СТВОРЕННЯ ТА АНАЛІЗ РІЗНОВИДІВ СТАТИЧНИХ ДІАГРАМ ЗАЛЕЖНОСТЕЙ КЛАСІВ В ПРОГРАМНИХ ПРОЕКТАХ

ПІДГОТУВАЛА: СТУДЕНТКА КН БП-4 КАТЕРИНА БОНДАР

НАУКОВИЙ КЕРІВНИК: БУБЛИК В.В.

ОСНОВНА ІДЕЯ

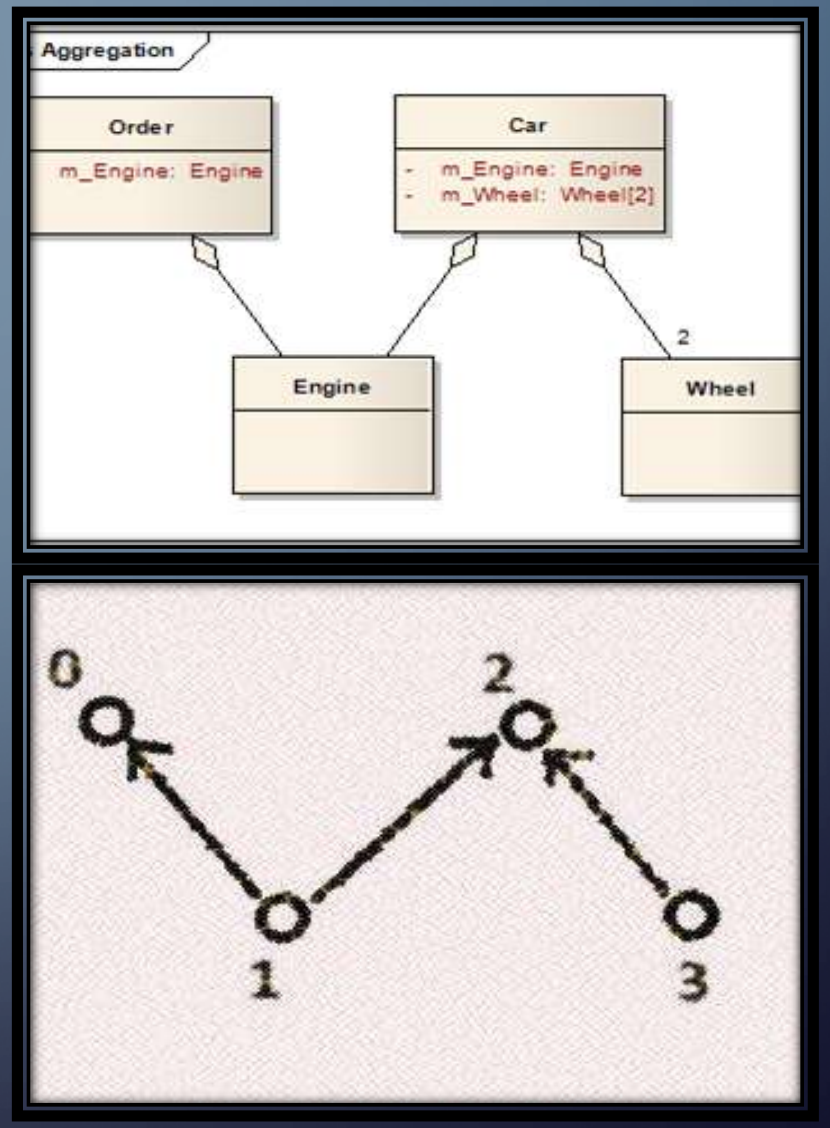
- Пошук циклів залежностей в проєкті за час компіляції;
- Створення реалізацій статичних діаграм залежностей класів;
- Реалізація алгоритмів на цих діаграмах;
- Конвертація ієрархії проєкту в діаграму.

ДІАГРАМИ ЗАЛЕЖНОСТЕЙ КЛАСІВ: ОРГРАФ ЗАЛЕЖНОСТЕЙ

Завдання аналізу статичної конструкції залежностей класів — **оргграф:**

- Багато вже існуючих алгоритмів пошуку циклів і шляхів.
- Оргграф це достатньо повна спрощена модель, зручна для аналізу.

3 стандартних представлень G(V, E), G(V, A) G(Aij) – обрано G(V, A).



ДІАГРАМИ ЗАЛЕЖНОСТЕЙ КЛАСІВ: РЕАЛІЗАЦІЯ

Загальна будова:

- двовимірний std::array розміром V на V+1;

Функції:

- neighbourhoods(v) – список сусідів вершини v;
- vertex_count() – кількість вершин в орграфі;
- has_oriented_edge(v_out, v_in) – показує чи наявне в орграфі ребро з вершини v_out до вершини v_in.

ДІАГРАМИ ЗАЛЕЖНОСТЕЙ КЛАСІВ: РЕАЛІЗАЦІЇ

Реалізацій 3:

- динамічна;
- статична, базована на constexpr засобах;
- статична, базована на на засобах метапрограмуваннння мовою шаблонів (створено прототип).

АЛГОРИТМИ

Адаптовано під реалізації три основні алгоритми:

- DFS – з функтором обробки пошукової ітерації;
- BFS – також з функтором обробки пошукової ітерації;
- Алгоритм пошуку циклів Шварцфітера та Лауера (базований на алгоритмі пошуку компонент сильної зв’язності Тарджана) – статично.

АЛГОРИТМ ПОШУКУ ЦИКЛІВ

Найефективніші алгоритми пошуку циклів в орграфах: алгоритм Джонсона і алгоритм Шварцфітера та Лауера.

- Базуються на алгоритмі пошуку компонент сильної зв’язності Тарджана – O(V+E).
- Алгоритм Джонсона є більш поширеним, O((V+E)(C+1)).
- Алгоритм Шварцфітера та Лауера є кращим за алгоритм Джонсона щодо детекції циклів, O(V+E(C+1)).

КОНВЕРТАЦІЯ ІЄРАРХІЇ

Відношення агрегації задаються вручну таким чином:

```
using Converter = HierarchyToGraphConverter<
        AdjacencyEntry<Aggregated, List<>>,
        AdjacencyEntry<Aggregator, List<Aggregated>>
>;
graph = Converter::graph_of_hierarchy();
```

Після конвертації утворений орграф можна використовувати незалежно.

Конвертор можна використати для повернення назви класу за індексом вершини.

ПРОРГАМА, ЩО ВИВОДИТЬ ЦИКЛИ ЗАЛЕЖНОСТЕЙ

Запускає перетворення в статичну чи динамічну діаграму;

Проводить пошук циклів залежностей;

Виводить цикли чи повідомлення про їх відсутність

МОЖЛИВІ РОЗШИРЕННЯ

- Підрахунок залежності класів від інших класів та подібні підрахунки на графі;
- Генерування візуалізації діаграми.

ВИСНОВКИ

- Реалізовано засоби для створення статичних діаграм залежностей в програмних проєктах;
- Реалізовано засоби для створення динамічних діаграм залежностей в програмних проєктах;
- Задачу аналізу залежностей зведено до задачі пошуку циклів в орієнтованих графах;
- Виконані програмні реалізації для створення, аналізу і порівняння обох видів діаграм залежностей класів в програмних проєктів.

ДЯКУЮ ЗА УВАГУ!