Лабораторна робота №23.
ООП Поліморфізм.

- **Вимоги**
- **Розробник**
  - Гладков Костянтинт Сергіович
  - Студент групи КІТ-320;
  - 03-march-2021.
- **Основне завдання**

Загальне завдання

Модернізувати попередню лабораторну роботу шляхом:
- базовий клас зробити абстрактним. Додати абстрактні методи;
- розроблені класи-списки поєднуються до одного класу таким чином, щоб він міг працювати як з базовим класом, так і з його спадкоємцями. При цьому серед полів класу-списку повинен бути лише один масив, що містить усі типи класів ієрархії. Оновити методи, що працюють з цим масивом.
- у функціях базового класу та класів-спадкоємців обов'язкове використання ключових слів final та override.

Додаткові умови виконання завдання:

- продемонструвати відсутність витоків пам'яті;
- продемонструвати роботу розроблених методів за допомогою модульних тестів;
- не використовувати конструкцію "using namespace std;", замість цього слід робити "using" кожного необхідного класу: using std::string, using std::cout;
- у проекті не повинні використовуватися бібліотеки введення / виведення мови С, а також не повинні використовуватися рядки типу `char*`.

- **Опис роботи**

Програма працює за принципом успадкування та поліморфізму об'єкта. Таким чином маємо великий функціонал з маленькою навантаженням.

- **Функціональне призначення**

Програма може бути використана для створення сховища з об'єктом, який потрібен вам, а також його подальшим користуванням.

- **Опис логічної структури**

Данна програма працює за принципом створення класу, в якому зберігається динамічний масив іншого класу, а також за допомогою методів, які контролюється цей масив.

Функція main містить в собі виклик функцій, а також корисних властивостей для функцій

class List - клас, який зберігає в собі динамічний масив іншого класу, а також які всередині себе має всі методи роботи з даним класом.

class Agency - базовий клас.

void List::add_ag - метод, який використовується для розширення нашого масиву, а точніше додавання в даній оновлений масив нового елемента за рахунок спеціального коструктора

void kharkov_agencies() - Функція знаходить все агенції в харкові

int most_wins_law() - метод знаходить агенство з Найбільший числом перемог

void defence_in_court() - метод знаходить все агенції із захистом в суді

void no_weekends_mar_agencies() - метод знаходжу агенства, які працюють без вихідних

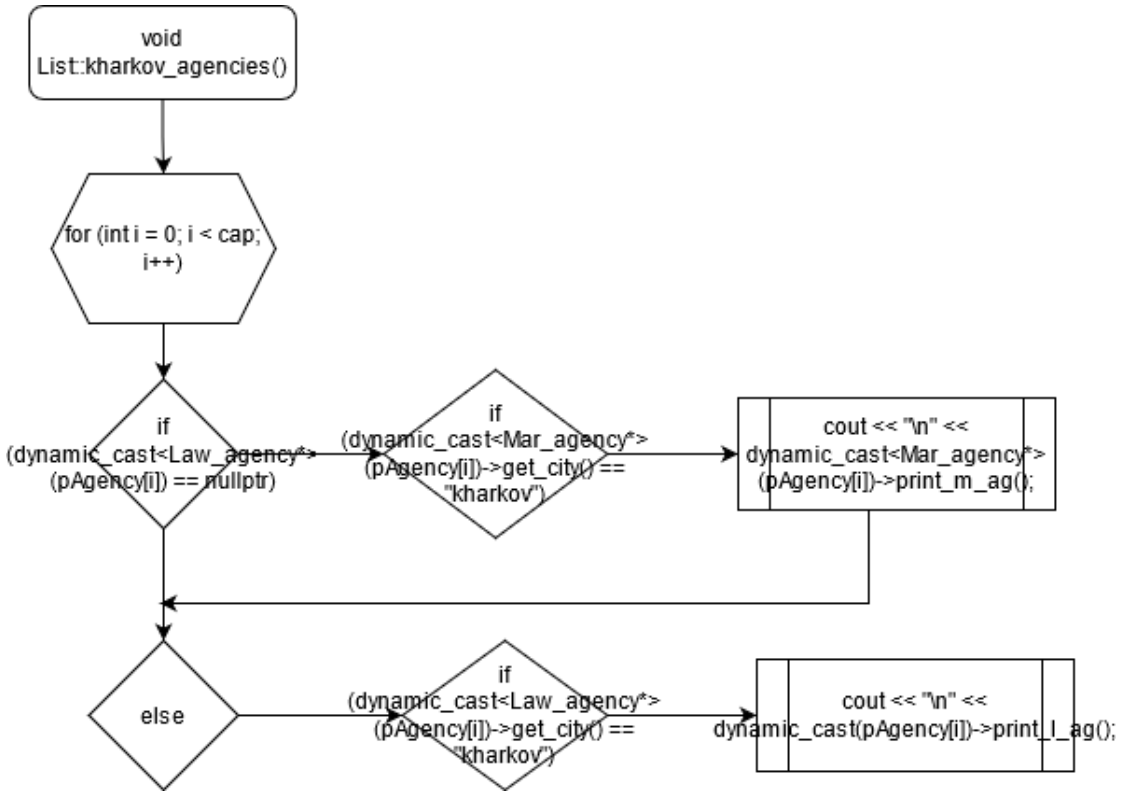

рис. 1 - kharkov_agencies

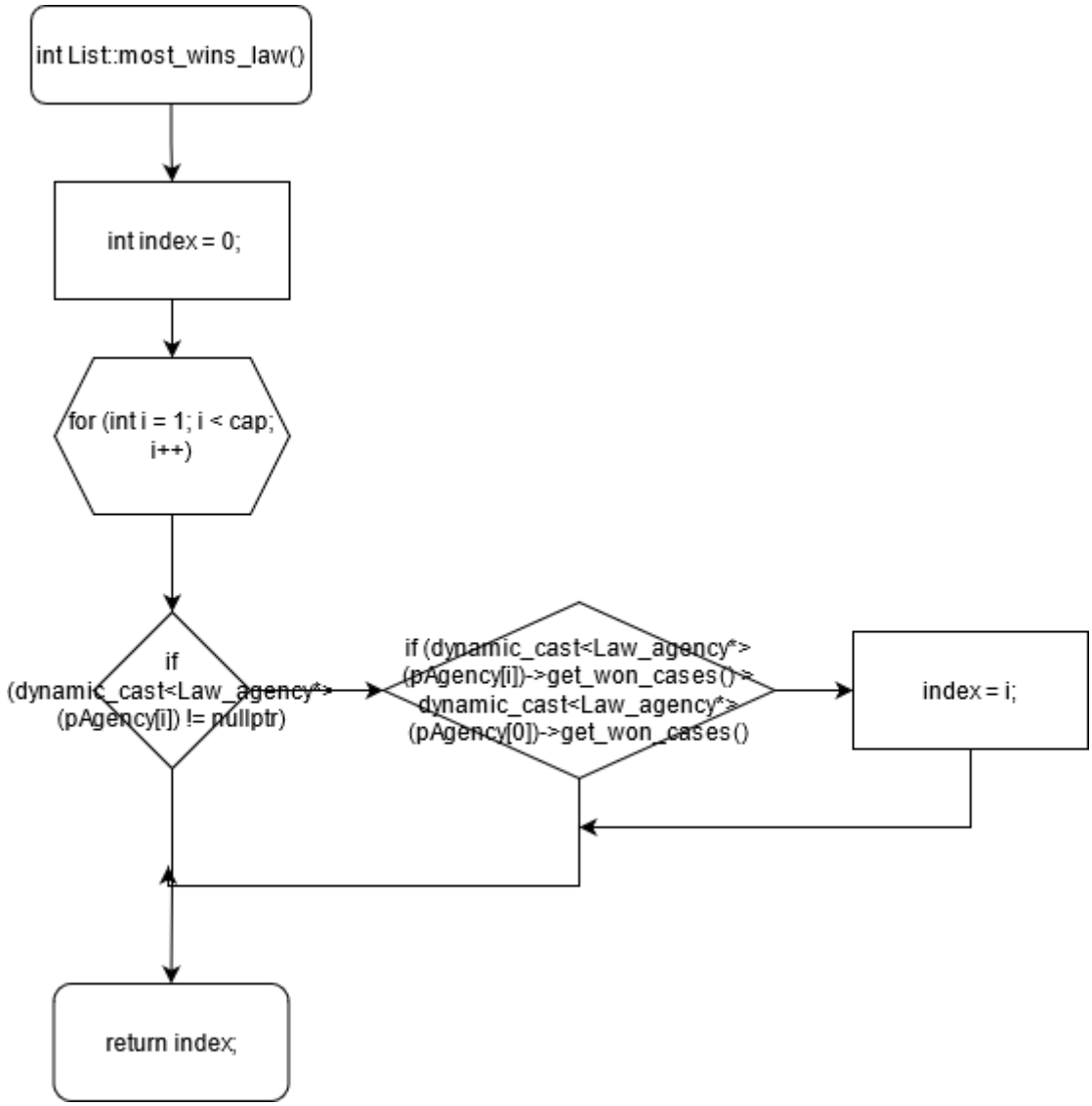

рис. 2 - most_wins_law

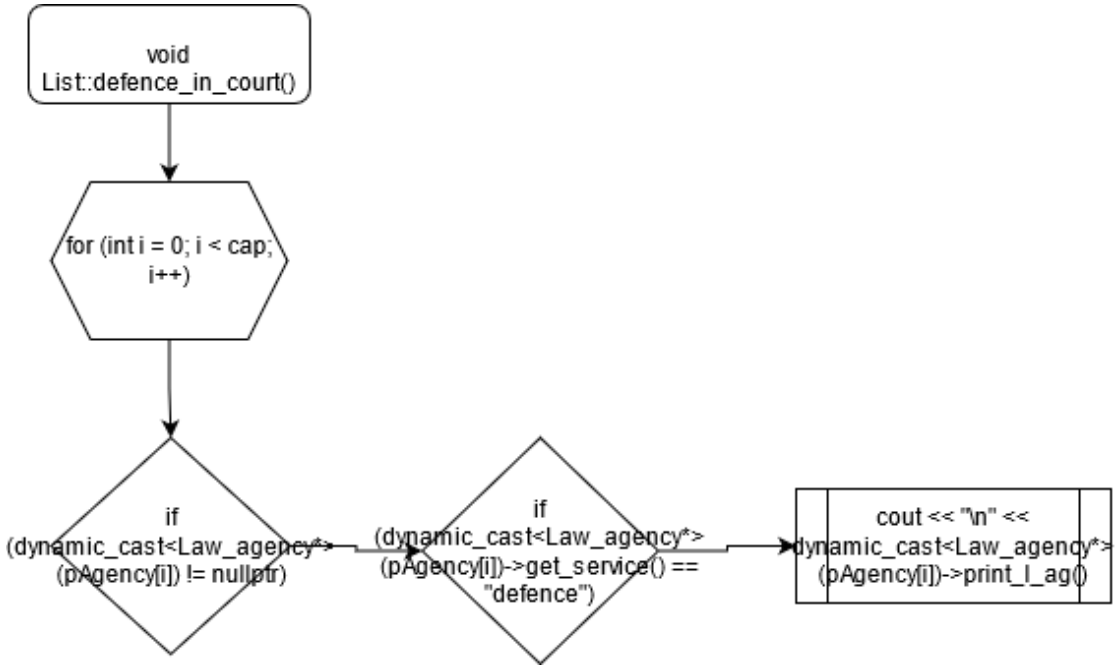

рис. 3 - defence_in_court()

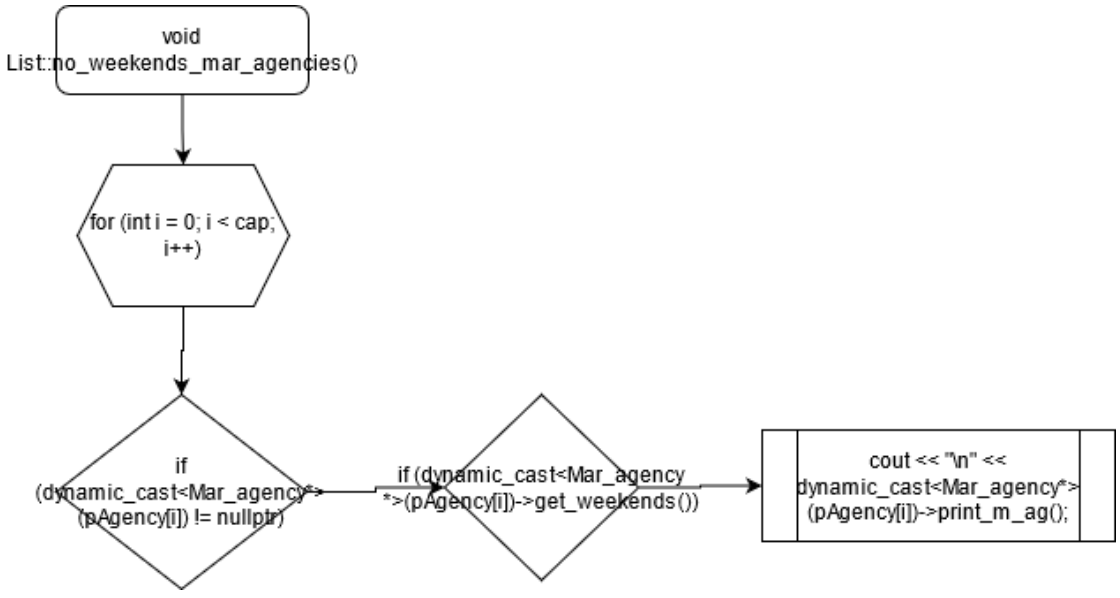

рис. 4 - no_weekends_mar_agencies()

- **Важливі елементи програми**

```
void List::add(Agency *ag){
    this->pAgency[index++] = ag;
}
```

```
Agency& List::get_ag(int index) const {

    if (index < 0 || index >= this->num_of_elem){
        cout << "Out of range!";
        exit(1);
    }

    return *this->pAgency[index];
}
```

```
int List::get_cap() {
    return cap;
}
```

- **Варіанти використання**

  Дана програма може бути використана для зберігання класів і роботи з ними

Висновки

У даній лабораторній роботі були придбані знання роботи з ООП поліморфізмом.