

Лабораторна робота №5. Агрегація і композиція

Тема: Класи. Агрегація. Композиція. Ключові слова `typedef` та `auto`

Мета: Порівняти поняття агрегація та композиція. Отримати знання про призначення ключових слів `typedef` та `auto`.

ВИМОГИ

1.1 Інформація про розробника:

- Кліщов Б. Р.
- КІТ 102.8а

1.2 Загальне завдання

Дослідити заздалегідь визначені типи даних з бібліотеки `<cstdlib>` / `<stddef.h>`. Модернізувати розроблені у попередній роботі класи наступним чином:

- замінити типи даних, що використовуються при індексуванні на типи з вказаної бібліотеки;
- створити власний синонім типу, визначивши його необхідність;
- створити/оновити функцію сортування масиву, де крім поля, по якому виконується сортування, передається і вказівник на функцію, яка визначає напрям сортування;
- в базовий клас додати два поля, що мають кастомний тип даних (тип даних користувача) та які будуть відображати відношення «агрегація» та «композиція», при цьому оновити методи читання та запису об'єкта;
- ввести використання ключового слова `auto` як специфікатор зберігання типу змінної. Визначити плюси та мінуси цього використання.

1.3 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 роботи `using` кожного необхідного класу: `using std::string;` `using std::cout;`
- в проекті не повинні використовуватися бібліотеки введення / виведення мови C, а також не повинні використовуватися рядки типу `char*`.

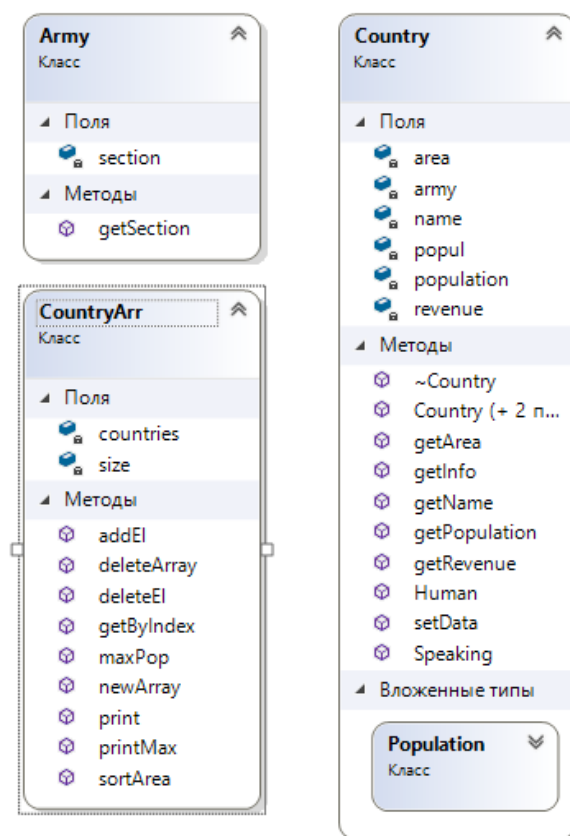
2. ОПИС ПРОГРАМИ

2.1 Функціональне призначення

Програма призначена щоб отримувати та зберігати інформацію щодо різних країн світу, сортувати масив цих країн та отримувати максимальне значення населення. Інформацію можна зчитувати з файлу та записувати в нього.

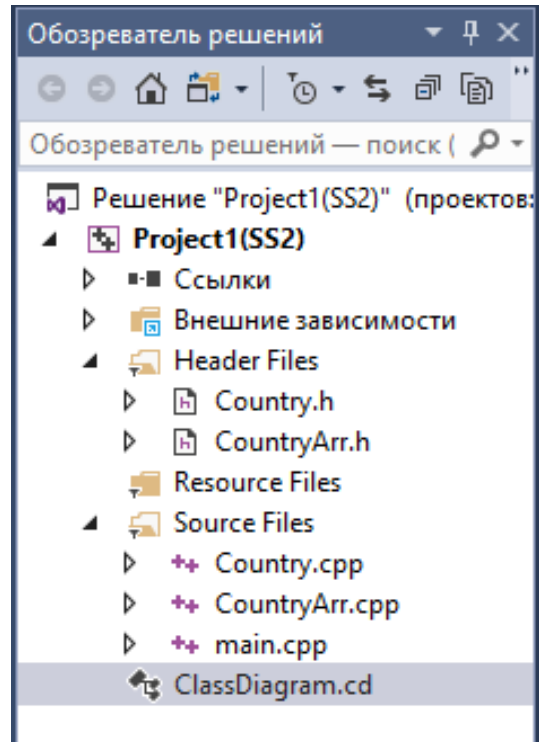
2.2 Опис логічної структури

На рисунку № 1 зображена діаграма класу



Малюнок №1. Діаграма класу

На рисунку № 2 зображена структура програми:



Малюнок №2: Структура програми

2.3 Важливі фрагменти програми

Код програми:

```
int main() {  
  
    CountryArr Countr;  
    Country Kompos;  
    Kompos.Human();  
    Country Aggreg;  
    Aggreg.Speaking();  
  
    std::regex regex_spaces("[\\s]{2,}");  
    std::regex regex_firstSymbol("^[A-Z]");  
  
    std::string name;  
    std::cout << "Please, the name of country: ";  
    getline(std::cin, name);  
    if (!(regex_search(name, regex_firstSymbol)) || regex_search(name, regex_spaces))  
    {  
        std::cout << "Incorrect entry, writing with large letters(A - Z) and  
without double spaces : " << std::endl;  
        std::cout << "Please, the name of country: ";  
        getline(std::cin, name);  
    }  
}
```

```

    }

    Countr.newArray(name);
    Countr.print();

    bool(*p)(int a, int b);

    int option = 0;
    do {
        std::cout << "Choose option:" << std::endl << "0 - Exit " << std::endl <<
        "1 - Add element" << std::endl << "2 - Delete element" << std::endl << "3 - Get by index"
        << std::endl << "4 - Search by max population" << std::endl << "5 - Sort by area" <<
        std::endl;

        std::cout << std::endl;
        std::cin >> option;

        switch (option) {
            case 1: {
                int population, area, revenue;
                std::string name;
                std::ifstream fin("data.txt");
                fin >> population >> area >> revenue >> name;
                Countr.addEl(population, area, revenue, name);
                fin >> population >> area >> revenue >> name;
                Countr.addEl(population, area, revenue, name);
                fin >> population >> area >> revenue >> name;
                Countr.addEl(population, area, revenue, name);
                system("cls");
                Countr.print();
                break;
            }
            case 2: {
                auto id = 0;
                std::cout << std::endl << "Enter index: ";
                std::cin >> id;
                std::cout << std::endl;
                Countr.deleteEl(id);
                system("cls");
                Countr.print();
                break;
            }
            case 3: {
                auto index = 0;
                std::cout << std::endl << "Enter index : ";
                std::cin >> index;
                std::cout << std::endl;
                system("cls");
                Countr.getByIndex(index);
                break;
            }
            case 4: {
                Country Max = Countr.maxPop();
                Countr.printMax(Max);
                break;
            }
            case 5: {
                int ch;
                std::cout << "Please, enter the type of sort" << std::endl << "1 -
up, 0 - down: ";

                std::cin >> ch;
                std::cout << std::endl;
                if (ch == 1) {
                    p = comp;
                }
                else if (ch == 0) {

```

```

        p = comp2;
    }
    else {
        std::cout << "You enter false variant" << std::endl;
        break;
    }
    Countr.sortArea(p);
    break;
}
default: {
    break;
}
}
}
} while (option != 0);

Countr.deleteArray();

_CrtSetReportMode(_CRT_WARN, _CRTDBG_MODE_FILE);
_CrtSetReportFile(_CRT_WARN, _CRTDBG_FILE_STDERR);
_CrtSetReportMode(_CRT_ERROR, _CRTDBG_MODE_FILE);
_CrtSetReportFile(_CRT_ERROR, _CRTDBG_FILE_STDERR);
_CrtSetReportMode(_CRT_ASSERT, _CRTDBG_MODE_FILE);
_CrtSetReportFile(_CRT_ASSERT, _CRTDBG_FILE_STDERR);

_CrtDumpMemoryLeaks();
return 0;
}

```

Регулярні вирази:

```
std::regex regex_spaces("[\\s]{2,}");  
  
std::regex regex_firstSymbol("[A-Z]");
```

Композиція та агрегація:

```
class Army {
public:
    std::string getSection() {
        return section;
    }
private:
    std::string section = "air
force";
};

class Country {
private:
    int population;
    int area;
    int revenue;
    std::string name;
    //-----
    //-----
Композиция-----
-----

    class Population {
    public:
        void Human() {
```

```

        std::cout << "I
live in a beautiful town :)" <<
std::endl;
    }
};
Population popul;
//-----
//-----
Агрегация-----
---
Army army;
//-----
public:
Country();
Country(int a, int b, int c,
std::string Name);
Country(const Country &obj);
int getPopulation();
int getArea();
int getRevenue();
std::string getName();
void getInfo(std::string s);
void setData(int population,
int area, int revenue, std::string
name);

void Human();
void Speaking();

~Country() {
}
};

```

Конструктори:

1. Без параметрів:

```

Country::Country() :population(0), area(0), revenue(0), name() {
    name = new char[24];
};

```

2. З параметрами:

```

Country::Country(int a, int b, int c, char* Name) :population(a), area(b), revenue(c) {
    name = new char[24];
    strcpy_s(name, 24, Name);
};

```





3. Копіювальний:

```

Country::Country(const Country &obj) :population(obj.population), area(obj.area),
revenue(obj.revenue), name(obj.name) {};

```

Файли:

 data.txt	26.03.2019 22:38	Текстовый докум...	1 КБ
 result.txt	26.03.2019 23:33	Текстовый докум...	1 КБ
 resultID.txt	26.03.2019 23:33	Текстовый докум...	1 КБ
 resultMax.txt	26.03.2019 23:33	Текстовый докум...	1 КБ

data.txt – для читання інформації про країни

result.txt – сюди записується результат роботи функції `void CountryArr::print()`

resultID.txt – для запису роботи функції `void getByIndex(int index);`

resultMax.txt – для запису результату `Country maxPop();`

Перевірка вхідних даних за допомогою регулярних виразів:

```
std::string name;
std::cout << "Please, the name of country: ";
getline(std::cin, name);
if (!(regex_search(name, regex_firstSymbol)) || regex_search(name, regex_spaces)) {
    std::cout << "Incorrect entry, writing with large letters(A - Z) and without
double spaces : " << std::endl;
    std::cout << "Please, the name of country: ";
    getline(std::cin, name);
}
```

Сортування:

```
void CountryArr::sortArea(bool(*comp)(int x, int y)) {
    Country temp;
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            if (comp(countries[i].getArea(), countries[j].getArea())) {
                temp = countries[i];
                countries[i] = countries[j];
                countries[j] = temp;
            }
        }
    }
}
```

3 ВАРІАНТИ ВИКОРИСТАННЯ

3.1 Результат роботи функцій

На рисунку № 3 зображено результат запису даних в файл



Рисунок № 3. Запис даних в файл

На рисунку № 4 зображено результат запису в файл країни, отриманої по ID

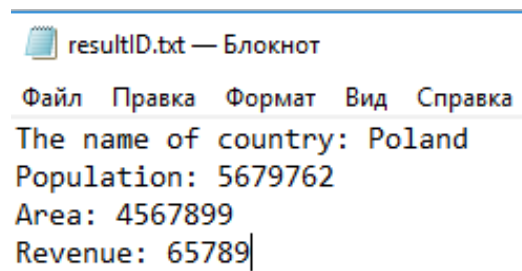
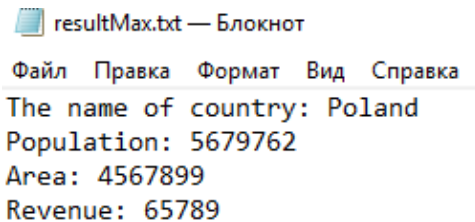


Рисунок № 4. Пошук по ID

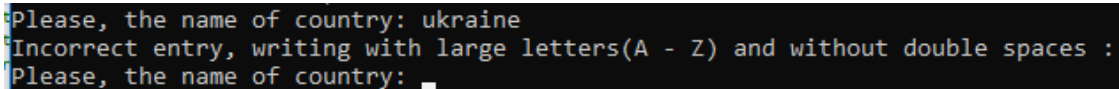
На рисунку № 5 зображено результат запису в файл країни, з максимальною кількістю населення



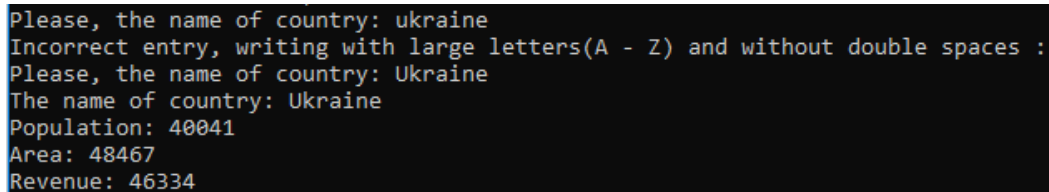
```
resultMax.txt — Блокнот
Файл  Правка  Формат  Вид  Справка
The name of country: Poland
Population: 5679762
Area: 4567899
Revenue: 65789
```

Рисунок № 5. Мінімальна щільність населення

На рисунку № зображено результат роботи регулярних виразів



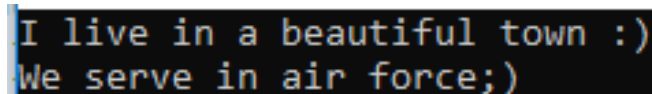
```
Please, the name of country: ukraine
Incorrect entry, writing with large letters(A - Z) and without double spaces :
Please, the name of country: _
```



```
Please, the name of country: ukraine
Incorrect entry, writing with large letters(A - Z) and without double spaces :
Please, the name of country: Ukraine
The name of country: Ukraine
Population: 40041
Area: 48467
Revenue: 46334
```

Рисунок № 6. Регулярні вірази

На рисунку № зображено результат роботи агрегації та композиції



```
I live in a beautiful town :)
We serve in air force;)
```

Рисунок № 7. Агрегація та композиція

Програма має декілька функцій:

1. Додавання елементу
2. Видалення елементу
3. Пошук по індексу
4. Запис в файл
5. Видалення масиву елементів

Висновок: порівняв поняття агрегація та композиція. Отримав знання про призначення ключових слів typedef та auto.

