Звіт

Автор: Чугунов В.Ю

КІТ-119а

Лабораторна робота №2

КЛАСИ. КОНСТРУКТОРИ ТА ДЕСТРУКТОРИ. ПЕРЕВАНТАЖЕННЯ МЕТОДІВ

**Мета:** Отримати базові знання про класи, конструктори та деструкто-ри. Дослідити механізм створення та видалення об’єктів.

1 ЗАВДАННЯ ДО РОБОТИ

**Основне завдання.** Поширити попередню лабораторну роботу таким чином:  
 ‒ в базовому класі необхідно додати:

‒ мінімум одне полу типу char\*;

‒ конструктор за замовчуванням, копіювання та конструктор з аргументами;

‒ деструктор;

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

**Індивідуальне завдання.** Визначити, яку кількість домашніх завдань виконує студент за семестр.

2 РОЗРОБКА АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ

**2.1. Внесені зміни**

**Клас Lessons** Додано:

В приватне поле класу:  
 int numOfWorks – кількість виконаних робіт за семестр  
1. Конструктор за замовчуванням:  
Lessons()

2. Конструктор з параметрами:  
Lessons(const string type, const string name, const string ID, const int time, const int qty)

3. Конструктор копіювання:  
Lessons(const Lessons & copyObject)

4. Деструктор:  
~Lessons()

**Клас Works** Додано:

1. Метод для обходу списку:  
   int Works::WorksForSem()

3 ТЕКСТ ПРОГРАМИ

**Текст файлу main.cpp**

#include <iostream>

#include "Lessons.h"

#include "Works.h"

#include "Func.h"

int main()

{

Lessons lesson;

Works work;

int choose;

do {

cout << endl;

MenuBar();

cin >> choose;

if (choose < 1 || choose > 4)

{

cout << "\nError (choose 1-4).\n";

continue;

}

switch (choose)

{

case 1:

lesson.generateLesson();

work.addWork(lesson);

break;

case 2:

cout << "Enter a psition of lesson you want to deleting: ";

int num;

while (true)

{

cin >> num;

if ((work.get\_size() == 0))

break;

if ((num - 1) >= work.get\_size() || (num - 1) < 0)

cout << "Error (pos is higher than qty of subs), try again: ";

else break;

}

work.removeWork(num-1);

cout << endl;

break;

case 3:

work.showAll();

break;

case 4:

work.endOfProgramm();

cout << "\nStudent performs " << work.WorksForSem() << " tasks per semester.";

cout << "\nThank that you using my code!\n";

break;

}

} while (choose != 4);

system("PAUSE");

return 0;

}

**Lessons.cpp**

#include <iostream>

#include "Lessons.h"

Lessons::Lessons()

{

audienceID = "NULL";

lecturerName = "NULL";

lessonType = "NULL";

lessonStartTime = 0;

numOfWorks = 0;

cout << "\nDefault constructor of " << this << " was called.\n"

}

Lessons::Lessons(const string type, const string name, const string ID, const int time, const int qty)

{

audienceID = ID;

lecturerName = name;

lessonType = type;

lessonStartTime = time;

numOfWorks = qty;

cout << "\nConstructor with the parameters of " << this << " was called.\n";

}

Lessons::Lessons(const Lessons & copyObject)

{

this->audienceID = copyObject.audienceID;

this->lecturerName = copyObject.lecturerName;

this->lessonStartTime = copyObject.lessonStartTime;

this->lessonType = copyObject.lessonType;

this->numOfWorks = copyObject.numOfWorks;

cout << "\nCopying constructor of " << this << " was called.\n";

}

Lessons::~Lessons()

{

cout << "\nThe destructor of " << this << " was called.\n";

}

void Lessons::generateLesson()

{

cout << "\nEnter the subject name: ";

cin >> lessonType;

cout << "Enter lecturer name: ";

cin >> lecturerName;

cout << "Enter an audience ID: ";

cin >> audienceID;

cout << "Enter lesson start time: ";

cin >> lessonStartTime;

cout << "Enter the number of works student wrote for the semester: ";

cin >> numOfWorks;

}

**Works.cpp**

#include <iostream>

#include "Works.h"

void Works::addWork(Lessons& lesson)

{

if (size == 0)

{

lessons = new Lessons[size + 1];

lessons[size] = lesson;

sum += lessons[size].get\_numOfWorks();

size++;

}

else {

Lessons \*newLessons = new Lessons[size + 1];

for (int i = 0; i < size; i++) {

newLessons[i] = lessons[i];

}

newLessons[size] = lesson;

delete[]lessons;

lessons = newLessons;

sum += lessons[size].get\_numOfWorks();

size++;

}

}

void Works::removeWork(int pos)

{

if (size == 0)

cout << "\nError (list is empty)\n";

else {

Lessons \*newLessons = new Lessons[size - 1];

for (int i = 0, j = 0; i < size; i++, j++)

{

if (i == pos) {

j--;

continue;

}

newLessons[j] = lessons[i];

}

delete[](lessons);

lessons = newLessons;

size--;

}

}

void Works::showAll()

{

if (size == 0)

cout << "\nError (the list is empty)\n";

else {

for (int i = 0; i < size; i++)

{

printf("\nInfo about lesson #%d: \n", i+1);

cout << "Subject: " << lessons[i].get\_lessonType() << endl;

cout << "Lecturer: " << lessons[i].get\_lecturerName() << endl;

cout << "Audience ID: " << lessons[i].get\_audienceID() << endl;

cout << "Lesson start time: " << lessons[i].get\_lessonStartTime() << endl;

cout << "The number of works student wrote for the semester: " << lessons[i].get\_numOfWorks() << endl;

}

}

}

int Works::WorksForSem()

{

return sum;

}

void Works::endOfProgramm()

{

delete[]lessons;

lessons = nullptr;

}

const int Works::get\_size() const {

return size;

}

**Lessons.h**

#include <string>

using namespace std;

#pragma once

class Lessons

{

public:

Lessons();

Lessons(const string type, const string name, const string ID, const int time, const int qty);

Lessons(const Lessons & copyObject);

~Lessons();

void set\_audienceID(const string index);

void set\_lecturerName(const string name);

void set\_lessonType(const string type);

void set\_numOfWorks(const int number);

void set\_lessonStartTime(const int time);

const string get\_audienceID() const;

const string get\_lecturerName() const;

const string get\_lessonType() const;

const int get\_numOfWorks() const;

const int get\_lessonStartTime() const;

void generateLesson();

private:

string audienceID;

string lecturerName;

string lessonType;

int numOfWorks;

int lessonStartTime;

};

**Works.h**

#pragma once

#include "Lessons.h"

class Works : public Lessons

{

public:

void addWork(Lessons& lesson);

void removeWork(int pos);

void showAll();

int WorksForSem();

void endOfProgramm();

const int get\_size() const;

private:

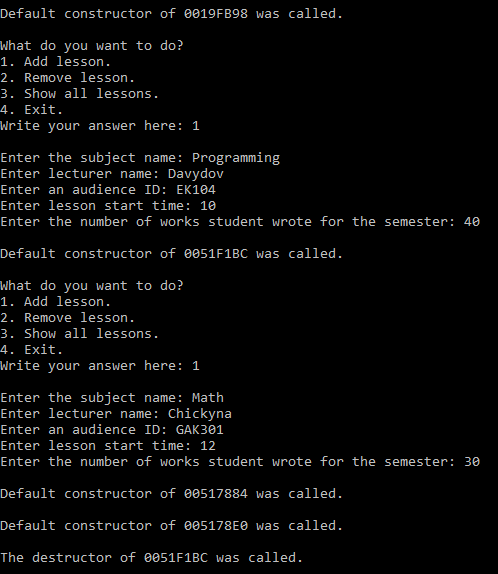
Lessons\* lessons;

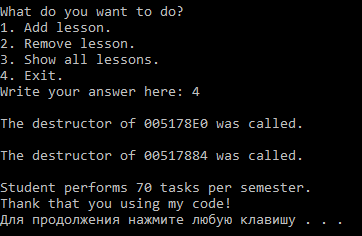
int size = 0;

int sum = 0;

};

4 РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ

Рисунок 1 – Результат роботи програми (частина 1)

  
Рисунок 2 – Результат роботи програми (частина 2)

ВИСНОВКИ

При виконанні лабораторної роботи до попередньо розробленого функціоналу (лабораторна робота №1) було додано наступне:

1. До класу Lessons:
2. конструктор за замовчуванням;
3. конструктор копіювання;
4. конструктор з параметрами;
5. деструктор;
6. До класу Works:
7. метод для обходу списку.

Також я дослідив механізм створення та видалення об’єктів, який полягає у використанні різних конструкторів та деструкторів.