

ЗВІТ

Лабораторна робота №2. Перевантаження методів

Тема. Класи. Конструктори та деструктори. Перевантаження методів.

Мета. Отримати базові знання про класи, конструктори та деструктори. Дослідити механізм створення та видалення об'єктів.

1 ВИМОГИ

1.1 Розробник

- Котенко Сергій Миколайович;

- Студент групи КІТ 102.8(а);

- 10-03-2019р..

1.2 Загальне завдання

Поширити попередню лабораторну роботу наступним чином:

в базовому класі необхідно додати: мінімум одне поле типу `char*`;

конструктор за замовчуванням, копіювання та конструктор з аргументами;

деструктор;

в клас-список потрібно додати метод обходу масиву для виконання індивідуального завдання.

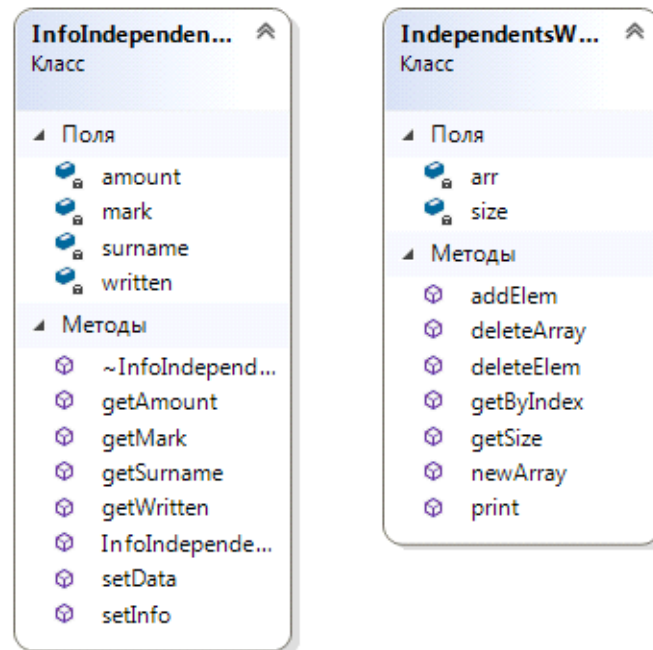
1.3 Індивідуальне завдання

В табл. 2.1 оберіть завдання для обходу колекції по варіанту у відповідності до номера у журналі групи.

(10 | Самостійна робота | Визначити, яку кількість домашніх завдань виконує студент за семестр)

2 ОПИС ПРОГРАМИ

2.1 Опис логічної структури



Діаграма класу InfoIndependentsWork - містить у собі наступні методи:

- ✓ getAmount , getMark , getSurname , getWritten - Отримання даних;
- ✓ setData , setInfo - Встановлення значень , перший метод генерує значення;
- ✓ InfoIndependentsWork - Конструктор класу;
- ✓ ~InfoIndependentsWork - Деструктор класу;
- ✓

Діаграма класу IndependentsWork - містить у собі наступні методи:

- ✓ getSize - Отримання розміру для створення масиву;
- ✓ newArray - Створення масиву;
- ✓ print - Вивід даних на екран;
- ✓ addElem - Додавання нового елементу;
- ✓ deleteElem - Видалення елементу;
- ✓ getByIndex - Отримання даних за індексом;
- ✓ deleteArray - Видалення масиву;

2.2 Фрагменти коду

```
void IndependentsWork::addElem(int amount, int written, int mark, char *surname) {  
    InfoIndependentsWork *mas = new InfoIndependentsWork[size + 1];  
  
    for (int i = 0; i < size; i++) {  
        mas[i] = IndependentsWork::arr[i];  
    }  
  
    size++;  
  
    mas[size - 1].setData(amount, written, mark, surname);  
  
    delete[] arr;  
    arr = mas;  
}
```

Рисунок 2.1 - Додавання нового елементу

```
void IndependentsWork::deleteElem(int l) {  
    if (size < 2) {  
        std::cout << "                Error                " << std::endl;  
        std::cout << " You cant delete last element " << std::endl;  
        system("pause");  
        return;  
    }  
  
    if (l - 1 >= size) {  
        std::cout << "                Error                " << std::endl;  
        std::cout << " You cant enter index more then size of array " << std::endl;  
        system("pause");  
        return;  
    }  
  
    size--;  
    InfoIndependentsWork* mas = new InfoIndependentsWork[size];  
  
    int j = 0;  
    for (int i = 0; i < l - 1; i++) {  
        mas[i] = IndependentsWork::arr[j];  
        j++;  
    }  
    j++;  
    for (int i = l - 1; i < size; i++) {  
        mas[i] = IndependentsWork::arr[j];  
        j++;  
    }  
  
    arr = mas;  
}
```

Рисунок 2.2 - Видалення елементу за індексом

```

void IndependentsWork::getByIndex(int index) {
    if (index-1 >= size) {
        std::cout << std::endl << "Error" << std::endl << std::endl;
        return;
    }

    std::cout << std::endl;
    std::cout << "-----" << std::endl;
    std::cout << "Student surname: " << arr[index - 1].getSurname() << std::endl;
    std::cout << "Amount of independent works: " << arr[index - 1].getAmount() << std::endl;
    std::cout << "Amount of written independent works: " << arr[index - 1].getWritten() << std::endl;
    std::cout << "Student mark (average): " << arr[index - 1].getMark() << std::endl << std::endl;
    std::cout << "-----" << std::endl;
    std::cout << std::endl << std::endl;
}

```

Рисунок 2.3 – Вивід елементу за індексом

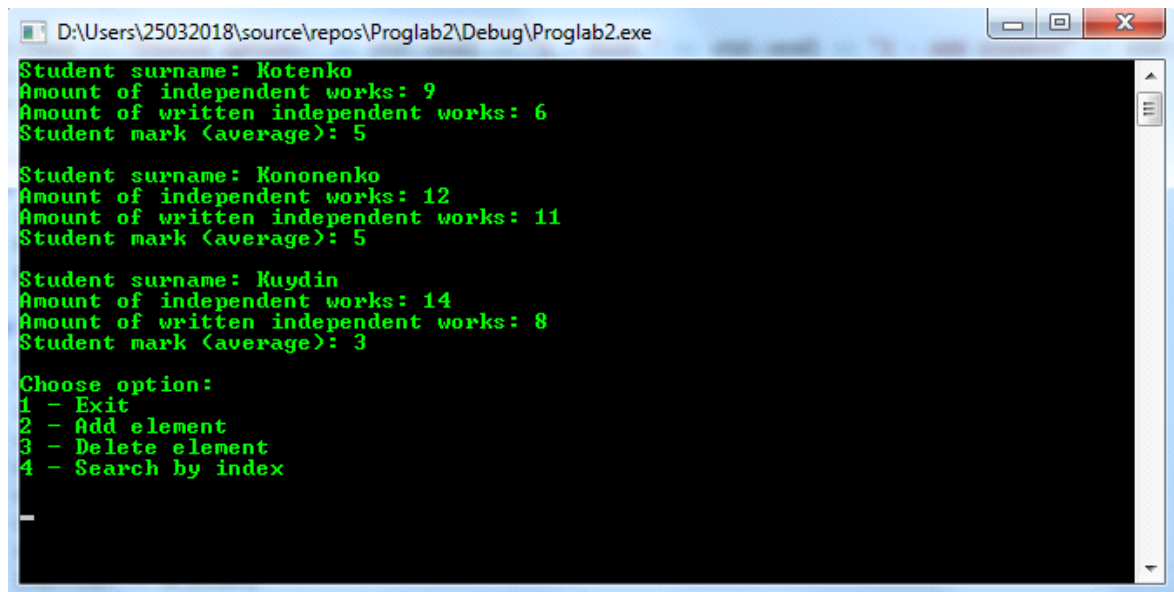
3 ВАРІАНТИ ВИКОРИСТАННЯ

3.1 Опис поведінки програми

Програма працює наступним чином:

- 1) Ввід користувачем кількості вивідних даних, створення масиву даних та виведення на екран
- 2) Вивід на екран можливих опцій програми , обирання користувачем опції:
 - 2.1) Вихід з програми
 - 2.2) Додавання нового елементу
 - 2.3) Видалення певного елементу
 - 2.4) Пошук за індексом
- 3) Перевірка на витоки пам'яті

3.2 Ілюстрація роботи програми



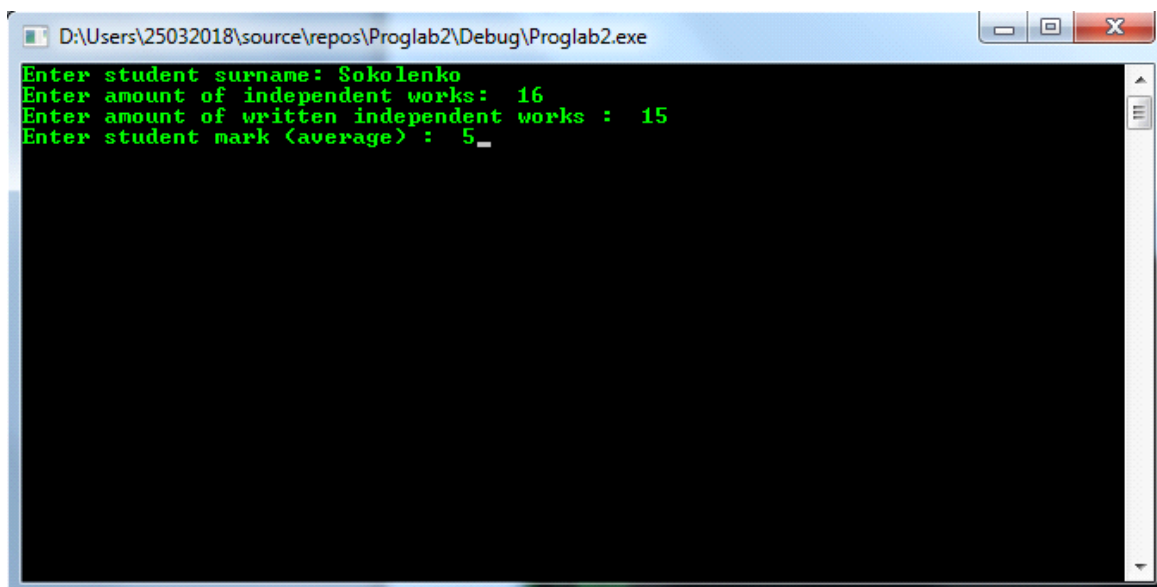
```
D:\Users\25032018\source\repos\Proglab2\Debug\Proglab2.exe
Student surname: Kotenko
Amount of independent works: 9
Amount of written independent works: 6
Student mark (average): 5

Student surname: Kononenko
Amount of independent works: 12
Amount of written independent works: 11
Student mark (average): 5

Student surname: Kuydin
Amount of independent works: 14
Amount of written independent works: 8
Student mark (average): 3

Choose option:
1 - Exit
2 - Add element
3 - Delete element
4 - Search by index
_
```

Рисунок 3.1 – Створений масив даних та можливі опції роботи з програмою



```
D:\Users\25032018\source\repos\Proglab2\Debug\Proglab2.exe
Enter student surname: Sokolenko
Enter amount of independent works: 16
Enter amount of written independent works : 15
Enter student mark (average) : 5_
_
```

Рисунок 3.2 – Додавання нового елементу(Введення даних про студента)

```
D:\Users\25032018\source\repos\Proglab2\Debug\Proglab2.exe

-----
Student surname: Kotenko
Amount of independent works: 9
Amount of written independent works: 6
Student mark (average): 5
-----

Choose option:
1 - Exit
2 - Add element
3 - Delete element
4 - Search by index
```

Рисунок 3.3 – Пошук за індексом (Результат виконання пошуку)

ВИСНОВОК

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