

## **Лабораторна робота №16**

**Тема.** Системна п'ярта з динамічною пам'яттю.

**Мета роботи.** Дослідити особливості мови C++ при роботі з динамічною пам'яттю.

### **1 ВИМОГИ**

#### **1.1 Розробник**

Інформація про розробника:

- Кулик Данііл Ігорович
- НТУ “ХПІ” КІТ 102.8а
- Варіант 12

#### **1.2 Загальне завдання**

Маючи класи з прикладної області РЗ (базовий клас), перевантажити оператори `new` / `new []` та `delete` / `delete []`. Продемонструвати їх роботу і роботу операторів розміщення `new` / `delete` при розробці власного менеджера пам'яті (сховища).

Детальна інформація про власне сховище: є статично виділений масив заданого обсягу. Організувати виділення і звільнення пам'яті елементів ієрархії класів тільки в рамках даного сховища

## **2 ОПИС ПРОГРАМИ**

#### **2.1 Функціональне призначення**

За допомогою цієї програми можна створити масив об'єктів, додавати та видаляти об'єкти, виводити вміст масиву на екран та вивід об'єкта по індексу. Також у цій програмі реалізоване зручне меню спілкування з користувачем.

#### **2.2 Важливі фрагменти програми**

На рисунку №1 зображено діаграми класів.

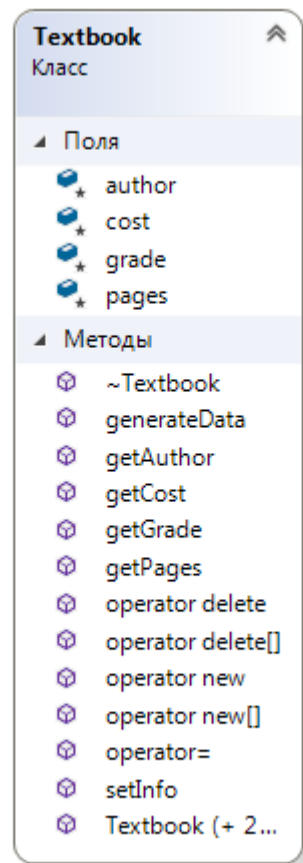


Рисунок №1 – діаграма класу Textbook

Методи класу **Textbook**:

- ~Textbook - Деструктор класу;
- generateData– Генерація випадкових значень;
- getPages , getGrade, getCost , getAuthor - Отримання даних;
- Textbook- Конструктор класу;
- operator= - Перевантаження оператора присвоювання;
- setInfo - Встановлення значень;
- operator delete - Перевантаження оператора delete;
- operator delete[] - Перевантаження оператора delete[];
- operator new - Перевантаження оператора new;

- operator new[]- Перевантаження оператора new[];

На рисунку №2 зображено наявність перевантажених операторів new, new[], delete, delete[].

```
void* Textbook::operator new(size_t size) {
    void* pointer = malloc(size);
    if (pointer == nullptr) {
        throw std::bad_alloc();
    }
    cout << "Memory was allocated for " << size << " elements." << endl;
    return pointer;
}

void Textbook::operator delete(void *pointer) {
    free(pointer);
    cout << "The memory has been freed successfully." << endl << endl;
}

void* Textbook::operator new[](size_t size) {
    void* pointer = malloc(size);
    if (pointer == nullptr) {
        throw std::bad_alloc();
    }
    cout << "Memory was allocated for " << size << " elements." << endl;
    return pointer;
}

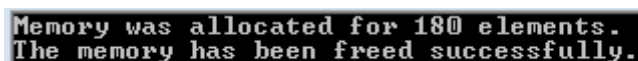
void Textbook::operator delete[](void *pointer) {
    free(pointer);
    cout << "The memory has been freed successfully." << endl << endl;
}
```

Рисунок №2 – перевантаженні оператори

### 3 ВАРІАНТИ ВИКОРИСТАННЯ

Програма може бути використана для створення масиву об'єктів. Програма має методи додавання, видалення об'єктів, вивід об'єкта по індексу з масиву та усіх об'єктів на екран, читання та запис об'єктів масиву з файлу, а також сортування за одним із властивостей об'єкта. Меню робить роботу з цією програмою зручною.

1.Результат роботи програми зображено на рисунку №3



```
Memory was allocated for 180 elements.
The memory has been freed successfully.
```

Рисунок №3 – результат роботи програми

## ***ВИСНОВКИ***

В інтегрованому середовищі VisualStudio розроблена програма мовою C++. Засоби налагодження дозволяють за допомогою меню спілкування створити масив об'єктів та змінювати його за допомогою методів класів.