ЛАБОРАТОРНА РОБОТА №6. СПАДКУВАННЯ

Тема. Класи. Спадкування.

Мета. Отримати знання про парадигму ООП – спадкування. Навчитися застосовувати отримані знання на практиці.

1 ВИМОГИ

1.1 Розробник

- Котенко Сергій Миколайович;

- Студент групи КІТ 102.8(а);

- 27-05-2019р..

1.2 Загальне завдання

Модернізувати попередню лабораторну роботу шляхом:

- додавання класу-спадкоємця, котрий буде поширювати функціонал «базового класу» у відповідності до індивідуального завдання;

- додавання ще одного класу-списку, що буде керувати лише елементами класу-спадкоємця;

- в функціях базового класу та класу-спадкоємця 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 роботи «using» кожного необхідного класу:using std::string, using std::cout;

- в проекті не повинні використовуватися бібліотеки введення / виведення мови С, а також не повинні використовуватися рядки типу char*.

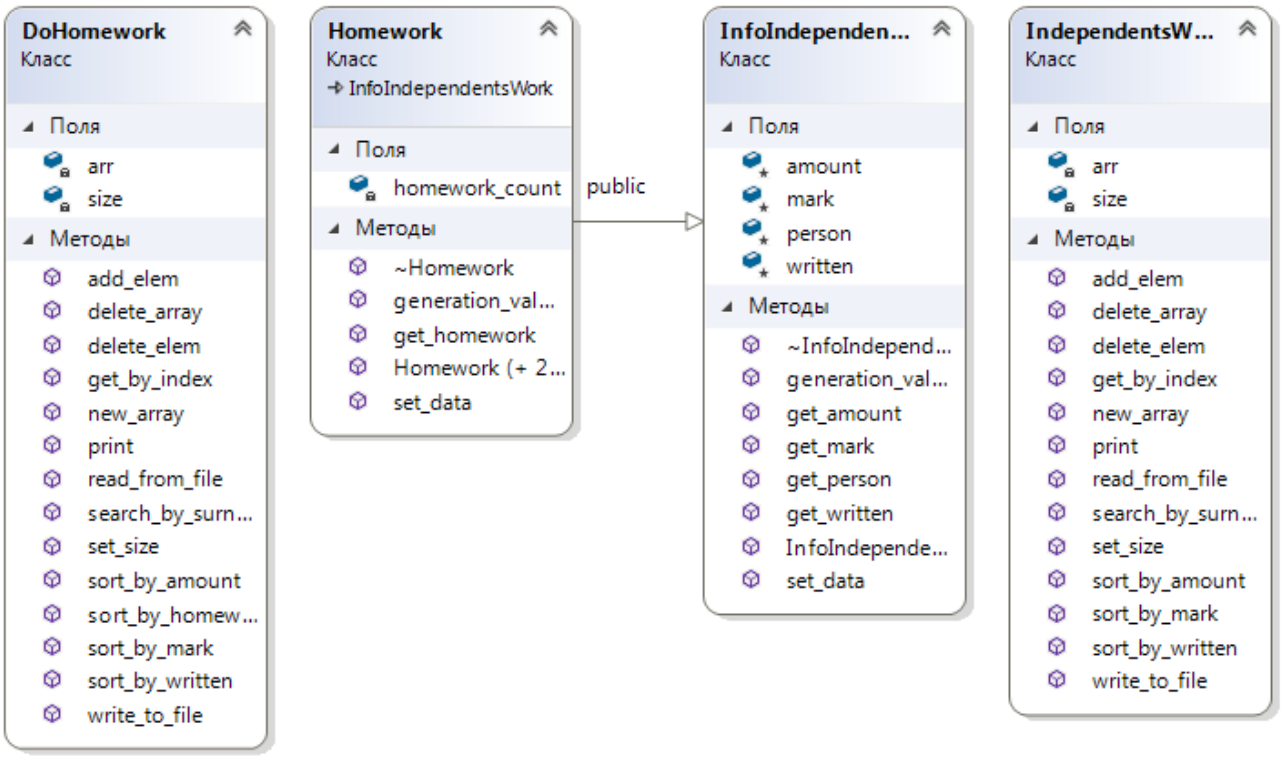
Індивідуальне завдання:

- В табл. 6.3 оберіть завдання для створення класу-спадкоємця у відповідності до номера у журналі групи.

10) Роботи студентів - кількість завдань для домашньої роботи

2 ОПИС ПРОГРАМИ

2.1 Опис логічної структури



Діаграма класу InfoIndependentsWork:

- ✓ ~InfoIndependentsWork - Деструктор класу;
- ✓ generation_values – Генерація випадкових значень;
- ✓ get_amount , get_mark , get_surname , get_written - Отримання даних;
- ✓ InfoIndependentsWork - Конструктор класу;
- ✓ set_data - Встановлення значень .

Діаграма класу IndependentsWork :

- ✓ add_elem - Додавання нового елементу;
- ✓ delete_array - Видалення масиву;
- ✓ delete_elem - Видалення елементу;
- ✓ get_by_index - Отримання даних за індексом;
- ✓ new_array - Створення масиву;
- ✓ print - Вивід даних на екран;
- ✓ read_from_file – Читання даних з файлу;
- ✓ search_by_surname – Пошук за прізвищем студента;
- ✓ set_size - Отримання розміру для створення масиву;
- ✓ sort_by_amount, sort_by_mark, sort_by_written – Сортування даних за певним критерієм;
- ✓ write_to_file – Запис результату у файл.

Діаграма класу Homework :

- ✓ ~ Homework - Деструктор класу;
- ✓ generation_values – Генерація випадкових значень;
- ✓ get_homework - Отримання даних;
- ✓ Homework - Конструктор класу;
- ✓ set_data - Встановлення значень .

Діаграма класу DoHomework :

- ✓ add_elem - Додавання нового елементу;
- ✓ delete_array - Видалення масиву;
- ✓ delete_elem - Видалення елементу;
- ✓ get_by_index - Отримання даних за індексом;
- ✓ new_array - Створення масиву;
- ✓ print - Вивід даних на екран;
- ✓ read_from_file – Читання даних з файлу;
- ✓ search_by_surname – Пошук за прізвищем студента;
- ✓ set_size - Отримання розміру для створення масиву;
- ✓ sort_by_amount, sort_by_mark, sort_by_written, sort_by_homework – Сортування даних за певним критеріем;
- ✓ write_to_file – Запис результату у файл.

2.2 Фрагменти коду

```
class Homework : public InfoIndependentsWork {
private:
    int homework_count;
public:
    Homework();
    Homework(int amount, int written, int mark, std::string person, int homework_count);
    Homework(const Homework &obj);

    int getHomework();
    void generation_values(std::string s);
    void setData(int homework_count,int amount, int written, int mark, std::string person);

    ~Homework();
};
```

Рисунок 2.1 – Клас спадкоємець

```
class DoHomework {
private:
    int size;
    Homework *arr;
public:
    void setSize(int size);
    void newArray(std::string *s);
    void print();
    void addElem(int amount, int written, int mark, int homework_count, std::string person);
    void deleteElem(int l);
    void getByIndex(int index);
    void deleteArray();
    void readFromFile(std::string *person);
    void writeToFile();
    void searchBySurname(std::string search_person);
    void sort_by_mark(bool(*sort)(int a, int b));
    void sort_by_amount(bool(*sort)(int a, int b));
    void sort_by_written(bool(*sort)(int a, int b));
    void sort_by_homework(bool(*sort)(int a, int b));
};
```

Рисунок 2.2 – Клас, що керує елементами класу спадкоємця

3 ВАРІАНТИ ВИКОРИСТАННЯ

3.1 Опис поведінки програми

Програма працює наступнім чином:

1) Введення користувачем кількості вивідних даних, створення масиву даних та виведення на екран

2) Вивід на екран можливих опцій програми , обирання користувачем опції:

2.0) Вихід з програми

2.1) Додавання нового елементу

2.2) Видалення певного елементу

2.3) Пошук за індексом

2.4) Пошук за прізвищем

2.5) Сортування за певним критерієм та у певному порядку

3) Перевірка на витоки пам’яті

3.2 Ілюстрація роботи програми

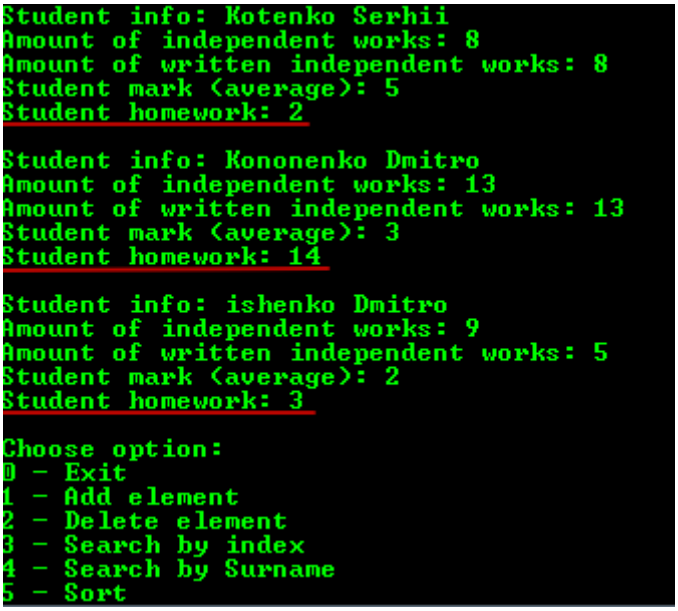


Рисунок 3.1 – Вивід даних з новою інформацією (кількість домашніх робіт)

ВИСНОВОК

В інтегрованому середовищі *Visual Studio* розроблена програма мовою C++. Виконання програми дозволяє продемонструвати коректність роботи програм для створення класів спадкоємців та їх використання.