any application of a computer to the solution of design problems.

6. You should remember that if you show **application** in your work, the boss may promote you soon.

7. This material **tailors** quite well.

8. This cloth is very difficult to **tailor**.

9. A **tailor-mad**e suit is usually much more expensive than a ready-made one.

10. The construction company **tailored** the house to the needs of the occupants.

11. All her novels are **tailored** to popular tastes.

12. A computer can **handle** numbers and letters.

13. It cost much effort to **handle** the situation.

14. This shop does not **handle** imported goods.

15. Oliver pulled the **handle** but the door proved to be locked on the inside.

16. Your indiscrete behavior may give your enemies a **handle** against you.

6. Possible topics for discussion

1.Bill Gates said: "A solid working knowledge of productivity software and other IT tools has become a basic foundation for success in virtually any career." Do you agree with him?
2.What is the greatest piece of software ever created?
3.Why do you think Microsoft has stuck to software and never went into producing computers, like Apple?
4.Do you think software is good value for money?
5.What do you think of people who design software?

Additional reading

Read and translate the following text .

Design and implementation

The software development lifecycle is a framework that project managers use to describe the stages and tasks associated with designing software. The first steps in the design lifecycle are planning the effort and then analyzing the needs of the individuals who will use the software and creating detailed requirements. After the initial requirements analysis, the design phase aims to specify how to fulfill those user requirements.

The next is step is implementation, where development work is completed, and then software testing happens. The maintenance phase involves any tasks required to keep the system running.

The software design includes a description of the structure of the software that will be implemented, data models, interfaces between system components and potentially the algorithms the software engineer will use.

The software design process transforms user requirements into a form that computer programmers can use to do the software coding and implementation. The software engineers develop the software design iteratively, adding detail and correcting the design as they develop it.

The different types of software design include the following:

Architectural design. This is the foundational design, which identifies the overall structure of the system, its main components and their relationships with one another using architectural design tools.

High-level design. This is the second layer of design that focuses on how the system, along with all its components, can be implemented in forms of modules supported by a software stack. A high-level design describes the relationships between data flow and the various modules and functions of the system.

Detailed design. This third layer of design focuses on all the implementation details necessary for the specified architecture.

How to maintain software quality

Software quality measures if the software meets both its functional and nonfunctional requirements.

Functional requirements identify what the software should do. They include technical details, data manipulation and processing, calculations or any other specific function that specifies what an application aims to accomplish.

Nonfunctional requirements -- also known as quality attributes -- determine how the system should work. Nonfunctional requirements include portability, disaster recovery, security, privacy and usability.

Software testing detects and solves technical issues in the software source code and assesses the overall usability, performance, security and compatibility of the product to ensure it meets its requirements.

The dimensions of software quality include the following characteristics:

Accessibility. The degree to which a diverse group of people, including individuals who require adaptive technologies such as voice recognition and screen magnifiers, can comfortably use the software.

Compatibility. The suitability of the software for use in a variety of environments, such as with different OSes, devices and browsers.

Efficiency. The ability of the software to perform well without wasting energy, resources, effort, time or money.

Functionality. Software's ability to carry out its specified functions.

Installability. The ability of the software to be installed in a specified environment.

Localization. The various languages, time zones and other such features a software can function in.

Maintainability. How easily the software can be modified to add and improve features, fix bugs, etc.

Performance. How fast the software performs under a specific load.

Portability. The ability of the software to be easily transferred from one location to another.

Reliability. The software's ability to perform a required function under specific conditions for a defined period of time without any errors.

Scalability. The measure of the software's ability to increase or decrease performance in response to changes in its processing demands.

Security. The software's ability to protect against unauthorized access, invasion of privacy, theft, data loss, malicious software, etc.

To maintain software quality once it is deployed, developers must constantly adapt it to meet new customer requirements and handle problems customers identify. This includes improving functionality, fixing bugs and adjusting software code to prevent issues. How long a product lasts on the market depends on developers' ability to keep up with these maintenance requirements.

When it comes to performing maintenance, there are four types of changes developers can make, including:

Corrective. Users often identify and report bugs that developers must fix, including coding errors and other problems that keep the software from meeting its requirements.

Adaptive. Developers must regularly make changes to their software to ensure it is compatible with changing hardware and software environments, such as when a new version of the OS comes out.

Perfective. These are changes that improve system functionality, such as improving the user interface or adjusting software code to enhance performance.

Preventive. These changes are done to keep software from failing and include tasks such as restructuring and optimizing code.

Software licensing and patents

A software license is a legally binding document that restricts the use and distribution of software.

Typically, software licenses provide users with the right to one or more copies of the software without violating copyright. The license outlines the responsibilities of the parties that enter into the agreement and may place restrictions on how the software can be used.

Software licensing terms and conditions generally include fair use of the software, the limitations of liability, warranties, disclaimers and protections if the software or its use infringes on the intellectual property rights of others.

Licenses typically are for proprietary software, which remains the property of the organization, group or individual that created it; or for free software, where users can run, study, change and distribute the software. Open source is a type of software where the software is developed collaboratively, and the source code is freely available. With open source software licenses, users can run, copy, share and change the software similar to free software.

Over the last two decades, software vendors have moved away from selling software licenses on a one-time basis to a software-as-a-service subscription model. Software vendors host the software in the cloud and make it available to customers, who pay a subscription fee and access the software over the internet.

Although copyright can prevent others from copying a developer's code, a copyright cannot stop them from developing the same software independently without copying. A patent, on the other hand, enables a developer to prevent another person from using the functional aspects of the software a developer claims in a patent, even if that other person developed the software independently.

In general, the more technical software is, the more likely it can be patented. For example, a software product could be granted a patent if it creates a new kind of database structure or enhances the overall performance and function of a computer.

[https://searchapparchitecture.techtarget.com/definition/software]

The history of software

The term *software* was not used until the late 1950s. During this time, although different types of programming software were being created, they were typically not commercially available. Consequently, users -- mostly scientists and large enterprises -- often had to write their own software.

June 21, 1948. Tom Kilburn, a computer scientist, writes the world's first piece of software for the Manchester Baby computer at the University of Manchester in England.

Early 1950s. General Motors creates the first OS, for the IBM 701 Electronic Data Processing Machine. It is called General Motors Operating System, or GM OS.

1958. Statistician John Tukey coins the word *software* in an article about computer programming.

Late 1960s. Floppy disks are introduced and are used in the 1980s and 1990s to distribute software.

Nov. 3, 1971. AT&T releases the first edition of the Unix OS.

1977. Apple releases the Apple II and consumer software takes off. 1979. VisiCorp releases VisiCalc for the Apple II, the first spreadsheet software for personal computers.

1981. Microsoft releases MS-DOS, the OS on which many of the early IBM computers ran. IBM begins selling software, and commercial software becomes available to the average consumer.

1980s. Hard drives become standard on PCs, and manufacturers start bundling software in computers.

1983The free software movement is launched with Richard Stallman's GNU (GNU is not Unix) Linux project to create a Unix-like OS with source code that can be freely copied, modified and distributed.
1984Mac OS is released to run Apple's Macintosh line.
Mid-1980s. Key software applications, including AutoDesk AutoCAD, Microsoft Word and Microsoft Excel, are released.

1985Microsoft Windows 1.0 is released.
1989. CD-ROMs become standard and hold much more data than floppy disks. Large software programs can be distributed quickly, easily and relatively inexpensively.

1991. The Linux kernel, the basis for the open source Linux OS, is released.

1997. DVDs are introduced and able to hold more data than CDs, making it possible to put bundles of programs, such as the Microsoft Office Suite, onto one disk.

1999Salesforce.com uses cloud computing to pioneer software delivery over the internet.
2000The term software as a service (SaaS) comes into vogue.
2007. IPhone is launched and mobile applications begin to take hold.

2010 to the present. DVDs are becoming obsolete as users buy and download software from the internet and the cloud. Vendors move to subscription-based models and SaaS has become common.

IV. OPERATING SYSTEMS. COMPARING OPERATING SYSTEMS

1.Read and translate the text.

Although in theory all operating systems perform very similar functions, in practice there are many different operating systems. One primary reason is the highly competitive nature of the computer industry. Operating systems, like other major computer products, evolve and undergo improvements and updates over time. In addition, applications are developed to take advantage of specific features in operating systems, so applications developed for one operating system may not be available for other operating systems.

MS-DOS. The ancestor of MS-DOS was developed in the early 1970s by Tim Patterson of Seattle Computer Products Inc. Later, Microsoft Corporation acquired the rights and began licensing it as MS-DOS (Microsoft Disk-Operating System). MS-DOS is a single-tasking operating system. Newer versions have a command-driven user interface. Their commands can be used to format disks; copy, rename, delete, and back up files; and organize and manage files on the disk. Newer versions also include a user interface called the shell with pull-down menus to access commands. MS-DOS has several advantages. An extensive number of applications have been written for MS-DOS, so you can generally find an application to meet your needs. MS-DOS does not require a powerful computer or a large amount of memory to run. It also has several disadvantages. MS-DOS was specifically developed for the Intel family of microprocessors. It has no network services or multimedia extensions . And it has no limitations on how it uses memory. Also, MS-DOS does not provide a user interface for application programs. Thus, applications from, say, Lotus Development Corporation and Borland International present the user with completely different interfaces.

A graphical user interface allows a user to learn one interface that works with all applications. MS-DOS users who want a graphical user interface can obtain Microsoft Windows, an extension to MS-DOS that provides standard methods for switching among applications, file and program management, windowing, icons, pull-down menus, scroll bars and dialog boxes — temporary windows that contain choices when the program needs additional information from the user. To take advantage of these features, most software vendors have released Windows versions of their MS-DOS application programs.

Windows NT. Windows NT (New Technology), introduced in 1992, is a descendant of DOS and the Windows extension to DOS developed at Microsoft. It is a multitasking operating system. It has several advantages. It can run DOS and Windows programs without modification. It has network services and multimedia extensions, and it is not limited to the Intel family of processors. It has several disadvantages. It is designed for high-performance computers and requires large amounts of memory and disk space. It does not have as many applications developed for it as do other operating systems.

Windows XP. Windows XP is a line of operating systems developed by Microsoft for use on personal computers, including home and business desktops, notebook computers, and media centers. The name “XP” stands for eXPerience. It was codenamed “Whistler”, after Whistler, British Columbia, as many Microsoft employees skied at the Whistler-Blackcomb ski resort during its development. Windows XP is the successor to both Windows 2000 Professional and Windows Me, and is the first consumer-oriented operating system produced by Microsoft to be built on the Windows NT kernel (version 5.1) and architecture. Windows XP was first released on October 25, 2001, and over 400 million copies were in use in January 2006. It is succeeded by Windows Vista, which was released to volume license customers on November 8, 2006 and

worldwide to the general public on January 30, 2007. The most common editions of the operating system are Windows XP Home Edition, which is targeted at home users, and Windows XP Professional, which has additional features such as support for Windows Server domains and two physical processors, and is targeted at power users and business clients. Windows XP Media Center Edition has additional multimedia features enhancing the ability to record and watch TV shows, view DVD movies, and listen to music.

Windows XP Tablet PC Edition is designed to run the ink-aware Tablet PC platform. Two separate 64-bit versions of Windows XP were also released, Windows XP 64-bit Edition for IA-64 (Itanium) processors and Windows XP Professional x64 Edition for x86-64. Windows XP is known for its improved stability and efficiency over the 9x versions of Microsoft Windows. It presents a significantly redesigned graphical user interface, a change Microsoft promoted as more user-friendly than previous versions of Windows.

New software management capabilities were introduced to avoid the "DLL hell" that plagued older consumer-oriented 9x versions of Windows. It is also the first version of Windows to use product activation to combat software piracy, a restriction that did not sit well with some users and privacy advocates. Windows XP has also been criticized by some users for security vulnerabilities, tight integration of applications such as Internet Explorer 6 and Windows Media Player, and for aspects of its default user interface.

Later versions with Service Pack 2, and Internet Explorer 7 addressed some of these concerns. Windows Vista. Windows Vista is a line of operating systems developed by Microsoft for use on personal computers, including home and business desktops, laptops, Tablet PCs, and media centers. Prior to its announcement on July 22, 2005, Windows Vista was known by its codename "Longhorn". Development was completed on November 8, 2006; over the following three months it was released in stages to computer hardware and software manufacturers, business customers, and retail channels. On January 30, 2007, it was released worldwide to the general public, and was made available for purchase and downloading from Microsoft's web site.

The release of Windows Vista comes more than five years after the introduction of its predecessor, Windows XP, the longest time span between successive releases of Microsoft Windows. Windows Vista contains many changes and new features, including an updated graphical user interface and visual style dubbed Windows Aero, improved searching features, new multimedia creation tools such as Windows DVD Maker, and redesigned networking, audio, print, and display sub-systems. Vista also aims to increase the level of communication between machines on a home network, using peer-to-peer technology to simplify sharing files and digital media between computers and devices. Windows Vista includes version 3.0 of the .

NET Framework, which aims to make it significantly easier for software developers to write applications than with the traditional Windows API. Microsoft's primary stated objective with Windows Vista, however, has been to improve the state of security in the Windows operating system. One common criticism of Windows XP and its predecessors has been their commonly exploited security vulnerabilities and overall susceptibility to malware, viruses and buffer overflows. In light of this, Microsoft chairman Bill Gates announced in early 2002 a company-wide "Trustworthy Computing initiative" which aims to incorporate security work into every aspect of software development at the company. Microsoft stated that it prioritized improving the security of Windows XP and Windows Server 2003 above finishing Windows Vista, thus delaying its completion. While these new features and security improvements have garnered positive reviews, Vista has also been the target of much criticism and negative press. Criticism of Windows Vista has targeted high system requirements, its more restrictive licensing terms, the

inclusion of a number of new digital rights management technologies aimed at restricting the copying of protected digital media, lack of compatibility with certain pre-Vista hardware and software, and the number of authorization prompts for User Account Control. As a result of these and other issues, Vista has seen adoption and satisfaction rates lower than Windows XP.

Operating System/2 (OS/2). When IBM decided to introduce a second generation of personal computers in 1987, it named them Personal Systems/2. The operating system, called Operating System/2 (OS/2) is an operating system for this generation of personal computers. It was developed under a joint agreement between IBM and Microsoft. OS/2 is a multitasking operating system. The latest version has a graphical user interface called the Workplace that provides a windowbased user interface for the operating system's file management functions and allows applications to share data among themselves. OS/2 has several advantages. It has network services and multimedia extensions. However, it also has disadvantages. It was specifically designed for the Intel family of microprocessors. It requires a high-performance computer with large amounts of memory and disk space. It does not have as many applications developed for it as do other operating systems.

Macintosh Operating System. Apple's Macintosh Operating System, simply called System, was introduced in 1984. It is a descendant of research done at Xerox PARC (Palo Alto Research Center) and Apple's Lisa computer, the user interface of which was designed by a team headed by Larry Tesler. The latest version, System 7, is a multitasking operating system. Its graphical user interface, called the Finder, has extensive graphics capabilities and is known for being very easy to use. System 7 has several advantages. It has network services and multimedia extensions. It also has disadvantages. It was developed specifically for the Motorola family of processors, although it can also run on the PowerPC microprocessors — a family of microprocessors jointly developed by Apple, IBM, and Motorola.

UNIX. UNIX was developed by Ken Thompson and Dennis Ritchie in 1969 at Bell Laboratories, the research and development arm of AT&T. Now available commercially from over 30 vendors, UNIX has been popular with programmers and scientific and engineering customers because of its flexibility. UNIX is a multitasking multiuser operating system licensed by Novell Corporation. There are two standard versions of UNIX: System V.4 is offered by UNIX International, an organization led by AT&T, Novell, and Sun Microsystems OSF/1 is offered by the Open Software Foundation, an organization led by IBM, Digital, and HewlettPackard. Both versions offer similar graphical user interfaces. 164 Unix's major advantage is that it is a very modular operating system that can be assembled like building blocks to perform various functions. It has network services and is processor independent. UNIX suffers from a lack of standardization: programs that run under one version often do not run under another version. Computer programmers with years of experience find UNIX commands hard to learn, and text editors (vi, emacs) are equally hard to use.

2.Answer the following questions .

1.How does competition within the computer industry influence the evolution and fragmentation of operating systems?

2.In what ways did MS-DOS's hardware dependence on Intel microprocessors limit its long-term scalability compared to later operating systems?

3.Analyze how the lack of a unified graphical user interface in MS-DOS affected software usability and user learning curves.

4.Why was Microsoft Windows initially positioned as an extension to MS-DOS rather than as a standalone operating system, and what limitations did this approach impose?

5.Compare Windows NT and MS-DOS in terms of multitasking, hardware independence, and system requirements. What trade-offs did Windows NT introduce?

6.Why was Windows XP considered a major turning point in Microsoft's operating system strategy for consumers?

7.How did Windows XP attempt to address software management and stability issues present in earlier Windows 9x versions?

8.Discuss the security criticisms directed at Windows XP and explain how Service Packs attempted to mitigate these concerns.

9.What were Microsoft's primary objectives with Windows Vista, and why did these goals lead to mixed reception among users?

10.Evaluate how Windows Vista's security enhancements conflicted with usability and hardware compatibility.

11.Compare OS/2 and the Macintosh Operating System in terms of processor dependence, multitasking capabilities, and application ecosystem.

12.Why is UNIX described as both highly flexible and difficult to use, and how does its lack of standardization impact software portability?

3.Say whether the following statements are true or false . Correct the false ones.

1.MS-DOS was originally developed by Microsoft in the early 1970s before being licensed to other companies.

2.MS-DOS is a single-tasking operating system that relies primarily on a command-driven interface.

3.The lack of a standard graphical user interface in **MS-DOS** caused application programs to present inconsistent user interfaces.

4.Windows NT can run DOS and Windows programs without requiring modification of those programs.

5.Windows NT was designed to run efficiently on low-performance computers with limited memory.

6.Windows XP was the first Microsoft operating system to unify consumer and business platforms using the Windows NT kernel.

7.All editions of **Windows XP** were designed exclusively for 64-bit processors.

8.Windows Vista introduced the Windows Aero interface and significantly increased Microsoft's focus on operating system security.

9.One major criticism of **Windows Vista** was its lower hardware requirements compared to Windows XP.

10.OS/2 was developed solely by IBM and was independent of Intel microprocessors.

11.The **Macintosh Operating System (System 7)** is known for a graphical user interface that emphasizes ease of use and strong graphics capabilities.

4.Choose the correct option

1. **UNIX** is processor-independent but suffers from portability issues due to differences among its versions.
   The word **"competitive"** in the phrase *"the highly competitive nature of the computer industry"* most nearly means:
   A) cooperative
   B) profitable
   C) rivalrous
   D) experimental

2. In the sentence *"operating systems evolve and undergo improvements"*, the verb **"undergo"** is closest in meaning to:
   A) prevent
   B) ignore
   C) experience
   D) initiate

3. The noun **"ancestor"** (used to describe MS-DOS) refers to something that:
   A) replaces an older system
   B) competes with newer systems
   C) precedes and gives rise to something else
   D) improves performance

4. In *"applications are developed to take advantage of specific features"*, the phrase **"take advantage of"** means:
   A) exploit unfairly
   B) copy illegally
   C) make effective use of
   D) depend entirely on

5. The adjective **"extensive"** in *"an extensive number of applications"* suggests that the number is:
   A) limited

B) outdated
C) large and wide-ranging
D) unverified

6. The word **"specifically"** in *"specifically developed for the Intel family"* functions as a(n):
A) adjective meaning exclusive
B) noun meaning purpose
C) adverb emphasizing precision
D) conjunction showing contrast

7. In *"Windows NT is designed for high-performance computers"*, the word **"designed"** most closely means:
A) decorated
B) tested
C) intended
D) marketed

8. The phrase **"did not sit well with some users"** most nearly means:
A) confused users
B) pleased users
C) caused dissatisfaction
D) was misunderstood

9. In the context of Vista, **"prioritized"** in *"Microsoft prioritized improving security"* means:
A) delayed
B) avoided
C) treated as most important
D) underestimated

10. The noun **"criticism"** in the text refers to:
A) neutral description
B) promotional feedback
C) expressed disapproval
D) technical documentation

11. In *"UNIX suffers from a lack of standardization"*, the verb **"suffers"** implies that UNIX:
A) is physically damaged
B) intentionally restricts itself
C) is negatively affected by something
D) benefits indirectly

12. The adjective **"modular"** (used to describe UNIX) most nearly means:
A) rigid and fixed
B) outdated
C) composed of interchangeable parts
D) visually complex

5.Read the text and find new information about Linux you did not know.

Brief History of Linux Linus Torvalds was a student at the University of Helsinki in 1991 when he decided to create a new and different "UNIX clone." Torvalds decided the new OS would be named "Linux," a combination of his name and "UNIX." Torvalds had two goals. First, he wanted to create a powerful, featurerich OS that provided the same functioning of UNIX. His OS would run on almost any computer, regardless of its architecture or the type of applications it hosted. Second, the OS would be completely open; anyone could contribute to its development and adapt or change its code, as long as they made their innovations public and did not take credit for anyone else's work. As part of its openness, Linux would be available for free to anyone who wanted it, although no one would be prohibited from selling their version of Linux as long as they made their innovations public and kept the OS completely open. To achieve these goals, Torvalds decided that Linux would be built from the ground up, without using any code from any commercial version of UNIX. To keep Linux open, Torvalds posted a message on the Internet in 1991 inviting programmers around the world to help him develop the new operating system. The call was taken up by dozens of programmers, who immediately wanted to share Torvalds' dream. Piecing the work out among themselves, different programmers tackled different aspects of the program, sharing code and ideas over the Internet. By 1994, enough pieces had been stitched together and the first version of Linux was released to anyone who wanted to download it. Although commercial versions of the OS now exist, Linux remains open and still attracts a community of developers interested in contributing to it. By some estimates, Linux runs on more than 10 million computers, a number that is growing rapidly. An ever-increasing number of corporate IT and database managers, Web site operators, and ISPs are using Linux as the core operating system on mission-critical computer networks. There is probably no greater mark of Linux's acceptance, however, than the acknowledgment it has received from Microsoft. Although few people believe that Linux will ever replace Windows as the desktop operating system of choice (or replace Windows NT as the primary network OS), Microsoft executives have said that they view Linux as a legitimate competitor. Leading hardware vendors – including Intel, IBM, Compaq, and others – have announced their support of Linux. In fact, some server-class computers are now being shipped with Linux, rather than UNIX or Windows NT, installed. In the software world, leading database companies such as Informix and Oracle have announced that their corporate database products will be tailored to run under Linux. For computer users in the home and in most average business settings, Linux is not an issue. With its cryptic command-driven interface, Linux is not likely to become the operating system of the masses, even though developers have created a Windows-like GUI for it

Windows XP

Windows XP is a line of operating systems developed by Microsoft for use on general-purpose computer systems, including home and business desktops, notebook computers, and media centers. The letters «XP» stand for experience. Windows XP is the successor to Windows 2000 and is the first consumer-oriented operating system produced by Microsoft to be built on the Windows NT kernel and architecture. Windows XP was first released in October 2001. The most common editions of the operating system are Windows XP Home Edition, which is targeted at home users, and Windows XP Professional, which has additional features, such as support for Windows Server domains and dual processors, and is targeted 92 PART I. COMPUTER BASICS at power users and business clients. Windows XP Media Center Edition consists of Windows XP Professional with new features enhancing the ability to record and watch TV shows, watch DVDs, listen to music and more. Windows XP Tablet PC Edition is designed to

run the ink-aware Tablet PC platform. Two separate 64-bit versions of Windows XP were also released, Windows XP 64-bit Edition for IA-64 (Itanium) processors and Windows XP Professional x64 Edition for AMD64/EM64T processors. Windows XP is known for its improved stability and efficiency over previous versions of Windows. It presents a significantly redesigned graphical user interface (GUI), a change Microsoft promoted as more user-friendly than previous versions of Windows. New software management capabilities were introduced to avoid the "DLL hell" that plagued older consumer versions of Windows. It is also the first version of Windows to use product activation to combat software piracy, a restriction that did not sit well with some users and privacy advocates. Windows XP has also been criticized by some users for security vulnerabilities, tight integration of applications such as Internet Explorer and Windows Media Player, and for aspects of its user interface. Windows Vista is scheduled to be the next major revision of Microsoft Windows, with a planned release date of November 2006 for business editions, and January 2007 for other editions.

6.Say whether the following statements are true or false .Correct the false ones.

1.Linux was created as a commercial alternative to UNIX with the goal of generating profit for its developer.

2.Linus Torvalds intended Linux to be an open system that anyone could modify and redistribute.

3.Linux was built using parts of existing commercial UNIX systems.

4.The first public version of Linux became available in 1994 after international collaboration among programmers.

5.Linux development has always been limited to academic and hobbyist environments.

6.Microsoft officially considers Linux a serious competitor in the operating system market.

7.Windows XP was the first Microsoft operating system built on the Windows NT architecture for home users.

8.Windows XP was primarily designed for servers and did not support home or multimedia use.

9.One of the major criticisms of Windows XP was its requirement for product activation.

10.Linux is expected to fully replace Windows as the dominant desktop operating system.

7. Fill in the gaps with suitable words from the text.

Windows XP is a line of operating systems developed by Microsoft for use on **(1)** _____-purpose computer systems, including both home and business desktops. The letters "XP" stand for **(2)** _____.

Windows XP is the **(3)** _____ to Windows 2000 and was the first consumer-oriented operating system built on the Windows NT **(4)** _____ and architecture. It was first **(5)** _____ in October 2001.

The most common **(6)** _____ of Windows XP are the Home Edition and the Professional Edition. The Professional Edition is **(7)** _____ at power users and business clients and includes additional features such as support for Windows Server **(8)** _____ and dual processors.

Windows XP Media Center Edition enhances the ability to record and **(9)** _____ TV shows, watch DVDs, and listen to music. The Tablet PC Edition is designed to run the **(10)** _____-aware Tablet PC platform.

Windows XP is known for improved **(11)** _____ and efficiency compared to earlier versions of Windows. It introduced new software management capabilities to prevent the so-called "DLL **(12)** _____".

The operating system was also the first version of Windows to use product **(13)** _____ to combat software piracy, a measure that did not sit well with some users.

Despite its popularity, Windows XP has been **(14)** _____ for security vulnerabilities and aspects of its user interface.

8. Discussion. Discuss the following questions with your groupmates.

1.How would using a computer be different if it had no operating system? How would programming be different?
2.How would using a computer be different if it had no operating system? How would programming be different?
3.List several mental tasks that people do better than computers. List several mental tasks that computers do better than people. Can you find any general characteristics that distinguish the items on the two lists?
4.Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc. What OS does your home computer have? Are you satisfied with it? Have you had any troubles?

4. Additional reading

Batch operating system

The users of a batch operating system do not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group. The programmers leave their programs with the operator and the operator then sorts the programs with similar requirements into batches.

Time-sharing operating systems

Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing.

The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of Multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time.

Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation. That is, if **n** users are present, then each user can get a time quantum. When the user submits the command, the response time is in few seconds at most.

The operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems have been modified to timesharing systems.

Distributed operating System

Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.

The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as **loosely coupled systems** or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

Network operating System

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

Examples of network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

Real Time operating System

A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. The time taken by the system to respond to an input and display of required updated information is termed as the **response time**. So in this method, the response time is very less as compared to online processing. Real-time systems are used when there are rigid time requirements on the operation of a processor or the flow of data and real-time systems can be used as a control device in a dedicated application. A real-time operating system must have well-defined, fixed time constraints, otherwise the system will fail. For example, Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

There are two types of real-time operating systems.

Hard real-time systems guarantee that critical tasks complete on time. In hard real-time systems, secondary storage is limited or missing and the data is stored in ROM. In these systems, virtual memory is almost never found.

Soft real-time systems are less restrictive. A critical real-time task gets priority over other tasks and retains the priority until it completes. Soft real-time systems have limited utility than hard real-time systems. For example, multimedia, virtual reality, Advanced Scientific Projects like undersea exploration and planetary rovers, etc.

V. PROGRAMMING. CREATING COMPUTER PROGRAMS

1. Read and translate the text.

A computer program is a set of commands that tell the CPU what to do. Software may contain only an executable program file, or it may have several other supporting files such as dynamic link libraries, initialization files, and help files. An executable file (EXE) is the part of a program that sends commands to the processor that executes the commands in the file. In fact, when you run a program, you are running the executable file. A dynamic link library (DLL) is a partial EXE file, it contains a part of an executable program and does not run on its own; its commands are accessed by another running program. These files allow programmers to break large programs into small components; it makes the entire program easier to upgrade.

DLL files can also be shared by several programs at one time. This feature makes them efficient for program storage. An initialization file (INI) contains configuration information, such as the size and starting point of a window, the color of the background, the user's name, and so on. Initialization files help programs start running or contain information that programs can use as they run. Although initializing files are still in use, many newer programs now store user preferences and other program variables in the Windows Registry. By including a help file (HLP), programmers can provide the user with PC-based help.

To create a program, using a programming language a programmer creates source code, which is compiled or interpreted to create object code that the computer can understand. Object code, also known as machine code, is the binary language file that tells the CPU what to do. When you launch a program, the computer begins reading and carrying out its statements. The order in which program statements are executed is called program flow control. When mapping a program, a programmer creates a flowchart. The steps represented in a flowchart are called an algorithm and usually lead to some desired result. To perform certain tasks, the actual programming process uses variables and functions. Variables are placeholders for data being processed (e.g. variable Age). Functions, or mini-algorithms, are discrete sets of codes used to perform one task like finding the square root of a number or the average of a set of numbers.

2. Answer the following questions.

1. What is a computer program?

2. What types of files can a program contain?

3. What is an executable file?

4. What is a dynamic link library?

5. What feature makes DLL files efficient for program storage?

6. What information do initialization files contain?

7. What information is stored in the Windows Registry?

8. What is the purpose of help files?

9. What tasks does a programmer perform to create a program?

10. What is a machine code?

11. What is called program flow control?

12. What does a programmer create when mapping a program?

13. What is an algorithm?

14. What is called a variable function?

3. Tell whether the following statements are true or false. Correct the false ones.

1. A computer program is a set of instructions or statements that is carried out by the computer's RAM.

2. Software may contain not only an executable program file, but also have several other supporting files.

3. An executable file provides the user with PC-based help.

4. A DLL is a complete, fully functioning executable program file.

5. DLL files can also be shared by several programs at one time.

6. A help file contains information about the size and starting point of a window, the color of the background, the user's name.

7. Programmers use tools that convert their human-language instructions into codes that computers can understand.

8. Using a flowchart, you can depict any step-by-step process, regardless of the desired result.

9. The steps in a flowchart represent an algorithm.

10. When writing a program, the programmer's first step is to create a source code.

11. Although it is possible for programmers to write programs in machine code, it is more practical to use special tools called programming languages, to write programs.

12. Programming process can use variables and functions.

4. Fill in the blanks using the information from the text .

1. When you run a program, you are running its _____ file.

2. A (an) _____ file is a partial EXE file.

3. _____ files can help the programs start running or contain information that programs can use as they run.

4. By including a help file (HLP), programmers can provide the user with _____.

5. When mapping a program, a programmer creates a (an) _____.

6. _____ also known as machine code, is the binary language file that tells the CPU what to do.

7. _____ is the order in which program statements are executed.

8. The steps represented in a flowchart are called _____.

9. A (an) _____ is a placeholder for data being processed.

10. Discrete sets of code used to perform one task are _____, or mini-algorithms.

5. Choose the right answer.

1. Which of the following types of files actually sends commands the processor?
a. EXE. b. DLL.
2. Which of the following types of files contains configuration information?
a. EXE. b. DLL. c. INI. d. HLP.

3. To be understood by the computer's hardware, program instructions must be formatted as ...
a. programming language. b. machine language.
4. A programming language enables the programmer to create a description of a program, and save the description in a file. The resulting description is called ...
a. source code. b. machine code.
5. Before it can be run on a computer, source code must be converted into ...
a. compiler code. b. assembler code.
6. The order in which program statements are executed is called ...
a. flowchart. b. structure.
7. An algorithm is a set of steps that always ...
a. leads to a solution. b. looks the same in a flowchart.
8. To perform certain tasks, the actual programming process uses...
a. variables. b. HLP files.

6. Match each item to the correct statement below.
a. EXE file b. DLL file c. INI file d. HLP file
1. Can be shared by several programs.
2. Stores configuration information.
3. Sends commands to the processor.
4. Newer programs use Windows Registry instead of this.
5. A partial EXE file.
6. When you run a program, you are running this.
7. Provides PC-based help.
8. Allows programmers to break large programs into small components.
a. Source code b. Object code c. Algorithm d. Program control flow
9. A description of a program.
10. The contents of an executable file.
11. Order of execution.
12. A set of steps that always leads to a solution.
13. The same whether done by a PC or by hand.
14. Can be depicted by a flowchart.

7.Structured and Object-Oriented Programming . Read and translate the text.

Structured Programming Structured programming evolved in the 1960s and 1970s. The name refers to the practice of building programs using a set of well-defined structures: Sequence structure defines the default control flow in a program. This structure is built into programming languages. A computer executes lines of code in the order in which they are written. It is possible, as a result of a conditional statement or a function call, that the flow may have the option of going in one of several different directions. Selection structures are built around the conditional statements. If the conditional statement is true, certain lines of code are executed. If the conditional statement is false, those lines of code are not executed. Repetition structures (or

looping structures) use the loops. In a repetition structure, the program checks a conditional statement and executes a loop based on the condition. If the condition is true, then a block of one or more commands is repeated until the condition is false.

Object-Oriented Programming. In the 1980s object-oriented programming (OOP) was developed. Many programmers claim that an object orientation is a natural way of thinking about the world and it makes programs simpler and programming faster. Concepts of object-oriented programming are objects and classes. OOP enhances structured programming. Objects are composed of structured program pieces, and the logic of manipulating objects is also structured. Every object has attributes and functions and may contain other objects. For example, the car object has attributes (color, size, shape, top speed), functions (moves forward, moves backward, opens its windows) and may encapsulate other objects (tires, chassis, motor) with their own attributes and functions. All objects belong to classes. A class consists of attributes and functions shared by more than one object. All cars, for example, have a steering wheel and four tires. All cars can drive forward, reverse, park, and accelerate. Class attributes are called data members, and class functions are represented as member functions or methods. Classes can be divided into subclasses. The car class, for example, could have a luxury sedan class, a sports car class, and an economy car class. Subclasses typically have all the attributes and methods of the parent class. Every sports car, for example, has a steering wheel and can drive forward. This phenomenon is called class inheritance. In addition to inherited characteristics, subclasses have unique characteristics of their own (fuel economy, trunk space, appearance). When an object is created, it automatically has all the attributes and methods associated with that class. In the language of OOP, objects are instantiated (created). Objects do not typically perform behaviors spontaneously. A car, for example, cannot move forward and backward at the same time or drive forward spontaneously. You send a signal to the car to move forward by pressing on the accelerator. Likewise, in OOP, messages are sent to objects, requesting them to perform a specific function. Part of designing a program is to identify the flow of sending and receiving messages among the objects.

8. Answer the following questions.

1. What is structured programming?
2. What are the three structures used in structured programming?
3. How does sequence structure work?
4. What is the difference between selection structure and repetition structure?
5. When was object-oriented programming developed?
6. What are the basic concepts of object-oriented programming?
7. What is an object made up of?
8. What is an object characterized by?
9. Give an example of an object and its attributes and functions.
10. What does a class consist of?
11. What are class attributes and functions called?
12. Give an example of a subclass.
13. What is meant by class inheritance?
14. How do objects perform functions in object-oriented programming?

9. Tell whether the following statements are true or false. Correct the false ones .

1. There are three basic structures used in structured programming.
2. Sequence structure defines the default control flow in a program.
3. Sequence structure and selection structure are similar.
4. With selection structure, if the condition is true, then a block of one or more commands is

repeated until the condition is false.
5. With the repetition structure, the program checks a conditional statement and executes a loop based on the condition.
6. Object-oriented programming was developed in the 1970s.
7. With the OOP programming becomes faster.
8. The basic concepts of object-oriented programming are objects, classes and structures.
9. Every object in OOP may encapsulate other objects.
10. All objects belong to subclasses.
11. Class inheritance means that subclasses of objects typically have all the attributes and methods of the parent class.
12. In OOP, messages are sent to objects for them to perform a specific function.

10. Fill in the blanks using the information from the text .

1. _____, _____ and _____ are used in structured programming.
2. Selection structures are built around _____.
3. In a repetition structure, the program _____ a conditional statement and _____ a loop based on the condition.
4. The acronym "OOP" stands for _____.
5. With the object-oriented programming programs become _____ and programming becomes _____.
6. A(an) _____ is a component of an object's overall description.
7. Class attributes are called _____, and class functions are represented as _____.
8. Subclasses have _____ and _____ characteristics.
9. In the language of OOP, objects are _____.
10. Objects send _____ to one another, to make requests.

11. Choose the right answer.

1. With this structure, if the condition is true, then a block of one or more commands is repeated until the condition is false.
a. Selection structure. b. Sequence structure. .c Repetition structure. d. All the above.
2. In structured programming selection structures rely on the use of ...
a. conditional statements. b. functions. c. loops. d. objects.
3. The basic concepts of object-oriented programming are ...
a. classes. b. none of the above. c. objects. d. all the above.
4. In object-oriented programming attributes and functions define a (an) ...
a. object. b. class. c. statement. d. subclass.
5. In object-oriented programming the term "encapsulate" means...
a. contain. b. evolve. c. enhance. d. none of the above.
6. Objects can share attributes and functions. Taken together, these shared attributes and functions constitute a ...
a. class. b. member. c. message. d. condition.
7. All objects belong to ...
a. members. b. messages. c. classes. d. statements.
8. In the language of object-oriented programming the term "instantiate" means ...
a. send. b. create. c. inherit. d. include.

12. Match each item to the correct statement below.

a. Structured programming

b. Object-oriented programming
c. Sequence structure
d. Selection structure
1. Uses attributes and functions.
2. Controls program flow in three ways.
3. Built around the conditional statements.
4. Built into programming languages.
5. Earlier development.
6. Makes programming faster.
a. Classes b. Subclasses c. Data members d. Data functions
7. Have all the attributes and methods of the parent class.
8. Objects belong to them.
9. Class attributes are referred to them.
10. Class functions are called so.
11. Have both inherited and unique characteristics.
12. Consist of attributes and functions shared by more than one object.

13.Programming Languages . Read and translate the text.

Programming is a way of sending instructions to the computer. To create these instructions, programmers use programming languages to create source code, and the source code is then converted into machine (or object) code, the only language that a computer understands. People, however, have difficulty understanding machine code. As a result, first assembly languages and then higher-level languages were developed. Programming languages require that information be provided in a certain order and structure, that symbols be used, and sometimes even that punctuation be used. These rules are called the syntax of the programming language, and they vary a great deal from one language to another.

Categories of Languages Based on evolutionary history, programming languages fall into one of the following three broad categories: Machine Languages. Machine languages consist of the 0s and 1s of the binary number system and are defined by hardware design. A computer understands only its machine language – the commands in its instruction set that instruct the computer to perform elementary operations such as loading, storing, adding, and subtracting. Assembly Languages. These languages were developed by using English-like mnemonics. Programmers worked in text editors to create their source files. To convert the source files into object code, researchers created translator programs called assemblers. Assembly languages are still much easier to use than machine language.
Higher-Level Languages. These languages use syntax that is close to human language, they use familiar words instead of communicating in digits. To express computer operations, they use operators, such as the plus or minus sign, that are the familiar components of mathematics. As a result, reading, writing, and understanding computer programs is easier. Machine languages are considered first-generation languages, and assembly languages are considered second-generation languages. The higher-level languages began with the third generation.

Third-generation languages (3GLs) can support structured programming, use true English like phrasing, make it easier for programmers to share in the development of programs. Besides, they are portable, that is, you can put the source code and a compiler or interpreter on practically any computer and create working object code. Some of the third-generation languages include the following: FORTRAN, COBOL, BASIC, Pascal, C, C++, Java, ActiveX.

Fourth-generation languages (4GLs) use either a text environment, much like a 3GL, or a visual environment. In the text environment, the programmer uses English-like words when generating source code. In a 4GL visual environment, the programmer uses a toolbar to drag and drop various items like buttons, labels, and text boxes to create a visual definition of an application. Many 4GLs are database-aware; that is, you can build programs with a 4GL that work as front end (an interface that hides much of the program from the user) to databases. Programmers can also use 4GLs to develop prototypes of an application quickly. Some of the fourth-generation languages are Visual Basic and Visual Age. A 5GL would use artificial intelligence to create software based on your description of what the software should do.

14. Answer the following questions.

1.What is programming?
2. How are instructions created?
3. What is called the syntax of the programming language?
4. What are the categories of programming languages?
5. What is a machine language?
6. What is the difference between machine language and assembly language?
7. What is a translator program?
8. Why is programming in higher-level languages easier than in machine and assembly languages?
9. What languages are considered first- and second-generation languages?
10. What are third-generation languages characterized by?
11. Give the names of third-generation languages.
12. What types of environment are used with fourth-generation languages?
13. Give the names of some fourth-generation languages.
14. What would fifth-generation languages be characterized by?

15. Tell whether the following statements are true or false. Correct the false ones.
1. Assembly-languages and higher-level languages were developed because people have difficulty understanding machine code.
2. Programming languages require the programmer to follow rules of syntax.
3. Compilers and interpreters can correct the programmer's syntax errors when creating object code.
4. Programming languages are usually grouped by their place in the evolution of programming languages.
5. Machine languages are the most advanced of all programming languages.
6. Machine languages and assembly languages are both considered first-generation programming languages.
7. Assembly languages are based on binary number system.
8. Third-generation languages use a visual environment.
9. Third-generation languages are highly portable.
10. Because they are more complex and sophisticated, fourth-generation languages are considerably more difficult to use than third generation languages.
11. One benefit of fourth-generation languages is their ability to generate prototype applications quickly.
12. Visual Basic lets programmers create applications only in a text environment.

16. Fill in the blanks.
1. Many programming languages follow a set of rules called _____.

2. Programming languages fall into _____, _____ and _____.
3. Machine languages are defined by _____ design.
4. A computer understands only its machine language and performs elementary operations such as _____, _____, _____ and _____.
5. _____ languages were developed by using Englishlike mnemonics for commonly used strings of machine language.
6. _____ are used for converting the source files into object code.
7. The term _____ means that you can put the source code and a compiler or interpreter on practically any computer and create working object code.
8. Fourth-generation languages use either a _____ environment or a _____ environment.
9. Examples of fourth-generation languages are _____ and _____.
10. A(n) _____ is an interface to a program that hides much of the program from the user.

15. Choose the right answer.

1. To create source code, programmer use ...
a. programming languages. b. object code. c. compilers. d. interpreters.
2. A programming language's syntax rules may require that ...
a. information is provided in a certain order. b. symbols are used. c. punctuation is used. d. all the above.
3.Which of the following is not a major category of programming languages?
a. Machine languages. b. Assembly languages. c. Lower-level languages. d. Higher-level languages.
4. Machine languages are defined by ...
a. hardware design. b. portability. c. compilers. d. syntax.
5.Which of the following is closest to human language?
a. Machine languages. b. Assembly languages. c. Higher-level languages. d. All are similar to human language.
6. Which of the following is not a category of higher-level languages?
a. Third-generation languages. b. Fourth-generation languages. c. Fifth-generation languages. d. Sixth-generation languages.
7. Which of the following is considered to be a second-generation programming language?
a. Machine language. b. Assembly language. c. Higher-level language. d. None of the above.
8. Which type of language was the first to use true English-style phrasing?
a. First-generation language. b. Second-generation language. c. Third-generation language. d. Fourth-generation language.

16. Match each item to the correct statement below.

a. Machine languages
b. Assembly languages
c. Higher-level languages
d. Third-generation languages
e. Fourth-generation languages
f. Fifth-generation languages

1. The most basic of languages.
2. Consist of strings of numbers.
3. Use familiar words.
4. Utilize artificial intelligence.

5. Were the first to use English-like mnemonics.
6. Use a text environment and a visual environment.
7. Developed to make programming easier.
8. Support structured programming.
9. Visual Basic is an example of them.
10. Translator programs are used with them

16. Discuss with your partner some amazing great programming quotes about software development (by Jeremy Morgan #programming)

1."Programming isn't about what you know; it's about what you can figure out." - *Chris Pine*
Especially important for beginners. At first, we're so anxious about knowing everything, especially language syntax. Problem-solving is the skill we end up using most.

2."The only way to learn a new programming language is by writing programs in it." - *Dennis Ritchie*
Programmers are mostly "learn by doing" types. No amount of academic study or watching other people code can compare to breaking open an editor and start making mistakes.

3."Sometimes it's better to leave something alone, to pause, and that's very true of programming." - *Joyce Wheeler*
When managing developers I would always encourage getting up and walking away from the computer when you have a problem. Some of your best solutions will come to you when you're not at the machine.

4."In some ways, programming is like painting. You start with a blank canvas and certain basic raw materials. You use a combination of science, art, and craft to determine what to do with them." - *Andrew Hunt*
Outsiders question whether programming is art. Programmers don't.

5."Testing leads to failure, and failure leads to understanding." - *Burt Rutan*
Some developers hate testing. However, shifting your attitude and embracing it makes you a better developer.

6."The best error message is the one that never shows up." - *Thomas Fuchs*
Remember this the next time you decide to focus on some great error reporting.

7."The most damaging phrase in the language is.. it's always been done this way" - *Grace Hopper*
To be a programmer long term, you have to love change. You can't just tolerate it, you have to love it.

Additional reading

Read the text and answer the questions below to see how well you understand the topic.

Quantum computing has been seeing increased attention over the last decade, since these computers, which function according to the principles of quantum physics, have enormous potential. Today, most researchers believe that these computers will one day be able to solve certain problems faster than classical computers, since to perform their calculations they use entangled quantum states in which various bits of information overlap at a certain point in time. This means that in the future, quantum computers will be able to

efficiently solve problems which classical computers cannot solve within a reasonable timeframe.

This quantum supremacy has still to be proven conclusively. However, some significant technical advances have been achieved recently. In late summer 2019, a quantum computer succeeded in solving a problem -- albeit a very specific one -- more quickly than the fastest classical computer.

For certain "quantum algorithms," i.e. computational strategies, it is also known that they are faster than classical algorithms, which do not exploit the potential of quantum computers. To date, however, these algorithms still cannot be calculated on existing quantum hardware because quantum computers are currently still too error-prone.

Utilizing the potential of quantum computation not only requires the latest technology, but also a quantum programming language to describe quantum algorithms. In principle, an algorithm is a "recipe" for solving a problem; a programming language describes the algorithm so that a computer can perform the necessary calculations.

Today, quantum programming languages are tied closely to specific hardware; in other words, they describe precisely the behavior of the underlying circuits. For programmers, these "hardware description languages" are cumbersome and error-prone, since the individual programming instructions must be extremely detailed and thus explicitly describe the minutiae needed to implement quantum algorithms.

Computer scientists refer to computer languages that abstract from the technical details of the specific type of computer as high-level programming languages. Silq is the very first high-level programming language for quantum computers. High-level programming languages are more expressive, meaning that they can describe even complex tasks and algorithms with less code. This makes them more comprehensible and easier to use for programmers. They can also be used with different computer architectures.

[Abridged from
https://www.sciencedaily.com/releases/2020/06/200615115820.htm]

1) Utilizing the potential of quantum computation requires _____.
a) the latest technology
b) a quantum programming language and the latest technology
c) a quantum programming language

2) High-level programming languages can describe _____.
a) complex tasks and algorithms with less code.
b) complex tasks and algorithms with much code
c) complex algorithms

3)Today, _____are tied closely to specific hardware.
a)software programs
b)quantum programming languages
c)high-level languages

4.For programmers, these "hardware description languages" are _____.

a)easily managed and error-prone

b)not easily managed and error

c)cumbersome and error-prone

5)Computer scientists refer to computer languages that abstract from the technical details of the specific type of computer as _____.

a) high-level programming languages

b)low-level programming languages

c)high-level and low-level programming languages

Read and translate the following text

How the brain is programmed for computer programming?

Countries around the world are seeing a surge in the number of computer science students. Enrolment in related university programs in the U.S. and Canada tripled between 2006-2016 and Europe too has seen rising numbers. At the same time, the age to start coding is becoming younger and younger because governments in many different countries are pushing K-12 computer science education. Despite the increasing popularity of computer programming, little is known about how our brains adapt to this relatively new activity. A new study by researchers in Japan has examined the brain activity of thirty programmers of diverse levels of expertise, finding that seven regions of the frontal, parietal and temporal cortices in expert programmer's brain are fine-tuned for programming. The finding suggests that higher programming skills are built upon fine-tuned brain activities on a network of multiple distributed brain regions.

"Many studies have reported differences between expert and novice programmers in behavioural performance, knowledge structure and selective attention. What we don't know is where in the brain these differences emerge," says Takatomi Kubo, an associate professor at Nara Institute of Science and Technology, Japan, and one of the lead authors of the study.

To answer this question, the researchers observed groups of novices, experienced, and expert programmers. The programmers were shown 72 different code snippets while under the observation of functional MRI (fMRI) and asked to place each snippet into one of four functional categories. As expected, programmers with higher skills were better at correctly categorizing the snippets. A subsequent searchlight analysis revealed that the amount of information in seven brain regions strengthened with the skill level of the programmer: the bilateral inferior frontal gyrus pars triangularis (IFG Tri), left inferior parietal lobule (IPL), left supramarginal gyrus (SMG), left middle and inferior temporal gyri (MTG/IT), and right middle frontal gyrus (MFG).

"Identifying these characteristics in expert programmers' brains offers a good starting point for understanding the cognitive mechanisms behind programming expertise. Our findings illuminate the potential set of cognitive functions constituting programming expertise," Kubo says.

More specifically, the left IFG Tri and MTG are known to be associated with natural language processing and, in particular, semantic knowledge retrieval in a goal-oriented way. The left IPL and SMG are associated with episodic memory retrieval. The right MFG and IFG Tri are functionally related to stimulus-driven attention control.

"Programming is a relatively new activity in human history and the mechanism is largely unknown. Connecting the activity to other wellknown human cognitive functions will improve our understanding of programming expertise. If we get more comprehensive theory about programming expertise, it will lead to better methods for learning and teaching computer programming," Kubo says.

Check yourself. Say whether the following statements are true or false. Correct the false ones.

1.Object-oriented programming languages, such as C++ and Java, provide a formal set of rules for creating and managing objects. The data are stored in a traditional relational database.

2.There is one major feature in object-oriented programming that makes them different than non-OOP languages. It is polymorphism.

3.Classes are created in hierarchies, and inheritance allows the structure and methods in one class to be passed down the hierarchy.

4.Object-oriented programming allows procedures about objects to be created whose exact type is not known until runtime.

5.The C++ language, developed in 2000s, extended C by adding objects to it while preserving the efficiency of C programs.

6.In the early 1990s Java was designed by Sun Microsystems, Inc., as a programming language for the World Wide Web (WWW).

7.Visual Basic can also be used within other Microsoft software to program small routines.

8.Java and JavaScript are not used for programming small and portable devices, such as mobile telephones.

9.In the 2010s, Python became one of the most popular programming languages, along with Java and JavaScript.

VI. DATABASES AND DATABASE MANAGEMENT SYSTEMS

1.Read and translate the text.

A database is a collection of related data or facts arranged in a specific structure. A database management system (DBMS) is a program, or collection of programs, that allows multiple users to store, access, and process data or facts into useful information. Three of the most important terms to know about databases are a table, a record, a field. Data is stored in tables. A table is divided into records (unnamed rows), and each record is divided into fields (named columns). The table consists of a set number of fields and an arbitrary number of records. For a record to exist, it must have data in at least one field. To help you understand how a database stores data, think about a typical address book. Each piece of information in the address book is stored in its own location, called a field. For example, each entry has a field for First Name and another field for Last Name, as well as fields for Address, City, State, ZIP Code, and Phone Number. Each unique type of information is stored in its own field. One full set of fields – that is, all the related information about one person or object – is called a record. Therefore, all the information for the first person is record 1, all the information for the second person is record 2, and so on. A complete collection of records makes a table. Once you have a structure for storing data (whether it is a printed address book, phone book, or electronic table), you can enter and view data, create reports, and perform other tasks with the data. For example, you may create a customer report that lists customers by ZIP Code. A DBMS provides tools to perform data management functions: creating tables, sorting tables, entering and editing data, querying the database, viewing data, generating reports. Many different DBMS programs are available. Enterprise-level products, such as Oracle, DB2, and Sybase, are designed to manage large special-purpose database systems. Programs such as Microsoft Access, Corel's Paradox, and Lotus Approach are popular among individual and small-business database users

2. Answer the following questions.

1. What is a database?

2. What is a database management system?

3. What are the most important terms to know about databases?

4. Where is data stored in a database?

5. What is a table? a record? a field?

6. What type of information is stored in the field?

7. What data can a field contain?

8. What functions does a DBMS provide?

9. What enterprise-level DBMS programs do you know?

10. What DBMS programs are popular among individual and small business users?

3. Tell whether the following statements are true or false. Correct the false ones.

1. A database management system is a repository for collection.

2. A database is at the heart of many types of computer applications.

3. In a database, each piece of information is stored in its own location, called a directory.

4. In a database, a record contains all the related information about a single person or object.

5. In a database table, all the records are organized according to the same set of fields.

6. One full set of fields is called a table.

7. A database management system allows to view data, create reports and perform other tasks with the data.

8. DBMS programs are used only by individual users.

9. In a database, the number of records is set, but the number of fields is arbitrary.

10. A record must have data in at least one field.

4. Fill in the blanks.

1. The three most important terms to know about databases are _____, _____, and _____. 2. The table _____ a set number of fields and an arbitrary number of records. 3. In a database, each unique type of information is stored in its own _____. 4. All the _____ information about one person or object is called a record. 5. For a database record to exist, it must have data in at least one _____. 6. The DBMS _____ the user with data and the tools to work with the data. 7. A DBMS provides tools to _____ tables, _____, _____ data, _____ the database, _____ reports. 8. A database management system _____ multiple users to store, access, and process data or facts into useful information.

5. Choose the right answer.

1. A tool that allows users to store, access, and process data is called..

a. database. b. database management system.

2. A database is a collection of related facts arranged in a specific...

a. database management system. d. structure.

3. In a database, names such as “Last name” and “Address” may be given to the ...

a. records. b. files.

4. A database table is a collection of ...

a. records. b. fields.

5. In a database table, fields are stored as ...

a. columns b. rows c. form. d. query.

6. Each object’s record contains a limited number of..

a. data. b. tables.

6. Match each item to the correct statement below.

a. Database

b. Record

c. Database management system

d. Field

e. Table

1. A data repository.
2. A software tool.
3. Helps users process data into information.
4. At the heart of many types of applications.
5. A complete collection of records.
6. All the related information about one thing.
7. Each piece of information is stored in one.
8. The database has arbitrary number of them.
9. Can store any number of records.
10. The database has a set number of them.

7. Database Structures I. Read and translate the text.

There are several types of database structures, such as flat-file, relational, hierarchical, network and object-oriented ones. A database that consists of a single data table is called a flat-file (sequential file) database. Flat-file databases are useful for certain single-user or small group situations, especially for maintaining address lists or inventories. Flat-file database systems are easy to learn and use, but difficult to maintain and limited in their power. When numerous files exist (one for each table or related document), there is often a lot of data redundancy, which increases the chance for errors, wastes time, and uses excess storage space.

A relational database is made up of a set of tables, and a common field existing in any two tables creates a relationship between the tables. For example, a Customer ID Number field in both the Customers table and the Orders table links the two tables, while a Product ID field links the Orders and Products tables. The relational database structure is widely used in today's business organizations. In a business, a typical relational database contains such data tables, as Customer information, Employee information, Vendor information, Order information, Inventory information.

The hierarchical database is an older style of database. The tables are organized into a fixed treelike structure, with each table storing one type of data. The trunk table (the main table) stores general information. Any field in that table may reference another table that contains subdivisions of data. Each one of those tables may, in turn, reference other tables that store finer subdivisions of data. The relationship between tables is said to be a parent-child relationship, or one-to many relationship, with any child table relating to only one parent table. Each parent table may have many child tables, but each child has only one parent. Hierarchical databases require little duplicated data and may locate data quickly. However, the tables' fixed relationships limit the flexibility of the database, making some kinds of queries or reports difficult or impossible.

The network database model is similar to the hierarchical structure except that anyone table can relate to any number of other tables. The network database's tables, therefore, are said to have a many-to-many relationship. Like the hierarchical structure, the network database is used in older (primarily mainframe) systems. The object-oriented database (OODB) developed in the late 1980s, groups data items into complex items called objects. These objects can represent anything: a product, an event, a customer complaint, or even a purchase. An object is defined by its characteristics (e.g. text, sound, graphics, video), attributes (e.g. color, size, style, quantity, price), and procedures (the processing associated with an object

8. Answer the following questions.

1. What types of database structures are there?
2. What is a flat-file database?
3. Where are flat-file databases used?
4. What are the drawbacks of flat-file databases?
5. What is the structure of a relational database?
6. What tables can a relational database contain?
7. How are the tables in a hierarchical database organized?
8. What information does the trunk table store?
9. What is the relationship between tables in a hierarchical database?
10. What are the advantages and drawbacks of a hierarchical database?
11. What does the network database structure differ from the hierarchical database model in?
12. What kind of systems are the network and hierarchical databases used in?
13. When was the object-oriented database developed?
14. What does it group data items into?
15. What can the objects represent?

9. Tell whether the following statements are true or false. Correct the false ones.

1.If a database file contains only one data table, it is called an informational database.
2. Flat-file database systems are difficult to learn and use.
3. A relational database is made up of two tables.
4. In a relational database, a common field existing in any two tables creates a relationship between the tables.
5. A parent-child relationship is the same as a one-to-many relationship.
6. Network databases and hierarchical databases function in the same way .
7. In an object-oriented database, items of data are grouped into complex objects.
8. In an object-oriented database, each object can have only one characteristic.

10. Fill in the blanks.

1. Older database systems that used only a single table are called _____.
2. In an object-oriented database, an object is defined by its _____, _____, and _____.
3. There are several types of database structures, such as _____, _____, _____, _____and _____ones.
4. Flat-file database systems are _____ in their power.
5. A_____ is made up of a set of tables, and a common field existing in any two tables creates a _____ between the tables.
6. In a hierarchical database the tables are organized into a fixed _____ structure.
7. The network database's tables have a _____ relationship.
8. The relationship between tables in a hierarchical database is called a _____ relationship.

11. Choose the right answer.

1. In flat-file databases with numerous files, a common problem is ...
a. data redundancy. b. data entry errors. c. disk space restrictions . d. all the above
2. “Sequential file database” is another term for ...
a. hierarchical database. b. flat-file database.
3. If a database allows any table to have a relationship with any other table, the tables are said to have ...

a. one-to-many relationship. b. parent-child relationship.
4. If a database is made up of a set of tables, it is called ...
a. flat-file database. b. sequential database.
5. Which of the following kinds of database structures is most widely used in organizations today?
a. Flat-file database. b. Relational database.
6. In an object-oriented database, an object can be used to represent ...
a. characteristics. b. nothing.

12. Match each item to the correct statement below.
a. Flat-file database
b. Relational database
c. Object
d. Hierarchical database
e. Network database
f. Attribute
1. Same as a sequential database.
2. Uses only one table.
3. Uses a set of tables.
4. Its tables have a one-to- many relationship.
5. Its tables have a many-to-many relationship.
6. Has characteristics and attributes.
7. Similar to the hierarchical structure.
8. Used in older systems.
9. An object has it.
10. Used for maintaining address lists or inventories.
11. Widely used in business organizations.
12. Characterized by procedures.

13. Discussion .

Give a short explanation of the terms 'Control Structure' and 'Data Structure'

Discuss the following questions in class.
1.Which language is best for Data Structure and algorithms? Most competitive programmers use C++. Can you explain why it is so?
2.How do you start learning DSA?
3.Is learning data structure and algorithms hard?
4.What are Python data structures?
5.How many days does it take to learn data structure and algorithms?

Additional Reading

Read and translate the following text .

C# (/si ʃɑːrp/ *see sharp*) is a general-purpose, multi-paradigm programming language. C# encompasses static typing, strong

typing, lexically coped, imperative, declarative, functional, generic, objectoriented (class-based), and component-oriented programming disciplines. During the development of the

.NET Framework, the class libraries were originally written using a managed code compiler system called *"Simple Managed C"* (SMC).

In January 1999, Anders Hejlsberg formed a team to build a new language at the time called Cool, which stood for "C-like Object Oriented Language". Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the .NET project was publicly announced at the July 2000 Professional Developers Conference, the language had been renamed C#, and the class libraries and ASP.NET runtime had been ported to C#.

Hejlsberg is C#'s principal designer and lead architect at Microsoft, and was previously involved with the design of Turbo

Pascal, Embarcadero Delphi (formerly CodeGear Delphi, Inprise Delphi and Borland Delphi), and Visual J++. In interviews and technical papers he has stated that flaws in most major programming languages (e.g. C++, Java, Delphi, and Smalltalk) drove the fundamentals of the Common Language Runtime (CLR), which, in turn, drove the design of the C# language itself.

James Gosling, who created the Java programming language in 1994, and Bill Joy, a co-founder of Sun Microsystems, the originator of Java, called C# an "imitation" of Java; Gosling further said that "[C# is] sort of Java with reliability, productivity and security deleted." Klaus Kreft and Angelika Langer (authors of a C++ streams book) stated in a blog post that "Java and C# are almost identical programming languages. Boring repetition that lacks innovation," "Hardly anybody will claim that Java or C# are revolutionary programming languages that changed the way we write programs," and "C# borrowed a lot from Java - and vice versa. Now that C# supports boxing and unboxing, we'll have a very similar feature in Java." In July 2000, Hejlsberg said that C# is "not a Java clone" and is "much closer to C++" in its design.

Since the release of C# 2.0 in November 2005, the C# and Java languages have evolved on increasingly divergent trajectories, becoming two quite different languages. One of the first major departures came with the addition of generics to both languages, with vastly different implementations. C# makes use of reification to provide "first-class" generic objects that can be used like any other class, with code generation performed at class-load time.

Furthermore, C# has added several major features to accommodate functional-style programming, culminating in the LINQ extensions released with C# 3.0 and its supporting framework of lambda expressions, extension methods, and anonymous types. These features enable C# programmers to use functional programming techniques, such as closures, when it is advantageous to their application. The LINQ extensions and the functional imports help developers reduce the amount of boilerplate code that is included in common tasks like querying a database, parsing an xml file, or searching through a data structure, shifting the emphasis onto the actual program logic to help improve readability and maintainability (2800)

[https://en.wikipedia.org/wiki/C_Sharp_(programming_language)]

VII NETWORK STRUCTURES

1. Read and translate the text.

A network is a way to connect computers for communication, information exchange, and resource sharing. The four most important benefits of networking are simultaneous access to programs and data, peripheral sharing, streamlined communications, and easier backups. E-mail, videoconferencing, and teleconferencing are examples of the personal communications that can be conducted over a network or the Internet. Networks can be categorized in different ways, such as by geography (how much terrain they cover) or by the use or absence of a central server.

A local area network (LAN) consists of computers that are relatively near one another. A LAN can have a few PCs or hundreds of them in a single building or in several buildings. On a network, data is broken into small groups called packets before being transmitted from one computer to another.

A packet is a data segment that includes a header, payload, and control elements that are transmitted together. The receiving computer reconstructs the packet into the original structure. The payload is the part of the packet that contains the actual data being sent. The header contains information about the type of data in the payload, the source and destination of the data, and a sequence number so that data from multiple packets can be reassembled at the receiving computer in the proper order.

Each LAN is governed by a protocol, which is a set of rules and formats for sending and receiving data. TCP/IP, IPX/SPX, and NetBEUI are examples of network protocols. LANs can be connected by a bridge or router to create a much larger network that covers a larger geographic area. To connect LANs, a gateway may be required to enable them to share data in a way that the different LANs can understand. A wide area network (WAN) is the result of connecting LANs through public utilities. Many networks are built around a central server. The PCs that connect to the server are called nodes. In a file server network, each node has access to the files on the server but not necessarily to files on other nodes. In a client/server network, nodes and the server share the storage and processing workload. A peer-to-peer network is a small network that usually does not include a central server. In a peer-to-peer network, users can share files and resources on all the network's nodes.

2. Answer the following questions.

1. What is a network?

2. What are the most important benefits of networking?

3. What types of the personal communications can be conducted over a network?

4. How can networks be categorized?

5. What is a LAN?

6. What is called a packet? What does it include?

7. What does the payload contain?

8. What information does the header include?

9. How is a network governed?

10. Give the names of some most commonly used protocols.

11. What devices can LANs be connected by?

12. What is a WAN?

13. What is the difference between a file server network and a client/ server network?

14. What is a peer-to-peer network?

3. Tell whether the following statements are true or false. Correct the false ones.

1. E-mail systems enable users to exchange text messages, but not to share data files such as word processing documents.

2. A teleconference is the same thing as a videoconference.

3. Any network within a single building is considered to be a LAN, but if the network extends beyond that building, it is no longer considered a LAN. 4. On a network, data is broken into small groups – called payloads – before being transmitted from one computer to another.

5. TCP/IP, IPX/SPX, and NetBEUI are examples of common networking protocols.

6. If you need to connect two different types of networks, you can use a bridge to translate the data so the networks can communicate.

7. The individual computers on a network are called gateways.

8. In a client/server network, individual computers share the processing and storage workload with a central server.

9. In a peer-to-peer network, individual computers can access one another's resources.

10. All peer-to-peer networks include a network server.

4. Fill in the blanks using the information from the text.

1. A(an) _____ is a way to connect computers so that they can communicate, exchange information, and share resources in real time.

2. In a process known as _____, audio and video signals are transmitted across a network , enabling participants to see and hear one another.

3. A(an) _____ is a network of computers located relatively near each other and connected so that they can communicate with one another.

4. _____ is known as the most commonly used protocol.

5. The individual computers on a network are called _____.

6. A (an) _____ is a data segment that includes a header, payload, and control elements that are transmitted together.

7. A (an) _____ and a (an) _____ are used for connecting LANs to create a much larger network.

8. In a peer-to-peer network, users can _____ files and resources on all the network's nodes.

5. Choose the right answer.

1. A network can include the computers and devices in ... a.

a department. b. a building. c. multiple buildings. d. any of the above.

2. This type of network service is like a cross between the postal system and a telephone answering system.

a. E-mail. b. Peripheral sharing. c. Data sharing. d. Data backup

3. This type of network software allows people in different locations to communicate by typing messages to one another. Each message is seen by all the participants .

a. E-mail. b. Teleconferencing. Videoconferencing. d. All the above

4. LANs and WANs are distinguished from one another mainly by ...

a. size. b. servers used. c. backup methods. d. geography.

5. Within a packet, a payload contains ...

a. information about the data being sent. b. the data itself. c. control elements. d. all the above.

6. This type of device simply forwards data from one LAN to another LAN.

a. Bridge. b. Router. c. Gateway. d. Protocol.

7. When two or more LANs are connected together, the result is a ...

a. client/server network. b. gateway. c. peer-to-peer network. d. wide area network.

8. This type of network allows a node to access files on the server, but not necessarily on other nodes.

a. Node network. b. File server network. c. Peer-to-peer network. d. Router network.

6. Match each item to the correct statement below.

a. LAN b. WAN c. Server d. Node

1. Can be two LANs connected together.

2. Provides a shared storage device.

3. Connects computers that are relatively close together.

4. An individual computer on a network.

5. Can be as small as two or three PCs.

6. Usually covers a large geographic area. a. File-server network b. Client/ server network

7. A way to connect computers.

8. In this type of network users can share files and resources on all the network's nodes.

9. Allows nodes and the server to share the storage and processing workload.

10. Allows each node to have access to the files on the server but not necessarily to files on other nodes.

7. Complete the sentences translating their Ukrainian parts into English.

1. Now it is hard to find a person who uses computers but (ніколи не чула про комп'ютерні мережі).

2. (Спільне користування ресурсами) is the main idea of networks.

3. Imagine there are several computers in an office (і всі вони працюють як із графікою, так і з документами).

4. (Ви марнуєте свій час і час людини, яка працює) on the computer with a printer.

5. (Тепер ви лише натискаєте кнопку друку) on the toolbar of your application.

6. A computer network consists of two or more computers that are (взаємопов'язані з тим, аби розподіляти ресурси, здійснювати обмін файлами або уможливити електронний зв'язок).

7. A local-area network is a computer network that (охоплює локальну територію).

8. (Крім фізичного з'єднання комп'ютерів і пристроїв зв'язку мережна система має функцію встановлення зв'язаної архітектури) that allows almost seamless data transmission while using various equipment types.

9. A LAN typically includes two or more PCs, printers, CD-ROMs and high-capacity storage devices, called file servers, (які дають можливість кожному комп'ютеру в мережі мати доступ до спільного набору файлів).

10. LAN users may also have access to other LANs (або підключатися до глобальних мереж).
11. LANs with different architectures use "gateways" (для перетворення даних, коли вони проходять між системами).

12. Computer networks may link the computers (за допомогою кабелів, оптичних волокон або супутників або модемів

8. Translate into English.

1. Комп'ютерна мережа — це група комп'ютерів та інших пристроїв, з'єднаних разом для спільного використання інформації.

2. Спільними ресурсами можуть бути інформація у файлах, принтери та інші периферійні пристрої.

3. Ви маєте відірвати від роботи людину, яка працює на комп'ютері з принтером, і тільки тоді можете роздрукувати те, що вам потрібно.

4. Тепер вам не треба виконувати зайвої роботи.

5. У комп'ютерній мережі окремими станціями, що називаються «вузлами», можуть бути комп'ютери, термінали або пристрої зв'язку різних типів.

6. Комп'ютери в мережі можуть бути з'єднані за допомогою кабелів, телефонних ліній, радіохвиль, супутників або інфрачервоних променів.

7. Топологія мережі зумовлює її фізичну структуру.

8. Загальноприйнятим максимальним розміром локальної мережі є 1 км2 .

9. Нині існує дві поширені технології з'єднання: Ethernet і маркерне кільце.

10. Локальні мережі схожої архітектури з'єднуються точками переходу, які називаються «мостами».

11. Найкращим прикладом глобальної мережі є Інтернет — сукупність мереж і шлюзів, які з'єднують мільйони користувачів на всіх континентах.

12. Глобальна мережа — це комп'ютерна мережа, яка охоплює велику географічну територію, що включає багато комп'ютерів.

9. Give derivatives of the following words and explain their meanings.

Change, local, wide, convert, browse, engine, connect, allow, establish, cover, dictate, accept, access, function, vary, collect, differ, type, store.

10. Give the opposites of the following words taken from text 6.1 and using them make up sentences of your own.

Transmission, allow, enable, cover, accept, variety, individual, various, different, particular, generally, connect, common, sophisticated.

11. Find in the text the equivalents for:

network topology, be connected through, permit, connect to a network, territory, different (dissimilar), main (major, principal), usually, collection of files, embrace (encompass), now, define a structure, aside from (apart from, besides), be made up of, with the purpose of sharing (for sharing), be located.

12. Read and translate the text.
The difference between web development and web design

Just like with software engineering, you might also hear the terms "web development" and "web design" used interchangeably, but these are two very different things.

Imagine a web designer and web developer working together to build a car: the developer would take care of all the functional components, like the engine, the wheels and the gears, while the designer would be responsible for both the visual aspects - how the car looks, the layout of the dashboard, the design of the seats - and for the user experience provided by the car, so whether or not it's a smooth drive.

Web designers design how the website looks and feels. They model the layout of the website, making sure it's logical, user-friendly and pleasant to use. They consider all the different visual elements: what color schemes and fonts will be used? What buttons, drop-down menus and scrollbars should be included, and where? Web design also considers the information architecture of the website, establishing what content will be included and where it should be placed.

Web design is an extremely broad field, and will often be broken down into more specific roles such as User Experience Design, User Interface Design, and Information Architecture.

It is the web developer's job to take this design and develop it into a live, fully functional website. A frontend developer takes the visual design as provided by the web designer and

builds it using coding languages such as HTML, CSS and JavaScript. A backend developer builds the more advanced functionality of the site, such as the checkout function on an ecommerce site.

In short, a web designer is the architect, while the web developer is the builder or engineer.

[https://en.wikipedia.org/wiki/Web_development]
[https://careerfoundry.com/en/blog/web-development/]

13. Match the following synonyms from the text .

1.to save	a) of the sight or vision
2.complex	b) outline
3.to execute	c) to prepare, to supply
4.design	d) complicated, composed of several elements
5.foundation	e) constituent
6.to provide	f) basis
7.to designate	g) to appoint, to assign
8.component	h) to keep, to preserve
9.visual	i) project, scheme
10.configuration	j) to accomplish

14.Match the following words and phrases from the text with their meanings .

1. Backend	a) a collection of interlinked web pages on the World Wide Web
2. domain	b) all of the behind-the-scenes digital operations that it takes to keep the front end of a website running, such as the coding, style, and plugins
3. Web designer	c) the address for a website as entered into the browser
4. website	d) the part of the website or app that the user sees. If the back end of your website is everything behind-the-scenes, this is what happens onstage
5. frontend	e) system software for creating and managing databases that makes it possible for end users to create, protect, read, update and delete data in a database
6. database management system	f) an IT professional who is responsible for designing the layout, visual appearance and the usability of a website

15.Read the text and decide if the following statements are true or false.

1.Web development is the process of building websites and applications for a private network.

2.Web development can range from developing a simple single static page of plain text to complex web applications, electronic businesses, and social network services.

3.There are two kinds of Web developer specialization: front-end developer and back-end developer.

4.The backend is not very actual because the user doesn't actually see it.

5.Web design is an extremely broad field.

6.Backend developers ensure that everything the frontend developer builds is fully functional.

7.A full-stack developer builds the more advanced functionality of the site, such as the checkout function on an e-commerce site.

8.Sometimes Websites do not rely on database technology.

16. Answer the following questions on the text

1.What does a full-stack developer do?

2.What is a backend developer responsible for? 3) What does a frontend developer do?

3.Can you name any types of web development?

4.What is the difference between web development and web design?

5.How many people can Web development team consist of?

6.What tasks does Web development include?

Additional reading

Read and translate the text about other development tools .

Other web development tools

Web development tools (often called devtools or inspect element) allow web developers to test and debug their code. They are different from website builders and integrated development environments (IDEs) in that they do not assist in the direct creation of a webpage, rather they are tools used for testing the user interface of a website or web application.

Web development tools come as browser add-ons or built-in features in web browsers. Most popular web browsers, such as Google Chrome, Firefox, Internet Explorer, Safari, Microsoft Edge and Opera, have built-in tools to help web developers, and many additional add-ons can be found in their respective plugin download centers.

Web development tools allow developers to work with a variety of web technologies, including HTML, CSS, the DOM, JavaScript, and other components that are handled by the web browser. Due to increasing demand from web browsers to do more, popular web browsers have included more features geared for developers.

Web developers will also use a text editor, such as Atom, Sublime or Visual Studio Code, to write their code; a web browser, such as Chrome or Firefox; and an extremely crucial tool: Git!

Git is a version control system where developers can store and manage their code. As a web developer, it's inevitable that you'll make constant changes to your code, so a tool like Git that enables you to track these changes and reverse them if necessary is extremely valuable. Git also makes it easier to work with other teams and to manage multiple projects

at once. Git has become such a staple in the world of web development that it's now considered really bad practice not to use it.

Another extremely popular tool is GitHub, a cloud interface for Git. While we explain more about what it is and how to use it in our GitHub guide, essentially this tool offers all the version control functionality of Git, but also comes with its own features such as bug tracking, task management and project wikis.

GitHub not only hosts repositories; it also provides developers with a comprehensive toolset, making it easier to follow best practices for coding.

It is considered the place to be for open-source projects, and also provides a platform for web developers to showcase their skills.

Wouldn't it be great if you could edit your HTML and CSS in real time, or debug your JavaScript, all while viewing a thorough performance analysis of your website?

Google's built-in Chrome Developer Tools let you do just that. Bundled and available in both Chrome and Safari, they allow developers access into the internals of their web application. On top of this, a palette of network tools can help optimize your loading flows, while a timeline gives you a deeper understanding of what the browser is doing at any given moment.

Google release an update every six weeks–so check out their website as well as the Google Developers YouTube channel to keep your skills up-to-date. (2500)

VIII. CYBERSECURITY

1. Choose one of the further sections devoted to different cybercrimes and analyze it.

Cyber threats What is a Cyber Attack? A tiny “creeper” virus was the first to show the possibility to connect computers within the network back in early 1970s. Since then, a purely scientific experiment has turned into a criminal industry which has already affected nearly one-third of the world’s computers. Hackers today attack people worldwide roughly every half a minute. A cyber attack is an assault launched by cybercriminals using one or more computers against a single or multiple computers or networks. A cyber attack can maliciously disable computers, steal data, or use a breached computer as a launch point for other attacks. The cyber threat landscape is constantly evolving. As cyberattackers become more skilled and organized, their attacks are becoming more sophisticated as well. In 2020, the most common forms of malware included:

• **Cryptominers**: Malware that uses the victim’s computer to mine cryptocurrency and make a profit for the attacker.

• **Mobile Malware**: Malware targeting mobile devices, including malicious applications and attacks exploiting SMS and social media apps.

• **Botnet Malware**: Malware that infects a system and adds it to a botnet, where it participates in cyberattacks and other illegal activity under the command of the botnet controller.

• **Infostealers**: Malware that collects sensitive information from an infected computer and sends it to the malware operator.

• **Banking Trojans**: Malware that specifically targets financial information and attempts to steal banking website credentials and similar information.

• **Ransomware**: Malware that encrypts the files on a user’s computer and demands payment for the decryption key. Today, organizations face generation V and VI cyber threats. These attackers are aware of the improvements made in enterprise cybersecurity in recent years and have tailored their attacks to bypass and overcome traditional defenses. The modern cyber attack is multi-vector and uses polymorphic code to evade detection. As a result, threat detection and response is more difficult than ever before. To add to the challenge, many organizations are facing a sudden and dramatic shift in how they perform “business as usual”. The COVID-19 pandemic drove many organizations to adopt a mostly or wholly remote workforce, often without adequate preparation. For organizations whose security strategy depended on employees working from the office, adapting to this new way of life is a challenge.

Inside the top cyber threats. Ransomware is malware designed to use encryption to force the target of the attack to pay a ransom demand. Once present on the system, the malware encrypts the user’s files and demands payment in exchange for the decryption key. Since modern encryption algorithms are unbreakable with the technology available, the only way to recover the encrypted files is to restore the data from a backup (if available) or to pay the ransom demand. Ransomware has become one of the most visible and prolific types of malware, and the COVID-19 pandemic provided an environment in which this type of malware has thrived. In recent years, some ransomware variants have also evolved to perform “double extortion” attacks.

Maze, Sodinokibi/REvil, DopplelPaymer, Nemty, and other ransomware variants steal copies of files before encryption, threatening to breach them if the user refuses to pay the ransom demand. While this trend began in late 2019 with Maze, it has continued to grow as more groups adopted

it throughout 2020. Fileless Attacks Antivirus solutions commonly attempt to detect malware on a device by inspecting each file on the device for signs of malicious content. Fileless malware tries to bypass this approach to threat detection by not using a file. Instead, the malware is implemented as a set of commands to functions that are built into the infected computer. This enables the malware to achieve the same objectives, but can make it harder to detect for some defensive solutions. The main differentiator of fileless malware is its lack of files; it performs many of the same functions as traditional malware. For example, FritzFrog – a fileless peer-topeer (P2P) botnet malware detected in August 2020 – is designed to infect systems and mine cryptocurrency.

Phishing. Phishing is one of the most common methods that attackers use to gain access to a target system. In phishing scams, emails or text messages appear to be from a legitimate company asking for sensitive information, such as credit card data or login information. Often, it is easier to trick a user into clicking on a malicious link or opening an attachment than it is to locate and successfully exploit vulnerability in an organization's network. Phishing attacks can achieve a variety of goals, including credential theft, malware delivery, financial fraud, and theft of sensitive data. Phishing has historically been the most common method for cyberattackers to launch a campaign due to its ease of use and high success rate. During the COVID-19 pandemic, this trend only accelerated as cybercriminals took advantage of employees working from outside the office and the climate of uncertainty regarding the virus. The COVID-19 pandemic also amplified the effect of common phishing lures. For example, Black Friday and Cyber Monday are a commonly exploited pretext for phishers, and the rise in online shopping due to COVID-19 made it especially effective in 2020. As a result, the volume of phishing emails doubled in the weeks leading up to Black Friday and Cyber Monday compared to the beginning of the previous month.

Man-in-the-Middle (MitM) Attack. Many network protocols are protected against eavesdroppers by encryption, which makes the traffic impossible to read. A Man-in-the-Middle (MitM) attack bypasses these protections by breaking a connection into two pieces. By creating a separate, encrypted connection with the client and the server, an attacker can read the data sent over the connection and modify it as desired before forwarding it on to its destination. MitM attacks can be defeated using protocols like HTTPS. However, the rise of mobile technologies makes this a more dangerous attack vector. Mobile apps provide little or no visibility to their users regarding their network connections and may be using insecure protocols for communication that are vulnerable to MitM attacks. Malicious Apps Many organizations focus their cybersecurity efforts on computers, but mobile devices are a growing threat to an organization's cybersecurity. As employees increasingly use mobile devices to do their work and access sensitive company data, malicious mobile applications have become a problem too serious to ignore. These applications can do anything that desktop malware can, including stealing sensitive data, encrypting files with ransomware, and more.

In 2020, mobile malware was the second most common type of malware worldwide. The most common mobile malware variants – including xHelper, PreAMo, and Necro – are all Trojans with additional functionality, including ad fraud and click fraud. Mobile malware commonly takes advantage of vulnerabilities in mobile operating systems, like the remote code execution (RCE) vulnerability fixed in a batch of 43 Android security patches in January 2021. Denial of Service Attack Organizations' IT infrastructure and services – like web applications, email, etc. – are critical to their ability to do business. The goal of Denial of Service (DoS) attacks is to deny access to critical services. This can be accomplished by exploiting vulnerability in an application (causing it to crash) or by flooding a system with more data or requests than it is able

to manage (rendering it unable to handle legitimate requests). In some cases, attackers will perform a ransom DoS attack where a ransom payment is demanded to either stop an ongoing attack or prevent a threatened one. During the remote work and learning driven by the COVID-19 pandemic, remote access solutions were a major target of DoS attacks. And during the 2020-2021 school year, Distributed DoS (DDoS) attacks against the education sector increased dramatically. These attacks attempted to render remote learning services unusable or solicited ransoms to prevent or stop the attacks.

Zero-Day Exploit Software contains weaknesses and vulnerabilities, and many of these vulnerabilities reach production, where they are potentially exploitable by attackers. These production vulnerabilities are discovered internally at the company, by external security researchers or by cyberattackers. In the third case, the cyberattackers can exploit these "zero day" vulnerabilities in the system. Until the organization manages to patch the vulnerability – rendering it safe – all users of the system are potentially vulnerable to attack. In 2020, one of the most famous zero-day vulnerabilities was Zerologon, which affected Windows Domain Controllers (DCs). Attackers who exploited this vulnerability could gain complete control over the network managed by the vulnerable DC. Cybercriminals were actively exploiting this vulnerability before many organizations patched it, prompting emergency security directives from the US government for government agencies to apply the patch immediately. .Protecting Against the Top Cyber Threats. While these potential attacks do not make the list of the most common and dangerous cyber threats, they still pose a significant risk. Enterprise security solutions should include the ability to detect, prevent, and remediate attacks using these vectors as well. Enterprise cybersecurity has grown more difficult with the surge in remote work driven by COVID-19. To protect employees working from home (potentially on personally-owned devices) is a new challenge for security teams. These systems connected directly to personal networks and the public Internet are more vulnerable to attacks. As a result, endpoint security – on computers and mobile devices alike – is an even greater priority for enterprise cybersecurity than before. With the wide range of potential cybersecurity threats, organizations require an endpoint detection and response solution how to effectively reveal and protect all of their employees' devices against top cyber threats.

2.Answer the following questions based on the text.

1.How did the evolution from the early "Creeper" virus to modern cybercrime illustrate the changing nature of networked computing?

2.What distinguishes a cyber attack from other forms of digital disruption, and why can a single compromised device pose a broader threat?

3.Why is the modern cyber threat landscape described as "constantly evolving," and what role do attacker skill and organization play in this evolution?

4.Compare cryptominers and botnet malware in terms of their objectives and impact on infected systems.

5.Why is ransomware considered one of the most visible cyber threats, and how has the tactic of "double extortion" changed the attacker–victim dynamic?

6.Explain how fileless malware challenges traditional antivirus detection methods. Why does its design make it harder to detect?

7.Why is phishing still one of the most successful attack vectors despite advances in cybersecurity technologies?

8.How did the COVID-19 pandemic amplify both the effectiveness and frequency of phishing attacks?

9.Describe how a Man-in-the-Middle (MitM) attack works and why mobile technologies have increased the risks associated with this type of attack.

10.In what ways do malicious mobile applications pose risks comparable to desktop malware, and why are mobile devices a growing concern for organizations?

11.What are the main differences between Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks, and why did remote services become prime targets during the pandemic?

12.Why are zero-day exploits particularly dangerous, and what does the Zerologon vulnerability reveal about the risks of delayed patching?

3. Say whether the following statements are true or false. Correct the false ones.

1. The first computer virus demonstrated that networked computers could be used for both scientific experimentation and criminal activity.

2. A cyber attack may involve using a compromised computer as a platform to launch further attacks against other systems.

3. Modern cyber attacks are becoming less effective because traditional security defenses have improved significantly.

4. Cryptominers directly steal financial credentials from victims in order to access their bank accounts.

5.Generation V and VI cyber threats commonly rely on multi-vector attacks and polymorphic code to evade detection.

6. Ransomware victims can always recover their encrypted files without paying the ransom by breaking modern encryption algorithms.

7. Some ransomware variants now threaten to leak stolen data in addition to encrypting files.

8. Fileless malware avoids detection by operating without stored files and using built-in system functions instead.

9.Phishing attacks are considered ineffective because exploiting software vulnerabilities is easier than manipulating users.

10.Mobile applications generally provide users with clear visibility into network security, reducing the risk of Man-in-the-Middle attacks.

11.Denial of Service attacks can target both software vulnerabilities and system capacity limitations.

12.Zero-day exploits remain dangerous until vulnerabilities are patched, regardless of whether users are aware of the risk.

4. Match synonyms to the words from the following text

1 sophisticated	***2 a victim***	***3 to amplify***	***4 prolific***	***5 to evade***
6 extortion	***7 to tailor***	***8 legitimate***	***9 to implement***	***10 to flood***
a blackmail	*b to increase*	*c to execute*	*d to modify*	*e complex*
f to avoid	*g to overwhelm*	*h a prey*	*i productive*	*j valid*

5. Match the words from the text and their definitions.

1	an assault	a	to break or act contrary to a law
2	malicious	b	a secret listener to private conversations
3	to remediate	c	to ask for or try to obtain something from someone a ransom
4	a ransom	d	wrongful or criminal deception intended to result in financial or personal gain
5	to breach	e	the sum or price paid or demanded
6	to encrypt	f	a sudden, violent attack
7	vulnerability	g	a sudden and great increase
8	a fraud	h	openness or susceptibility to attack or harm
9	a lure	i	intentionally harmful
10	eavesdropper	j	something that is used to tempt a person or animal to do something
11	to render	k	to encode
12	to solicit	l	to cause to become
13	a surge	m	to lessen the effect of

6. Fill in the gaps with a suitable word or word combination .

Cyber Threats in the Modern Digital Environment

Over the past decades, cybercrime has transformed from isolated technical experiments into a highly organized global phenomenon. Today, digital attacks are rarely random; instead, they are carefully planned operations that exploit both technological weaknesses and human behavior. A cyber attack typically involves one or more systems being compromised in order to **(1)** _____, **(2)** _____, or secretly provide attackers with a platform for further activity.

One of the defining characteristics of modern cyber threats is their increasing **(3)** _____. Attackers now use multiple techniques simultaneously, often combining social engineering with advanced malicious software. This approach allows them to bypass **(4)** _____ security mechanisms that were designed to detect simpler forms of malware.

Among the most damaging forms of malware are those that restrict access to data through **(5)** _____, forcing victims to choose between permanent data loss and financial payment. In recent years, attackers have escalated these attacks by introducing **(6)** _____, in which sensitive information is stolen before systems are locked.

Not all malicious software relies on files stored on a device. Some attacks operate by abusing **(7)** _____ already present in operating systems, making detection significantly more difficult. At the same time, deceptive communication remains one of the most effective attack methods, as users are often manipulated into **(8)** _____ or sharing confidential information.

The shift toward remote work has expanded the attack surface, exposing systems connected directly to **(9)** _____ networks. Mobile devices have become particularly attractive targets due to **(10)** _____ visibility and inconsistent security controls.

Another serious threat involves disrupting access to essential services. By exploiting vulnerabilities or overwhelming systems with traffic, attackers can **(11)** _____ critical infrastructure. In some cases, such attacks are accompanied by **(12)** _____ demands.

Finally, vulnerabilities unknown to software vendors present extreme risks. These **(13)** _____ flaws can be exploited until organizations manage to **(14)** _____ them, leaving entire networks exposed. As a result, effective protection increasingly depends on **(15)** _____ solutions capable of detecting and responding to threats in real time.

7. Read the text. Complete it with the missing sentences. One sentence is extra.

A. Attack vectors are contained.

B. Risks are well-known.

C. Complex Passwords Cannot Be Cracked.

D. Cybercriminals are outsiders.

E. My industry is safe.

Dangerous cybersecurity myths The volume of cybersecurity incidents is on the rise across the globe, but misconceptions continue to persist, including the notion that: __(1)__ In reality, cybersecurity breaches are often the result of malicious insiders, working for themselves or in concert with outside hackers. These insiders can be a part of well-organized groups, backed by nation-states. __(2)__ In fact, the risk surface is still expanding, with thousands of new vulnerabilities being reported in old and new applications and devices. And opportunities for human error – specifically by negligent employees or contractors who unintentionally cause a data breach – keep increasing. __(3)__ Cybercriminals are finding new attack vectors all the time – including Linux systems, operational technology (OT), Internet of Things (IoT) devices, and cloud environments. __(4)__ Every industry has its share of cybersecurity risks, with cyber adversaries exploiting the necessities of communication networks within almost every government and private-sector organization. For example, ransomware attacks are targeting more 141 sectors than ever, including local governments and non-profits, and threats on supply chains, ".gov" websites, and critical infrastructure have also increased.

Word bank

legitimate company · trick people into · appear to be · competition · Victims · install malware · Be wary · confirm · click on links · genuine

8. Fill in the gaps with one suitable word or phrase.

legitimate company · trick people into · appear to be · competition · Victims · install malware · Be wary · confirm · click on links · genuine

Phishing is an attempt to **(1)** _____ handing over their personal information such as bank details or passwords by posing as a **(2)** _____. Attackers may send out e-mails, text messages or make phone calls that **(3)** _____ from banks or service providers. Often the victim will be told they need to take part in a **(4)** _____ or account reactivation, and may even be told their account will be closed if they refuse. **(5)** _____ are taken to a fake web site, which can look just like the real thing. They may be asked to enter personal information, which can be used by criminals, or the fake web site may **(6)** _____, which then can steal personal data.

Tricking people into handing over their passwords or other private information is still one of the biggest ways criminals get hold of personal details. But there are ways you can protect yourself. **(7)** _____ of calls, emails or text messages from unknown sources, especially ones that don't address you by name. Big companies will never ask you to **(8)** _____ your personal information in emails or phone calls – and be suspicious of emails and text messages that ask you to **(9)** _____. If you are ever not sure whether the message is **(10)** _____, you could always call the company.

Additional reading

Cybersecurity

Together with the evolvement of new technologies and the growth of a number of devices, the amount of possible cyber threats continually increases. Usually, the main goal of cyberattacks is accessing, changing, or destroying sensitive information, and extorting money from users or interrupting normal business processes. These attacks are ceaseless with growth in their frequency and sophistication. Protection of computers, networks and clouds is commonly provided by the means of nextgeneration firewalls, DNS filtering, malware protection, antivirus software, and email security solutions. Cybersecurity is one of the key trends for organizations, whose business processes are based on data-driven technologies. Much more attention is being paid to privacy and data protection since the European Union's General Data Protection Regulations (GDRP) has been signed. The latest cybersecurity threats are phishing, ransomware, cryptojacking, cyber-physical attacks, state-sponsored attacks, and IoT attacks. Data breaches are the biggest cybersecurity concern, and trading personal data remains quite lucrative at the black market. According to the report of cybersecurity unit ElevenPaths of the multinational communications company Telefonica, in 2020, such main technologies as Ransomware attacks, Cloud Computing, Machine Learning, Phishing attacks, Open Banking and Mobile Malware, 5G will be related to cyberattacks. One of the ways to provide security, especially for IoT devices is blockchain. This technology was initially launched for storage and sending the first cryptocurrency, Bitcoin. Blockchains are distributed networks that can be used by millions of users throughout the world. Data is managed by a cluster of computers not owned by any single entity, thus data can be only added, not changed or copied. All data is secured through cryptography. During 2019 FedEx, IBM, Walmart, and Mastercard continued investment in blockchain and it is likely to start to show real-world results, encouraging its adoption. On these days, when thousands of people are forced to work remotely, volumes of private data may

become totally vulnerable or at least not protected in a proper way. 250 This emerging issue may give another impetus to the development of this technology. Cybersecurity may also be applied in crowdfunding, governance, supply chain auditing, file storage, prediction markets, protection of intellectual property, IoT, neighborhood microgrids, and stock trading.

IX. VIRTUAL REALITY. ARTIFICIAL INTELLIGENCE

1.Read the text and add more spheres in which VR can be used. Choose the correct option.

VR could revolutionize education by ***enable / enabling*** students ***to learn / learning*** in an immersive, experiential way, from anywhere in the world. VR provides the opportunity ***to democratize / democratizing*** education by ***open / opening*** up 169 opportunities to students of all backgrounds, which may not have been possible before. For example, Victory XR has partnered with Engage ***to provide / providing*** digital twin campuses ***to enable / enabling*** students ***to learn / learning*** in live, interactive classes from the brightest minds in the world. Other companies like Tech Row enable students ***to go / going*** on a space mission to Pluto, explore Antarctica, and ***experience / experiencing*** the wonders of Machu Picchu. Field trips to the Colosseum and Ancient Rome can be completed from the classroom, and ***be / being*** taken on a journey of the human body as a white blood cell is not an impossibility any more! Since VR enables individuals ***to meet / meeting*** in places virtually, it's no surprise that the pandemic brought a rise in VR events, conferences and meetings. Platforms such as Glue, Arthur and Meeting Room can be used ***to hold / holding*** collaborative, interactive meetings with colleagues from anywhere in the world. You can put on your headset in London, and meet virtually with your colleagues in New York and Madrid, and connect and work with them as if you were all in the same room. With collaboration tools such as whiteboards and freehand 3D drawings, they help remote or hybrid meetings ***become / becoming*** as good as face-to-face meetings, without the time and cost needed ***to travel / travelling***.

2.Read the text and choose the option .

Virtual Reality Technology NASA has big hopes for virtual reality technology. The agency is developing a suite of virtual reality environments at Goddard Spaceflight Center in Maryland, that could be used for everything from geological research to repairing orbiting satellites. One displays fiery ejections from the Sun. In another, scientists can watch magnetic fields pulse around the earth. A virtual rendering of an ancient lava tube in Idaho makes scientists ***feel / to feel*** like they're standing at the bottom of an actual cave. "I think, and I hope, this can be extremely useful for NASA scientists, " explains NASA engineer Thomas Grubb, who manages the program. The goal, he says, is to ***scale up / scaling*** up the use of virtual reality technology in NASA labs, and ***go / going*** beyond public applications like the Mars immersion program that allows users ***exploring / to explore*** the Martian surface. For example, NASA volcanologist Brent Garry is hoping that virtual visits to a rock formation in Idaho can help him ***planning / plan*** research trips in real life. That same VR environment also allows users to measure / measuring distances and leave notes in the landscape. "You know, it's cheaper to have people ***go / to go*** to a lava tube in VR than to actually fly them out there for two weeks," says Grubb. Another application in development could allow technicians to repair satellites. People on earth could watch in real-time as they manipulate actual tools in space. If the repairs are successful, satellites that would have died when their batteries did could keep ***working / to work*** instead. "All of these things can save a lot of money or time, or just enable new things, " says Grubb. And Grubb has stumbled upon a new talent source to help develop the pilot programs: young students, some of them still in high school. "I went into this [thinking] 'I'll take a couple of interns or whatever,'" he says, imagining he'd get a single college student to help with some coding. But he says when he posted the job, "I got all these amazing students coming back. And I was like 'I want more than this.' I ended up with five [interns]." One of them was a high school senior Jackson Ames. In addition to taking some computer science classes in school, Ames plays video games. "A lot of the games require strategy and teamwork. One of my favourites is called 'Onward'," he explains. Onward is a war simulation game. It's supposed ***to make / making*** players ***to feel / feel***

like they're soldiers fighting a battle. You play with a VR headset covering your eyes and a controller in each hand. "It's much more realistic than anything else," says Ames. "It adds a whole new layer." Ames plays the game many times a week, which gives him an intuitive sense of what works, and what doesn't, in VR. Young people also bring certain ease with learning new technologies. Stewy Slocum, a 17-year-old college freshman who worked on the lava tube simulation, says video games got him interested in virtual reality programming too. But he's not that into gaming anymore – that was more of a high school thing – and he hadn't had much experience with VR technology before he arrived at NASA. Still, he quickly learned how to use the VR system he was working with. That is often not the case for some more experienced researchers. "There were definitely some older people who tried [the virtual reality system] and struggled at first, because they're not used ***to have / having*** all ten fingers working at once," says Garry. "They're like 'Am I going to fall? Am I going to trip on something?" Once people navigate the initial learning curve, ***exploring / to explore*** virtual reality can go from alarming to fun pretty quickly. The growing popularity of virtual reality systems like the HTC Vibe and Oculus Rift for video games and education has made virtual reality technology more mainstream, but that can actually act as an obstacle ***to use / using*** VR technology for research, Grubb says. "People think 'This is too cool and too much fun. How can this be work?'," he laughs. "We're trying ***to show / showing*** scientists that, 'Hey, this technology has a lot to offer".

3.Read about one more of the applications of Virtual Reality and complete the gaps by choosing the correct options.

Virtual reality makes users want to exercise Businesses are (1) ____ more uses for Virtual Reality (VR) as the technology develops. VR is no longer only for gaming or enjoyment. An American company (2) ____ Blue Goji is using VR to improve health by making exercise more fun. (3) ____ company demonstrated its workout machine, called the Infinity treadmill, at the recent event in Austin. A person (4) ____ the treadmill wears a virtual reality headset when exercising. Before starting, the user is connected to a belt to prevent falls. Then, the user plays a VR game (5) ____ running on the machine. The game can transport the user into the virtual world, (6) ____ he or she can be racing against (7) ____ people.

1 A. found B. finding C. founding D. find

2 A called B. which called C. calling D. cold

3 A ---- B. an C. a D. the

4 A using B. who using C. used D. was using

5 A before B. after C. and D. while

6 A where B. which C. that D. who

7 A really B. reality C. virtually D. virtual

The cost of the hardware and computer software program is $12,000. That is a lot of money for most people. But the virtual reality treadmill is ideal (8) ____ places where people go to exercise, like a high-end gym or recreation center. It can also be used for physical therapy or rehabilitation. Users who tested the treadmill while wearing the VR headset (9) ____ different

experiences. Very often the first time users feel lost, but the more you do it, the more you get used to it. After carefully studying the (10) ____ experiences, Blue Goji plans to begin selling the Infinity treadmill to the public in 2019.

8 A from B. into C. for D. of

9 A had B. has C. having D. has had

10 A. users's B. users' C. user'es D. users'es

4. Put these words into the spaces in the paragraphs below.

a) overtake b) giant c) devices d) order e) twice f) reality g) yet h) creatures

The Japanese games (1) _____ Nintendo has released an app that is taking the world by storm. The next big thing, and new Internet sensation, is an augmented (2) _____ game called Pokemon Go. Players must physically move around the real world in (3) _____ to capture mystical (4) _____ called Pokemon (short for pocket monsters). It was only officially released last week, in the USA, New Zealand and Australia. It is about to (5) _____ Twitter in the number of daily active users and it hasn't even been launched globally (6) _____. Analysts report that in just 48 hours, Go was installed on 5.6 per cent of all Android (7) _____ in the USA. On average, users are spending (8) _____ as much time on Go than on apps like Snapchat.

i)impact j) burgle k) spots l) top m) unsuspecting n) fix o) infect p) tracks

Pokemon Go has already jumped to the (9) _____ of the App Store and Google Play Store. It is also having an unprecedented social (10) _____. Hackers have targeted illegal copies of the app to (11) _____ millions of smart phones. Other criminals have used the game to lure (12) _____ players to go to a location to collect a Pokemon character and then rob them or (13) _____ their empty house. The app (14) _____ your location via GPS as you walk around looking for Pokemon. The website PCmag.com advised: "Don't go walking around neighborhoods late at night for your Pokemon (15) 179 _____ . If you can, collect Pokemon in public, crowded areas, we recommend doing that instead of shady (16) _____ at two a.m."

5.Read the text and change the words in bold so they suit the sentence .

The **(precedent)** growth in the number of sentient entities and their human-level (**compete)** (Level 4 AGI) should bring about significant changes, both beneficial and dangerous. Various AI safety measures may help mitigate some of these dangers. However, these efforts do not need to be considered in crude explorations if the AI system design is straightforward enough to fall far short of AGI on its own . AI has made tremendous strides in learning skills that are vastly more sophisticated than the anticipated calculations of human beings. (**Dissimilar)**, it seems increasingly likely that soon it will be possible to create computers capable of sparking a renaissance in the development of mathematical theories.

Intelligent agents include 'sentience', also called an "inner life," which refers to a kind of awareness or (**conscious)**. Due to a historical accident, it seems unlikely that code executing in

brain-like analogs will cause(**reason)**concern. Awareness comes in degrees, and at the level of intelligent agents—specifically at the human level—it is likely textual. This (**occure)** exists downstream from script-like thought and appears to be a mere simulation, a movie of thought. Having it can shape feelings; however, any feeling without thought is challenging to conceive. The (**able)** to formulate words and alter outputs without thought may lead to ideation delusions, but this process is **(ordinarily)** strenuous. Nevertheless,(**insufficient**)high levels are worthy of concern . Humans find themselves in increasingly complex epochs that promote development but not necessarily well-being. It is difficult to believe that evolution-designed (**be)** to serve this purpose. This can be an enviable position for AI. There is barely one level at which they can act or reverse it, and on which universes can be significantly improved.

The rapid development of artificial intelligence (AI) is transforming many industries. Surveys show that over three-quarters of organizations already use AI in at least one business area, while generative AI could boost global GDP by several percentage points and put millions of jobs at risk of automation. This narrative review compiles insights from the academic literature, industry reports, and policy documents to emphasize the (**transform)** potential of AI across healthcare, manufacturing, finance, education, governance, and transportation.

Despite promising productivity gains, AI adoption raises significant socio-economic and ethical concerns. Estimates suggest that around 40% of global jobs are exposed to AI-driven automation, risking greater **(equal**) if benefits are not broadly shared. Studies also document the risk of algorithmic bias, hallucinations and lack of(**transparent)** in AI systems, underscoring the need for robust oversight, privacy protection and fairness.

This review emphasizes the importance of interdisciplinary collaboration between researchers, industry and policymakers. Future research should conduct comparative sectoral analyses, develop methods to evaluate socio-economic impacts over time, and design (**regular)** frameworks that balance innovation with ethical and legal safeguards. As AI technologies continue to **evolve**, sustained investment in human capital, education and inclusive policies will be crucial to **(sure)** that AI serves the public interest. This review is limited by its analytical and narrative approach, which does not rely on primary data or systematic meta-analysis. While this enables a broader (**society)** perspective, it also implies constraints in empirical (**valid)** and the sector-specific granularity expected in technical studies. Future research could complement this work with data-driven analyses and longitudinal studies across specific domains.

6. Answer the questions .

1.What level of artificial intelligence is described as having human-level competence in the text?

2.Why does the author suggest that some AI systems may not pose serious risks despite their advanced capabilities?

3.How does the text define “sentience” in the context of intelligent agents?

4.Why is awareness described as “a simulation” or “a movie of thought”?

5.What concern is raised about the ability of AI systems to generate outputs without genuine thought?

6.According to the text, why might AI development be considered both beneficial and dangerous?

7.What role does AI play in transforming global industries such as healthcare and finance?

8.What economic risks are associated with the widespread adoption of AI technologies?

9.Why does the author emphasize the need for interdisciplinary collaboration in AI development?

10.What limitations does the review acknowledge regarding its own methodology?

7.Decide whether each statement is true or false according to the text. Correct the false ones .

1.The text argues that all advanced AI systems inevitably possess consciousness.

2.AI has already surpassed humans in all forms of reasoning and creativity.

3.The author suggests that awareness may exist on a spectrum rather than as a binary trait.

4.The development of AI is expected to have a negative impact only on low-skilled workers.

5.Algorithmic bias and lack of transparency are identified as major ethical concerns.

6.AI systems are currently capable of fully understanding emotions in the same way humans do.

7. The review emphasizes the importance of regulation and ethical safeguards in AI development.

8.The text claims that economic benefits of AI will automatically be shared equally across society.

9. The review relies primarily on empirical datasets and large-scale experiments.

10. Future research is encouraged to include longitudinal and data-driven studies.

8.Virtual Reality (VR) and Augmented Reality (AR): A Comparative Analysis. Read the text and find the information you did not know about AR.

Virtual Reality (VR) and Augmented Reality (AR) represent two of the most transformative technologies of the digital age, reshaping how humans interact with information, environments, and each other. Although they are often discussed together under the umbrella of immersive technologies, VR and AR differ fundamentally in their goals, technical design, and applications. Understanding these differences is essential for evaluating their current impact and future potential.

At its core, **Virtual Reality** creates a fully artificial environment that replaces the user's physical surroundings. Using head-mounted displays such as the Meta Quest or HTC Vive, users are immersed in a computer-generated world that responds to their movements in real time. This immersion is achieved through stereoscopic visuals, spatial audio, and motion tracking, which together create the illusion of physical presence inside a digital environment. The defining feature of VR is *total immersion*: the real world is temporarily excluded, allowing users to experience scenarios that would otherwise be impossible, dangerous, or impractical.

In contrast, **Augmented Reality** overlays digital information onto the real world rather than replacing it. AR systems, such as Microsoft HoloLens or smartphone-based applications like Pokémon GO, enhance the user's perception of their environment by adding virtual objects, text, or animations that appear to coexist with physical reality. Unlike VR, AR does not isolate the user from their surroundings; instead, it enriches real-world perception by blending physical and digital elements in real time.

These fundamental differences lead to distinct applications. VR excels in fields that benefit from total immersion, such as simulation-based training, gaming, therapy, and virtual tourism. For example, pilots can practice emergency procedures in a controlled virtual cockpit, while medical students can rehearse complex surgeries without risk to patients. In entertainment, VR allows users to inhabit fictional worlds, offering a level of presence that traditional media cannot replicate. However, VR's immersive nature also presents challenges, including motion sickness, physical disorientation, and limited long-term usability due to headset discomfort.

AR, on the other hand, is particularly effective in contexts where real-world interaction is essential. In industrial settings, AR can project step-by-step instructions directly onto machinery, improving efficiency and reducing human error. In medicine, surgeons can view patient data or 3D anatomical models overlaid onto the body during procedures. In education, AR enables interactive learning experiences by visualizing abstract concepts—such as molecular structures or historical reconstructions—within the learner's physical environment. Because AR maintains awareness of the real world, it is generally safer and more practical for everyday use.

Another key distinction lies in technological requirements and accessibility. VR typically demands powerful hardware, including headsets, sensors, and sometimes dedicated physical space. As a result, it remains relatively expensive and less accessible to the general public. AR, by contrast, can function on widely available devices such as smartphones and tablets, significantly lowering the barrier to entry. This accessibility has contributed to the rapid integration of AR into commercial, educational, and consumer applications.

Despite their differences, VR and AR are not competing technologies but rather complementary ones. The convergence of both has given rise to the concept of **Mixed Reality (MR)**, which allows digital and physical objects to interact in real time. As hardware becomes more compact and software more sophisticated, the boundaries between VR, AR, and MR are likely to blur further, creating hybrid experiences that combine immersion with contextual awareness.

In conclusion, VR and AR represent two distinct yet interconnected approaches to reshaping human experience through technology. VR offers total immersion and transformative experiences within artificial environments, while AR enhances reality by integrating digital content into the physical world. Each technology has unique strengths, limitations, and applications, making them suitable for different purposes. As technological innovation continues, both VR and AR will play increasingly significant roles in education, industry, entertainment, and everyday life, ultimately redefining how humans perceive and interact with reality itself.

9.Complete the sentences using the correct form of the word given in brackets.

1.AR increases user _____ by making digital content more engaging. (INTERACT)

2.One major _____ of AR is its ability to provide instant visual support. (ADVANTAGE)

3.Overuse of AR may lead to reduced _____ among users. (CONCENTRATE)

4.The _____ of AR technologies raises serious ethical concerns. (DEVELOP)

5.Many experts question the long-term _____ of relying on AR systems. (EFFECT)

6.High costs limit the _____ of AR devices for ordinary users. (ACCESS)

7.AR can significantly _____ workplace efficiency. (IMPROVEMENT)

8.Data _____ remains one of the biggest challenges in AR applications. (SECURE)

9.Some educators worry about students' growing _____ on digital tools. (DEPEND)

10.Despite its benefits, AR still faces technical and social _____. (LIMIT)

10. Put the proper words into the following sentences:

fiber-optic, swoop, go astray, clutching, gear, to one's mind content, enhance, cyberspace, eye phones.

1. Virtual reality is sometimes called...

2.3-D ... are really individual computer screens for the eyes.

3 .Virtual reality can ...possibilities of the disabled.

4. The manual ...box allows you to slow down without braking, while the automatic one doesn't.

5. Cyberspace allows everybody to change it...

6. The letters wrongly addressed...

7. ... unknown things may cause an accident.

8. By the end of the 20th century metal wires had been replaced by ... ones.

9. In one of the s the ...the NATO has lost their most expensive fighter.

11. Guess the meaning of the italicized words. Make your own sentences using these words.

1. Virtual reality ***straddles*** the foggy boundary between fantasy and fact.

2. Imagine a place and you'll be able to step into it. ***Conjure*** up a dream and you'll be able to fly through it.

3. He's ***launched*** one of the first computers to mass-produce virtual reality systems.

4. Virtual reality techniques have been used to make a 3D model of the planet Mars. There are, of course, more ***down-to-earth*** applications. Virtual reality models of urban landscapes are allowing urban planners to redesign Main Street without leaving the room.

5. We're now reaching a point where the simulations are so realistic that the line between playing a game or a simulation and actually blowing people up is becoming ***blurred.***

Additional reading

Read and translate the following text.

Extended Reality Extended Reality (XR) technologies aren't groundbreaking at this point, but since recently they are actively adopted in entertainment to create more immersive digital experiences (Snapchat filters, Pokemon Go-style games). XR most commonly includes virtual, augmented, and mixed reality. Virtual reality (VR) is the use of computer technology to create a simulated environment using headsets that blend out the real world, instead of watching on a display, immersing a person in a digital 3D environment. Unlike VR, augmented reality (AR) overlays digital objects onto the real world via smartphone screens or displays, it does not create the whole artificial environments to replace real with a virtual one. Mixed reality (MR) is an extension of AR, which means users can interact with digital objects placed in the real world (think playing a holographic piano that you have placed into your room via an AR headset). The influence of virtual and augmented reality will grow in training and simulation, as well as offering new ways to interact with customers, providing new shopping experience (especially during the lockdown, when thousands of people are shopping online). In 2020, the workplace will become smaller with the growth of virtual collaboration tools (Slack, Zoom), which began replacing the need for costly office space. AR and VR in 2020, however, will bring workers virtually into the physical space.

Read and translate the following text.
Artificial Intelligence

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision.

As the hype around AI has accelerated, vendors have been scrambling to promote how their products and services use AI. Often what they refer to as AI is simply one component of AI, such as machine learning. AI requires a foundation of specialized hardware and software for writing and training machine learning algorithms. No one programming language is synonymous with AI, but a few, including Python, R and Java, are popular.

In general, AI systems work by ingesting large amounts of labeled training data, analyzing the data for correlations and patterns, and using these patterns to make predictions about future states. In this way, a chatbot that is fed examples of text chats can learn to produce lifelike exchanges with people, or an image recognition tool can learn to identify and describe objects in images by reviewing millions of examples.

AI programming focuses on three cognitive skills: learning, reasoning and self-correction.

Learning processes. This aspect of AI programming focuses on acquiring data and creating rules for how to turn the data into actionable information. The rules, which are called algorithms, provide computing devices with step-by-step instructions for how to complete a specific task.

Reasoning processes. This aspect of AI programming focuses on choosing the right algorithm to reach a desired outcome.

Self-correction processes. This aspect of AI programming is designed to continually fine-tune algorithms and ensure they provide the most accurate results possible.

AI is important because it can give enterprises insights into their operations that they may not have been aware of previously and because, in some cases, AI can perform tasks better than humans. Particularly when it comes to repetitive, detail-oriented tasks like analyzing large numbers of legal documents to ensure relevant fields are filled in properly, AI tools often complete jobs quickly and with relatively few errors.

This has helped fuel an explosion in efficiency and opened the door to entirely new business opportunities for some larger enterprises. Prior to the current wave of AI, it would have been hard to imagine using computer software to connect riders to taxis, but today Uber has become one of the largest companies in the world by doing just that. It utilizes sophisticated machine learning algorithms to predict when people are likely to need rides in certain areas, which helps proactively get drivers on the road before they're needed. As another example, Google has become one of the largest players for a range of online services by using machine learning to understand how people use their services and then improving them. In 2017, the company's CEO, Sundar Pichai, pronounced that Google would operate as an "AI first" company.

Today's largest and most successful enterprises have used AI to improve their operations and gain advantage on their competitors.

Artificial neural networks and deep learning artificial intelligence technologies are quickly evolving, primarily because AI processes large amounts of data much faster and makes predictions more accurately than humanly possible.

While the huge volume of data being created on a daily basis would bury a human researcher, AI applications that use machine learning can take that data and quickly turn it into actionable information. As of this writing, the primary disadvantage of using AI is that it is expensive to process the large amounts of data that AI programming requires.

Advantages

1.Good at detail-oriented jobs;
2.Reduced time for data-heavy tasks;
3.Delivers consistent results; and
4.AI-powered virtual agents are always available.

Disadvantages

1.Expensive;
2.Requires deep technical expertise;
3.Limited supply of qualified workers to build AI tools;
4.Only knows what it's been shown; and
5.Lack of ability to generalize from one task to another. (3500)

[https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence]

Answer the following questions:

1.How does AI work?
2.Why is artificial intelligence important?
3.What are the advantages and disadvantages of artificial intelligence?

Artificial Intelligence

The term Artificial Intelligence (AI) is not all that new for the information technology sphere, and it's still increasing its impact. Generally, AI refers to the usage of algorithms to solve specific tasks by studying large amounts of data to make generalizations and/or some statistic estimations. These algorithms enable a computer to "behave" just like a human brain. AI technologies are already used by 77% of consumers, daily appear new applications. According to PwC analytics, one of the leading consulting companies, by 2030 AI Products will contribute more than $15.7 trillion to the global economy. A number of technological innovations such as data processing, and face and speech recognition have become possible due to AI. AI software is a vast area to which one can include AI platforms, chatbots, machine learning (algorithm category consisting of various libraries and frameworks) and deep learning (using artificial neuron networks), and analytics tasks for financial services. It's expected that in 2020 more tailored applications and services for specific or specialized tasks will be offered. Acceptance of AI chatbots by service is growing in such areas as online retail, healthcare, telecommunications, banking, financial advice, insurance, dealership and government. One of the chatbots' specific purposes for business is automation, which also makes jobs simpler for employees (but won't cause job shortening as it's sometimes considered). Within AI trend can be outlined embedded AI and Machine learning as a service (AIaaS and MLaaS) sub-trends, which include such types as bots and digital assistance, cognitive computing APIs, machine learning frameworks, and fully-managed machine learning services. As-a-service platforms will be used for the creation of AI applications because of the high cost of AI-based systems. They will enable feeding in our own data and paying for the algorithms or compute resources as we use them. Among well-known AIaaS are Amazon Web Services (AWS), Microsoft Azure, Google Cloud, and IBM Cloud. Under conditions of COVID-19 spreading all over the world, AI may contribute to the forecasting of consumers' wishes, which became hardly predictable, and to help 248 businesses organize effective logistics. Chatbots may provide clients' support 24/7, one of the 'must-have' during the lockdown. The popularity of machine learning may grow due to the necessity of the improvement of algorithm-moderators of posts and visual content in social networks (which sometimes block reliable sources of information about coronavirus and don't detect fakes). AI may be included to long-lasting trends due to its efficiency, amazing speed and accuracy, less inclined to make errors in comparison with a human, as well as the ability to work 24/7 in dangerous and risky situations. It also has some threats, such as cost, restriction because of the code limits, and machine dependency.

X. COMPUTER TERMS GLOSSARY

abstraction. The separation of the logical properties of data or function from its implementation in a computer program. See: encapsulation, information hiding, software engineering.

access time. The time interval between the instant at which a call for data is initiated and the instant at which the delivery of the data is completed.

accuracy.) (1) A qualitative assessment of correctness or freedom from error. (2) A quantitative measure of the magnitude of error. Contrast with precision. (3) The measure of an instrument's capability to approach a true or absolute value. It is a function of precision and bias.

algorithm. (1) A finite set of well-defined rules for the solution of a problem in a finite number of steps. (2) Any sequence of operations for performing a specific task.

analog device. A device that operates with variables represented by continuously measured quantities such as pressures, resistances, rotations, temperatures, and voltages.

application software. Software designed to fill specific needs of a user; for example, software for navigation, payroll, or process control. Contrast with support software; system software.

architectural design. (1) The process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system. (2) The result of the process in (1).

archival database. An historical copy of a database saved at a significant point in time for use in recovery or restoration of the database.

archive file. A file that is part of a collection of files set aside for later research or verification, for security purposes, for historical or legal purposes, or for backup.

arithmetic logic unit. The [high speed] circuits within the CPU which are responsible for performing the arithmetic and logical operations of a computer.

array. An n-dimensional ordered set of data items identified by a single name and one or more indices, so that each element of the set is individually addressable; e.g., a matrix, table, or vector.

assembler. A computer program that translates programs [source code files] written in assembly language into their machine language equivalents [object code files]. Contrast with compiler, interpreter.

assembly language. A low level programming language, that corresponds closely to the instruction set of a given computer, allows symbolic naming of operations and addresses, and usually results in a one-to-one translation of program instructions [mnemonics] into machine instructions. See: low-level language.

asynchronous. Occurring without a regular time relationship, i.e., timing independent.

BIOS. basic input/output system.

BASIC. An acronym for Beginners All-purpose Symbolic Instruction Code, a highlevel programming language intended to facilitate learning to program in an interactive environment.

basic input/output system. Firmware that activates peripheral devices in a PC. Includes routines for the keyboard, screen, disk, parallel port and serial port, and for internal services such as time and date. It accepts requests from the device drivers in the operating system as well from application programs. It also contains autostart functions that test the system on startup and prepare the computer for operation. It loads the operating system and passes control to it.

batch processing. Execution of programs serially with no interactive processing. Contrast with real time processing.

bias. A measure of how closely the mean value in a series of replicate measurements approaches the true value. See: accuracy, precision, calibration.

binary. The base two number system. Permissible digits are "0" and "1".

bit. A contraction of the term binary digit. The bit is the basic unit of digital data. It may be in one of two states, logic 1 or logic 0.

block. (1) A string of records, words, or characters that for technical or logical purposes are treated as a unity. (2) A collection of contiguous records that are recorded as a unit, and the units are separated by interblock gaps. (3) In programming languages, a subdivision of a program that serves to group related statements, delimit routines, specify storage allocation, delineate the applicability of labels, or segment parts of the program for other purposes. In

FORTRAN, a block may be a sequence of statements; in COBOL, it may be a physical record.

block diagram. A diagram of a system, instrument or computer, in which the principal parts are represented by suitably annotated geometrical figures to show both the basic functions of the parts and the functional relationships between them.

block length. (1) The number of records, words or characters in a block. (2) A measure of the size of a block, usually specified in units such as records, words, computer words, or characters.

boot. (1) To initialize a computer system by clearing memory and reloading the operating system. (2) To cause a computer system to reach a known beginning state. **bootstrap.** A short computer program that is permanently resident or easily loaded into a computer and whose execution brings a larger program, such an operating system or its loader, into memory.

buffer. A device or storage area [memory] used to store data temporarily to compensate for differences in rates of data flow, time of occurrence of events, or amounts of data that can be handled by the devices or processes involved in the transfer or use of the data.

bug. A fault in a program which causes the program to perform in an unintended or unanticipated manner. See: anomaly, defect, error, exception, fault.

bus. A common pathway along which data and control signals travel between different hardware devices within a computer system.

byte. A sequence of adjacent bits, usually eight, operated on as a unit.

CAM. computer aided manufacturing. The automation of manufacturing systems and techniques, including the use of computers to communicate work instructions to automate machinery for the handling of the processing [numerical control, process control, robotics, material requirements planning] needed to produce a workpiece.

CASE. computer aided software engineering. An automated system for the support of software development including an integrated tool set, i.e., programs, which facilitate the accomplishment of software engineering methods and tasks such as project planning and estimation, system and software requirements analysis, design of data structure, program architecture and algorithm procedure, coding, testing and maintenance.

COTS. configurable, off-the-shelf software. Application software, sometimes general purpose, written for a variety of industries or users in a manner that permits users to modify the program to meet their individual needs.

CPU. central processing unit. The unit of a computer that includes the circuits controlling the interpretation of program instructions and their execution. The CPU controls the entire computer. It receives and sends data through input-output channels, retrieves data and programs from memory, and conducts mathematical and logical functions of a program.

C. A general purpose high-level programming language. Created for use in the development of computer operating systems software. It strives to combine the power of assembly language with the ease of a high-level language.

C++. An object-oriented high-level programming language.

change tracker. A software tool which documents all changes made to a program.

client-server. A term used in a broad sense to describe the relationship between the receiver and the provider of a service. In the world of microcomputers, the term client-server describes a networked system where front-end applications, as the client, make service requests upon another networked system.

COBOL. Acronym for COmmon Business Oriented Language. A high-level programming language intended for use in the solution of problems in business data processing.

code audit. An independent review of source code by a person, team, or tool to verify compliance with software design documentation and programming standards.

code inspection. A manual [formal] testing [error detection] technique where the programmer reads source code, statement by statement, to a group who ask questions analyzing the program logic, analyzing the code with respect to a checklist of historically common programming errors, and analyzing its compliance with coding standards.

coding. (1) In software engineering, the process of expressing a computer program in a programming language. (2) The transforming of logic and data from design specifications (design descriptions) into a programming language.

comparator. A software tool that compares two computer programs, files, or sets of data to identify commonalities or differences. Typical objects of comparison are similar versions of source code, object code, data base files, or test results.

compatibility. The capability of a functional unit to meet the requirements of a specified interface.

compilation. Translating a program expressed in a problem-oriented language or a procedure oriented language into object code.

compiler. (1) A computer program that translates programs expressed in a high-level language into their machine language equivalents. (2) The compiler takes the finished source code listing as input and outputs the machine code instructions that the computer must have to execute the program.

computer. A functional unit that can perform substantial computations, including numerous arithmetic operations, or logic operations, without human intervention during a run.

computer aided design. The use of computers to design products. CAD systems are high speed workstations or personal computers using CAD software and input devices such as graphic tablets and scanners to model and simulate the use of proposed products.

computer language. A language designed to enable humans to communicate with computers. See: programming language.

computer system. A functional unit, consisting of one or more computers and associated peripheral input and output devices, and associated software. **configuration.** (1) The arrangement of a computer system or component as defined by the number, nature, and interconnections of its constituent parts. (2) In configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product.

control flow. In programming languages, an abstraction of all possible paths that an execution sequence may take through a program.

controller. Hardware that controls peripheral devices such as a disk or display screen. It performs the physical data transfers between main memory and the peripheral device.

cross-compiler. A compiler that executes on one computer but generates assembly code or object code for a different computer.

DOS. disk operating system.

data. Representations of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automated means.

data analysis. Evaluation of the data structure and usage in the code to ensure each is defined and used properly by the program. Usually performed in conjunction with logic analysis.

data bus. A bus used to communicate data internally and externally to and from a processing unit or a storage device.

data element. (1) A named unit of data that, in some contexts, is considered indivisible and in other contexts may consist of data items. (2) A named identifier of each of the entities and their attributes that are represented in a database.

data flow analysis. A software V&V task to ensure that the input and output data and their formats are properly defined, and that the data flows are correct.

data integrity. The degree to which a collection of data is complete, consistent, and accurate.

data set. A collection of related records. Syn: file.

data structure. A physical or logical relationship among data elements, designed to support specific data manipulation functions.

data validation. A process used to determine if data are inaccurate, incomplete, or unreasonable. The process may include format checks, completeness checks, check key tests, reasonableness checks and limit checks.

database. A collection of interrelated data, often with controlled redundancy, organized according to a schema to serve one or more applications. The data are stored so that they can be used by different programs without concern for the data structure or organization.

dead code. Program code statements which can never execute during program operation. Such code can result from poor coding style, or can be an artifact of previous versions or debugging efforts. Dead code can be confusing, and is a potential source of erroneous software changes.

debugging. Determining the exact nature and location of a program error, and fixing the error.

design. The process of defining the architecture, components, interfaces, and other characteristics of a system or component.

design phase. The period of time in the software life cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements.

design requirement. A requirement that specifies or constrains the design of a system or system component.

design standards. Standards that describe the characteristics of a design or a design description of data or program components.

developer. A person, or group, that designs and/or builds and/or documents and/or configures the hardware and/or software of computerized systems.

development methodology. A systematic approach to software creation that defines development phases and specifies the activities, products, verification procedures, and completion criteria for each phase.

different software system analysis. Analysis of the allocation of software requirements to separate computer systems to reduce integration and interface errors related to safety. Performed when more than one software system is being integrated.

digital. Pertaining to data [signals] in the form of discrete integral values.

disk operating system. An operating system program; e.g., DR-DOS from Digital Research, MS-DOS from Microsoft Corp., OS/2 from IBM, PC-DOS from IBM, System-7 from Apple.

driver. A program that links a peripheral device or internal function to the operating system, and providing for activation of all device functions.

dynamic analysis. Analysis that is performed by executing the program code.

editing. Modifying the content of the input by inserting, deleting, or moving characters, numbers, or data.

embedded software. Software that is part of a larger system and performs some of the requirements of that system; e.g., software used in an aircraft or rapid transit system. Such software does not provide an interface with the user. See: firmware. **encapsulation.** A software development technique that consists of isolating a system function or a set of data and the operations on those data within a module and providing precise specifications for the module.

end user. A person, device, program, or computer system that uses an information system for the purpose of data processing in information exchange.

error. A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

error detection. Techniques used to identify errors in data transfers. See: check summation, cyclic redundancy check [CRC], parity check, longitudinal redundancy.

execution trace. A record of the sequence of instructions executed during the execution of a computer program. Often takes the form of a list of code labels encountered as the program executes.

FTP. file transfer protocol.

failure. The inability of a system or component to perform its required functions within specified performance requirements.

failure analysis. Determining the exact nature and location of a program error in order to fix the error, to identify and fix other similar errors, and to initiate corrective action to prevent future occurrences of this type of error.

fault. An incorrect step, process, or data definition in a computer program which causes the program to perform in an unintended or unanticipated manner.

file. (1) A set of related records treated as a unit; e.g., in stock control, a file could consists of a set of invoices. (2) The largest unit of storage structure that consists of a named collection of all occurrences in a database of records of a particular record type.

file maintenance. The activity of keeping a file up to date by adding, changing, or deleting data.

file transfer protocol. (1) Communications protocol that can transmit binary and ASCII data files without loss of data. See: Kermit, Xmodem, Ymodem, Zmodem. (2) TCP/IP protocol that is used to log onto the network, list directories, and copy files. It can also translate between ASCII and EBCDIC.

firmware. The combination of a hardware device; e.g., an IC; and computer instructions and data that reside as read only software on that device. Such software cannot be modified by the computer during processing.

FORTRAN. An acronym for FORmula TRANslator, the first widely used high-level programming language. Intended primarily for use in solving technical problems in mathematics, engineering, and science.

functional analysis. Verifies that each safety-critical software requirement is covered and that an appropriate criticality level is assigned to each software element.

functional configuration audit. An audit conducted to verify that the development of a configuration item has been completed satisfactorily, that the item has achieved the performance and functional characteristics specified in the functional or allocated configuration identification, and that its operational and support documents are complete and satisfactory.

gigabyte. Approximately one billion bytes; precisely 230 or 1,073,741,824 bytes.

graph. A diagram or other representation consisting of a finite set of nodes and internode connections called edges or arcs.

graphic software specifications. Documents such as charts, diagrams, graphs which depict program structure, states of data, control, transaction flow, HIPO, and causeeffect relationships; and tables including truth, decision, event, state-transition, module interface, exception conditions/responses necessary to establish design integrity.

hardware. Physical equipment, as opposed to programs, procedures, rules, and associated documentation.

high-level language. A programming language which requires little knowledge of the target computer, can be translated into several different machine languages, allows symbolic naming of operations and addresses, provides features designed to facilitate expression of data structures and program logic, and usually results in several machine instructions for each program statement. Examples are PL/1, COBOL, BASIC, FORTRAN, Ada, Pascal, and "C".

I/0. input/output.

ISO. International Organization for Standardization.

implementation. The process of translating a design into hardware components, software components, or both.

implementation phase. The period of time in the software life cycle during which a software product is created from design documentation and debugged.

implementation requirement. A requirement that specifies or constrains the coding or construction of a system or system component.

incremental development. A software development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative [rather than sequential] manner, resulting in incremental completion of the overall software product.

information hiding. The practice of "hiding" the details of a function or structure, making them inaccessible to other parts of the program.

input/output. Each microprocessor and each computer needs a way to communicate with the outside world in order to get the data needed for its programs and in order to communicate the results of its data manipulations. This is accomplished through I/0 ports and devices.

installation. The phase in the system life cycle that includes assembly and testing of the hardware and software of a computerized system. Installation includes installing a new computer system, new software or hardware, or otherwise modifying the current system.

installation and checkout phase. The period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required.

instruction. (1) program statement that causes a computer to perform a particular operation or set of operations. (2) In a programming language, a meaningful expression that specifies one operation and identifies its operands, if any.

integrated circuit (IC). Small wafers of semiconductor material [silicon] etched or printed with extremely small electronic switching circuits. Syn: chip.

interface. A shared boundary between two functional units, defined by functional characteristics, common physical interconnection characteristics, signal characteristics, and other characteristics, as appropriate. The concept involves the specification of the connection of two devices having different functions.

interpret. To translate and execute each statement or construct of a computer program before translating and executing the next.

interpreter. A computer program that translates and executes each statement or construct of a computer program before translating and executing the next. The interpreter must be resident in the computer each time a program [source code file] written in an interpreted language is executed.

invalid inputs. Test data that lie outside the domain of the function the program represents.

KB. kilobyte. Approximately one thousand bytes. This symbol is used to describe the size of computer memory or disk storage space. Because computers use a binary number system, a kilobyte is precisely 210 or 1024 bytes.

key element. An individual step in an critical control point of the manufacturing process.

life cycle methodology. The use of any one of several structured methods to plan, design, implement, test. and operate a system from its conception to the termination of its use. See: waterfall model.

linkage editor. A computer program that creates a single load module from two or more independently translated object modules or load modules by resolving cross references among the modules and, possibly, by relocating elements. Syn: link editor

loader. A program which copies other [object] programs from auxiliary [external] memory to main [internal] memory prior to its execution.

low-level language. The advantage of assembly language is that it provides bit-level control of the processor allowing tuning of the program for optimal speed and performance. For time critical operations, assembly language may be necessary in order to generate code which executes fast enough for the required operations. The disadvantage of assembly language is the high-level of complexity and detail required in the programming. This makes the source code harder to understand, thus increasing the chance of introducing errors during program development and maintenance.

Mb. megabit. Approximately one million bits. Precisely 1024 K bits, 220 bits, or 1,048,576 bits.

MB. megabyte. Approximately one million bytes. Precisely 1024 K Bytes, 220 bytes, or 1,048,576 bytes. See: kilobyte.

MIPS. million instructions per second.

machine code. Computer instructions and definitions expressed in a form [binary code] that can be recognized by the CPU of a computer. All source code, regardless of the language in which it was programmed, is eventually converted to machine code. Syn: object code.

macroinstruction. A source code instruction that is replaced by a predefined sequence of source instructions, usually in the same language as the rest of the program and usually during assembly or compilation.

main memory. A non-moving storage device utilizing one of a number of types of electronic circuitry to store information.

main program. A software component that is called by the operating system of a computer and that usually calls other software components. See: routine, subprogram.

maintenance. Activities such as adjusting, cleaning, modifying, overhauling equipment to assure performance in accordance with requirements. Maintenance to a software system includes correcting software errors, adapting software to a new environment, or making enhancements to software.

measure. A quantitative assessment of the degree to which a software product or process possesses a given attribute.

megahertz. A unit of frequency equal to one million cycles per second.

memory. Any device or recording medium into which binary data can be stored and held, and from which the entire original data can be retrieved.

menu. A computer display listing a number of options; e.g., functions, from which the operator may select one. Sometimes used to denote a list of programs.

metric, software quality. A quantitative measure of the degree to which software possesses a given attribute which affects its quality.

microcode. Permanent memory that holds the elementary circuit operations a computer must perform for each instruction in its instruction set.

microprocessor. A CPU existing on a single IC. Frequently synonymous with a microcomputer.

modeling. Construction of programs used to model the effects of a postulated environment for investigating the dimensions of a problem for the effects of algorithmic processes on responsive targets.

module. (1) In programming languages, a self- contained subdivision of a program that may be separately compiled. (2) A discrete set of instructions, usually processed as a unit, by an assembler, a compiler, a linkage editor, or similar routine or subroutine. (3) A packaged functional hardware unit suitable for use with other components. See: unit.

multi-processing. A mode of operation in which two or more processes [programs] are executed concurrently [simultaneously] by separate CPUs that have access to a common main memory. Contrast with multi-programming.

multi-programming. A mode of operation in which two or more programs are executed in an interleaved manner by a single CPU.

multi-tasking. A mode of operation in which two or more tasks are executed in an interleaved manner.

network. (1) An arrangement of nodes and interconnecting branches. (2) A system [transmission channels and supporting hardware and software] that connects several remotely located computers via telecommunications.

OOP. object oriented programming. A technology for writing programs that are made up of self-sufficient modules that contain all of the information needed to manipulate a given data structure. The modules are created in class hierarchies so that the code or methods of a class can be passed to other modules. New object modules can be easily created by inheriting the characteristics of existing classes.

object. In object oriented programming, A self contained module [encapsulation] of data and the programs [services] that manipulate [process] that data.

object code. A code expressed in machine language ["1"s and "0"s] which is normally an output of a given translation process that is ready to be executed by a computer.

object oriented design. A software development technique in which a system or component is expressed in terms of objects and connections between those objects.

object oriented language. A programming language that allows the user to express a program in terms of objects and messages between those objects. Examples include C++, Smalltalk and LOGO.

object program. A computer program that is the output of an assembler or compiler.

operating system. Software that controls the execution of programs, and that provides services such as resource allocation, scheduling, input/output control, and data management. Usually, operating systems are predominantly software, but partial or complete hardware implementations are possible.

Oracle. A relational database programming system incorporating the SQL programming language. A registered trademark of the Oracle Corp.

original equipment manufacturer. A manufacturer of computer hardware.

PROM. programmable read only memory.

parallel. (1) Pertaining to the simultaneity of two or more processes. (2) Pertaining to the simultaneous processing of individual parts of a whole, such as the bits of a character or the characters of a word, using separate facilities for the various parts. (3) Term describing simultaneous transmission of the bits making up a character, usually eight bits [one byte].

parallel processing. See: multi-processing, multi- programming.

parameter. A constant, variable or expression that is used to pass values between software modules.

parity. An error detection method in data transmissions that consists of selectively adding a 1-bit to bit patterns [word, byte, character, message] to cause the bit patterns to have either an odd number of 1-bits [odd parity] or an even number of 1-bits [even parity].

Pascal. A high-level programming language designed to encourage structured programming practices.

path. A sequence of instructions that may be performed in the execution of a computer program.

peripheral device. Equipment that is directly connected a computer. A peripheral device can be used to input data; e.g., keypad, bar code reader, transducer, laboratory test equipment; or to output data; e.g., printer, disk drive, video system, tape drive, valve controller, motor controller. Syn: peripheral equipment.

pixel. (1) In image processing and pattern recognition, the smallest element of a digital image that can be assigned a gray level. (2) In computer graphics, the smallest element of a display surface that can be assigned independent characteristics. This term is derived from the term "picture element".

platform. The hardware and software which must be present and functioning for an application program to run [perform] as intended. A platform includes, but is not limited to the operating system or executive software, communication software, microprocessor, network, input/output hardware, any generic software libraries, database management, user interface software, and the like.

program. A sequence of instructions suitable for processing. Processing may include the use of an assembler, a compiler, an interpreter, or another translator to prepare the program for execution. The instructions may include statements and necessary declarations.

program design language. A specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a program design.

programmable logic device. A logic chip that is programmed at the user's site.

programmable read only memory. A chip which may be programmed by using a PROM programming device. It can be programmed only once. It cannot be erased and reprogrammed. Each of its bit locations is a fusible link.

programming language. A language used to express computer programs.

PROM programmer. Electronic equipment which is used to transfer a program [write instructions and data] into PROM and EPROM chips.

protocol. A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication.

prototyping. Using software tools to accelerate the software development process by facilitating the identification of required functionality during analysis and design phases. A limitation of this technique is the identification of system or software problems and hazards.

pseudocode. A combination of programming language and natural language used to express a software design. If used, it is usually the last document produced prior to writing the source code.

qualification, operational. Establishing confidence that process equipment and subsystems are capable of consistently operating within established limits and tolerances.

qualification, process performance. Establishing confidence that the process is effective and reproducible.

RAM. random access memory. Chips which can be called read/write memory, since the data stored in them may be read or new data may be written into any memory address on these chips. The term random access means that each memory location [usually 8 bits or 1 byte] may be directly accessed [read from or written to] at random.

ROM. read only memory. A memory chip from which data can only be read by the CPU. The CPU may not store data to this memory. The advantage of ROM over RAM is that ROM does not require power to retain its program. This advantage applies to all types of ROM chips; ROM, PROM, EPROM, and EEPROM.

real time. Pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process. Contrast with batch. See: conversational, interactive, interrupt, on-line.

real time processing. A fast-response [immediate response] on-line system which obtains data from an activity or a physical process, performs computations, and returns a response rapidly enough to affect [control] the outcome of the activity or process; e.g., a process control application. Contrast with batch processing.

record. (1) a group of related data elements treated as a unit. [A data element (field) is a component of a record, a record is a component of a file (database)].

relational database. Database organization method that links files together as required. Relationships between files are created by comparing data such as account numbers and names. A relational system can take any two or more files and generate a new file from the records that meet the matching criteria.

reliability. The ability of a system or component to perform its required functions under stated conditions for a specified period of time.

requirement. (1) A condition or capability needed by a user to solve a problem or achieve an objective. (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. (3) A documented representation of a condition or capability as in (1) or (2).

requirements phase. The period of time in the software life cycle during which the requirements, such as functional and performance capabilities for a software product, are defined and documented.

robustness. The degree to which a software system or component can function correctly in the presence of invalid inputs or stressful environmental conditions. See: software reliability.

routine. A subprogram that is called by other programs and subprograms. Note: This term is defined differently in various programming languages.

SOPs. standard operating procedures.

SQL. structured query language.

safety. Freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property, or damage to the environment.

server. A high speed computer in a network that is shared by multiple users. It holds the programs and data that are shared by all users.

simulation. (1) Use of an executable model to represent the behavior of an object. During testing the computational hardware, the external environment, and even code segments may be simulated. (2) A model that behaves or operates like a given system when provided a set of controlled inputs. Contrast with emulation.

software. Programs, procedures, rules, and any associated documentation pertaining to the operation of a system.

software characteristic. An inherent, possibly accidental, trait, quality, or property of software; e.g., functionality, performance, attributes, design constraints, number of states, lines or branches.

software design description. A representation of software created to facilitate analysis, planning, implementation, and decision making. The software design description is used as a medium for communicating software design information, and may be thought of as a blueprint or model of the system.

software development process. The process by which user needs are translated into a software product. the process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes installing and checking out the software for operational activities.

software documentation. Technical data or information, including computer listings and printouts, in human readable form, that describe or specify the design or details, explain the capabilities, or provide operating instructions.

software element. A deliverable or in- process document produced or acquired during software development or maintenance.

software engineering. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; i.e., the application of engineering to software.

software item. Source code, object code, job control code, control data, or a collection of these items. Contrast with software element.

software life cycle. Period of time beginning when a software product is conceived and ending when the product is no longer available for use. The software life cycle is typically broken into phases denoting activities such as requirements, design, programming, testing, installation, and operation and maintenance.

software reliability. (1) the probability that software will not cause the failure of a system for a specified time under specified conditions. The probability is a function of the inputs to and use of the system in the software. The inputs to the system determine whether existing faults, if any, are encountered. (2) The ability of a program to perform its required functions accurately and reproducibly under stated conditions for a specified period of time.

source code. (1) Computer instructions and data definitions expressed in a form suitable for input to an assembler, compiler or other translator. (2) The human readable version of the list of instructions [program] that cause a computer to perform a task.

source program. A computer program that must be compiled, assembled, or otherwise translated in order to be executed by a computer. See: source code.

specification. A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior,or other characteristics of a system or component, and often, the procedures for determining whether these provisions have been satisfied.

specification, product. A document which describes the as built version of the software.

specification, requirements. A specification that documents the requirements of a system or system component. It typically includes functional requirements, performance requirements, interface requirements, design requirements [attributes and constraints], development [coding] standards, etc.

spiral model. A model of the software development process in which the constituent activities, typically requirements analysis, preliminary and detailed design, coding, integration, and testing, are performed iteratively until the software is complete.

standard operating procedures. Written procedures [prescribing and describing the steps to be taken in normal and defined conditions] which are necessary to assure control of production and processes.

storage device. A unit into which data or programs can be placed, retained and retrieved.

structured programming. Any software development technique that includes structured design and results in the development of structured programs.

structured query language. A language used to interrogate and process data in a relational database. Originally developed for IBM mainframes, there have been many implementations created for mini and micro computer database applications. SQL commands can be used to interactively work with a data base or can be embedded with a programming language to interface with a database.

subprogram. A separately compilable, executable component of a computer program. Note: This term is defined differently in various programming languages.

subroutine. A routine that returns control to the program or subprogram that called it. Note: This term is defined differently in various programming languages.

support software. Software that aids in the development and maintenance of other software; e.g., compilers, loaders, and other utilities.

syntax. The structural or grammatical rules that define how symbols in a language are to be combined to form words, phrases, expressions, and other allowable constructs.

system analysis. A systematic investigation of a real or planned system to determine the functions of the system and how they relate to each other and to any other system.

system design. A process of defining the hardware and software architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.

system life cycle. The course of developmental changes through which a system passes from its conception to the termination of its use; e.g., the phases and activities associated with the analysis, acquisition, design, development, test, integration, operation, maintenance, and modification of a system. See: software life cycle.

system software. (1) Application- independent software that supports the running of application software. (2) Software designed to facilitate the operation and maintenance of a computer system and its associated programs; e.g., operating systems, assemblers, utilities.

terminal. A device, usually equipped with a CRT display and keyboard, used to send and receive information to and from a computer via a communication channel.

test. An activity in which a system or component is executed under specified conditions, the results are observed or recorded and an evaluation is made of some aspect of the system or component.

test phase. The period of time in the software life cycle in which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied.

test procedure. A formal document developed from a test plan that presents detailed instructions for the setup, operation, and evaluation of the results for each defined test.

testing. (1) The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component. (2) The process of analyzing a software item to detect the differences between existing and required conditions, i.e. bugs, and to evaluate the features of the software items.

testing, compatibility. The process of determining the ability of two or more systems to exchange information. In a situation where the developed software replaces an already working program, an investigation should be conducted to assess possible comparability problems between the new software and other programs or systems

testing, unit. (1) Testing of a module for typographic, syntactic, and logical errors, for correct implementation of its design, and for satisfaction of its requirements. (2) Testing

conducted to verify the implementation of the design for one software element; e.g., a unit or module; or a collection of software elements.

time sharing. A mode of operation that permits two or more users to execute computer programs concurrently on the same computer system by interleaving the execution of their programs. May be implemented by time slicing, priority-based interrupts, or other scheduling methods.

top-down design. Pertaining to design methodology that starts with the highest level of abstraction and proceeds through progressively lower levels.

transaction. (1) A command, message, or input record that explicitly or implicitly calls for a processing action, such as updating a file. (2) An exchange between and end user and an interactive system. (3) In a database management system, a unit of processing activity that accomplishes a specific purpose such as a retrieval, an update, a modification, or a deletion of one or more data elements of a storage structure.

unambiguous. (1) Not having two or more possible meanings. (2) Not susceptible to different interpretations. (3) Not obscure, not vague. (4) Clear, definite, certain.

unit. (1) A separately testable element specified in the design of a computer software element. (2) A logically separable part of a computer program. Syn: module.

UNIX. A multitasking, multiple-user (time-sharing) operating system developed at Bell Labs to create a favorable environment for programming research and development.

utility program. A computer program in general support of the processes of a computer; e.g., a diagnostic program, a trace program, a sort program.

utility software. Computer programs or routines designed to perform some general support function required by other application software, by the operating system, or by the system users. They perform general functions such as formatting electronic media, making copies of files, or deleting files.

VV&T. validation, verification, and testing.

valid input. Test data that lie within the domain of the function represented by the program.

validation. (1) Establishing documented evidence which provides a high degree of assurance that a specific process will consistently produce a product meeting its predetermined specifications and quality attributes.

validation, software. Determination of the correctness of the final program or software produced from a development project with respect to the user needs and requirements. Validation is usually accomplished by verifying each stage of the software development life cycle.

verification, software. In general the demonstration of consistency, completeness, and correctness of the software at each stage and between each stage of the development life cycle. See: validation, software.

virus. A program which secretly alters other programs to include a copy of itself, and executes when the host program is executed. The execution of a virus program compromises a computer system by performing unwanted or unintended functions which may be destructive.

WAN. wide area network.

watchdog timer. A form of interval timer that is used to detect a possible malfunction.