

Лабораторна робота №4. Регулярні вирази

Тема: Регулярні вирази.

Мета: Отримати знання про базові регулярні вирази та досвід роботи по застосуванню їх на практиці.

ВИМОГИ

1.1 Інформація про розробника:

- Кліщов Б. Р.
- КІТ 102.8а

1.2 Загальне завдання

Поширити попередню лабораторну роботу наступним чином:

- при введенні інформації про базовий клас (нема різниці, чи з клавіатури, чи з файлу), організувати перевірку відповідності наступним критеріям з використанням регулярних виразів:
- можна вводити тільки кириличні символи, латинські символи, цифри, пропуски, розділові знаки;
- не повинно бути пропусків та розділових знаків, які повторюються;
- перше слово не повинно починатися з маленького символу;
- в клас-списку додати метод, що виводить на екран список усіх об'єктів, які мають одне або більше полів з щонайменше двома словами (перевірку організувати за допомогою регулярних виразів).

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*`.

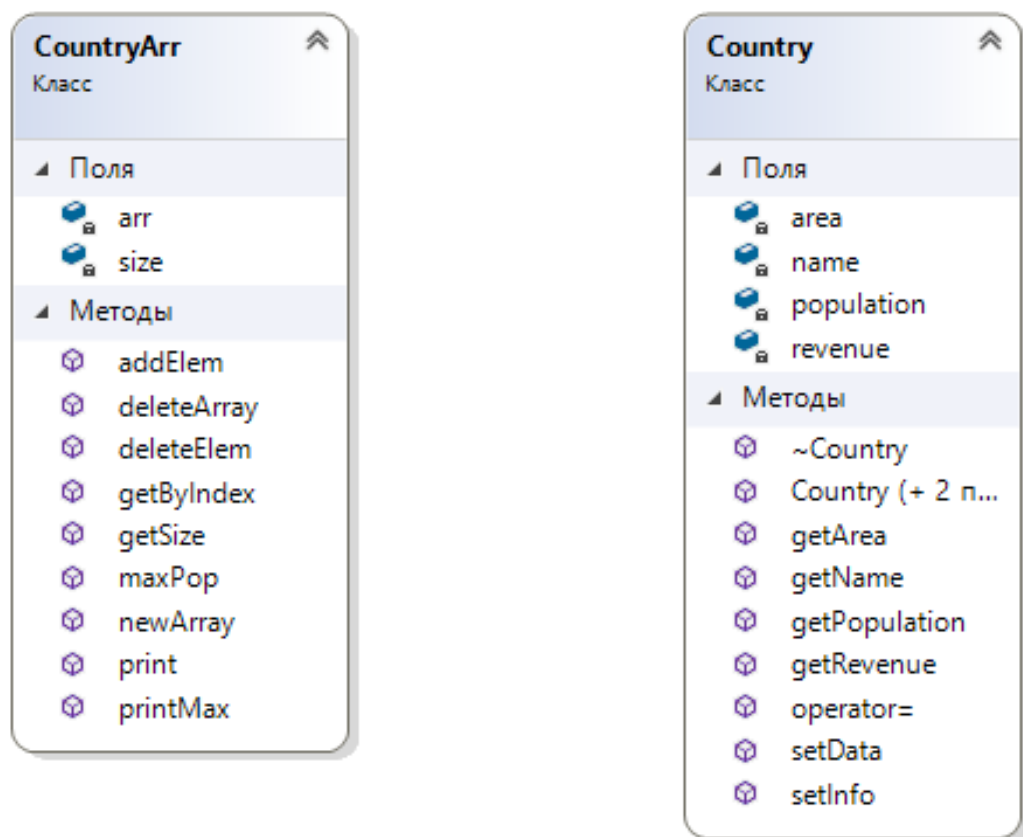
2. ОПИС ПРОГРАМИ

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.

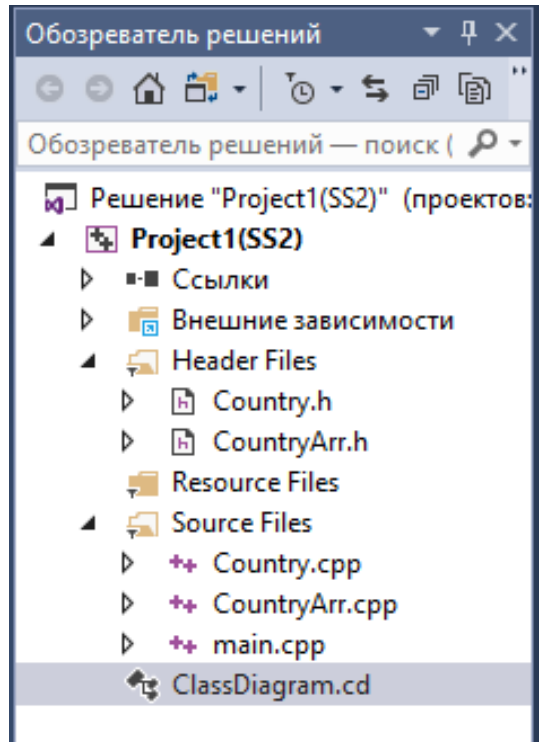
2.2 Опис логічної структури

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;
```

```
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();

        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;
            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: {
                int 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: {
        int 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;
    }
    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]");

```

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);  
}
```

}

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

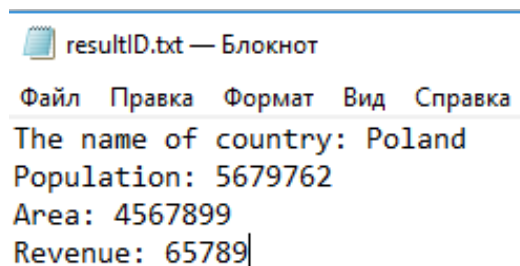
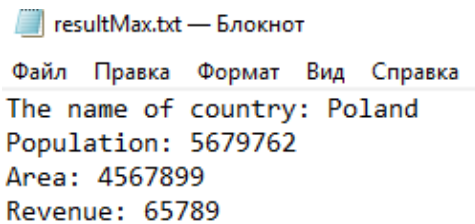


Рисунок № 4. Пошук по ID

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



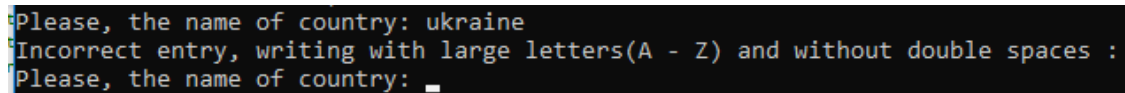
resultMax.txt — Блокнот

Файл Правка Формат Вид Справка

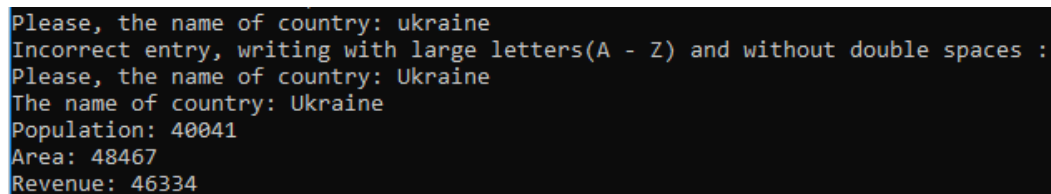
The name of country: Poland
Population: 5679762
Area: 4567899
Revenue: 65789

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

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`.