Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

Дисципліна: ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

**Лабораторна робота №2**

**Тема: «Конструктори і деструктори»**

|  |
| --- |
| Виконав: ст. гр. КI-23 |
| Карпець О. Р. |
| Перевірив викладач:  Козірова Н. Л. |
|  |

Кропивницький 2024

**Тема:** Конструктори і деструктори.

**Мета:**ознайомитись з основними поняттями конструктор і деструктор в ООП та навчитись їх програмно реалізовувати мовою С++.

**Варіант 7**

**Завдання:**

**Завдання 1**

1. Реалізуйте конструктор за замовчуванням, конструктор з параметрами та копіюючий конструктор для вашого класу з лабораторної роботи 1.

• Конструктор за замовчуванням має встановлювати значення полів за замовчуванням.

• Конструктор з параметрами має приймати значення для кожного поля.

• Копіюючий конструктор має копіювати значення полів з іншого об'єкта класу. Публічні методи класу:

2. Реалізуйте деструктор для класу. Деструктор має виводити повідомлення про знищення об'єкта.

3. У функції main створіть об'єкт за допомогою конструктора за замовчуванням та виведіть значення його полів.

4. Створіть новий об'єкт за допомогою конструктора з параметрами та встановіть значення для полів. Виведіть значення полів цього об'єкта.

5. Створіть ще один об'єкт і скопіюйте значення полів з першого об'єкта за допомогою копіюючого конструктора. Виведіть значення полів цього об'єкта

6. Завершіть функцію main, що призведе до виходу з області видимості створених об'єктів і виклику їх деструкторів. Переконайтесь, що повідомлення про знищення об'єктів виводяться.

**Завдання 2**

Розробіть клас «Множина (символів)» – Set потужності 𝑛. Клас повинен містити конструктори. Реалізуйте методи для визначення належності заданого елемента множині, перетину, об'єднання та різниці двох множин. Створіть масив об'єктів і передавайте пари об'єктів у функцію, яка будує множину, що складається з елементів, які входять тільки в одну з заданих множин і повертає її в головну програму.

**Лістинг movie.h:**

#ifndef MOVIE\_H

#define MOVIE\_H

#include <string>

#include <iostream>

using namespace std;

class Movie

{

public:

Movie();

Movie(string newTitle, string newDirector, int newDuration, int newYear);

Movie(Movie& copy);

~Movie();

bool setTitle(string newTitle);

string getTitle() { return title; }

bool setDirector(string newDirector);

string getDirector() { return director; }

bool setYear(int newYear);

int getYear() { return year; }

bool setDuration(int newDuration);

int getDuration() { return duration; }

private:

string title;

string director;

int year;

int duration;

bool checkStrings(string checkString);

bool checkInts(int checkInt);

};

#endif // MOVIE\_H

**Лістинг set.h:**

#ifndef SET\_H

#define SET\_H

#include <iostream>

#include <vector>

using namespace std;

class Set {

private:

vector<char> elements;

public:

Set();

Set(string elems);

bool contains(char checkElem);

Set plusSets(Set otherSet);

Set intersectionSets(Set otherSet);

Set differenceSet(Set otherSet);

Set symmetricDifference(Set other);

void print();

};

#endif // SET\_H

**Лістинг set.cpp:**

#include "set.h"

Set::Set() {}

Set::Set(string elems)

{

for (char elem : elems)

{

if (!contains(elem))

{

elements.push\_back(elem);

}

}

}

bool Set::contains(char checkElem)

{

for (char elem : elements)

{

if (elem == checkElem)

{

return true;

}

}

return false;

}

Set Set::plusSets(Set otherSet)

{

Set result = \*this;

for (char elem : otherSet.elements)

{

if (!result.contains(elem))

{

result.elements.push\_back(elem);

}

}

return result;

}

Set Set::intersectionSets(Set otherSet)

{

Set result;

for (char elem : elements)

{

if (otherSet.contains(elem))

{

result.elements.push\_back(elem);

}

}

return result;

}

Set Set::differenceSet(Set otherSet)

{

Set result;

for (char elem : elements)

{

if (!otherSet.contains(elem))

{

result.elements.push\_back(elem);

}

}

return result;

}

Set Set::symmetricDifference(Set other)

{

Set result;

for (char elem : elements)

{

if (!other.contains(elem))

{

result.elements.push\_back(elem);

}

}

for (char elem : other.elements)

{

if (!contains(elem))

{

result.elements.push\_back(elem);

}

}

return result;

}

void Set::print()

{

cout << "{ ";

for (char elem : elements) {

cout << elem << " ";

}

cout << "}" << endl;

}

**Лістинг movie.cpp:**

#include "movie.h"

Movie::Movie()

{

this->setDirector("default");

this->setTitle("default");

this->setDuration(1);

this->setYear(1);

}

Movie::Movie(string newTitle, string newDirector, int newDuration, int newYear)

{

this->setDirector(newDirector);

this->setTitle(newTitle);

this->setDuration(newDuration);

this->setYear(newYear);

}

Movie::Movie(Movie &copy)

{

this->setDirector(copy.getDirector());

this->setTitle(copy.getTitle());

this->setDuration(copy.getDuration());

this->setYear(copy.getYear());

}

Movie::~Movie()

{

cout << "Class Movie has been deleted" << endl;

}

bool Movie::setTitle(string newTitle)

{

if (checkStrings(newTitle))

{

this->title = newTitle;

return true;

}

return false;

}

bool Movie::setDirector(string newDirector)

{

if (checkStrings(newDirector))

{

this->director = newDirector;

return true;

}

return false;

}

bool Movie::setYear(int newYear)

{

if (checkInts(newYear))

{

this->year = newYear;

return true;

}

return false;

}

bool Movie::setDuration(int newDuration)

{

if (checkInts(newDuration))

{

this->duration = newDuration;

return true;

}

return false;

}

bool Movie::checkStrings(string checkString)

{

if (checkString.length() > 0)

{

return true;

}

return false;

}

bool Movie::checkInts(int checkInt)

{

if (checkInt > 0)

{

return true;

}

return false;

}

**Лістинг main.cpp:**

#include <QCoreApplication>

#include "movie.h"

#include "set.h"

Set buildSymmetricDifference(Set set1, Set set2) {

return set1.symmetricDifference(set2);

}

int main(int argc, char \*argv[])

{

QCoreApplication a(argc, argv);

char order = '\0';

int userInputInt;

string userInputStr;

Movie \*defaultMovie = new Movie();

Movie \*params = new Movie("Cyberpunk 2077", "CDPR", 3600, 2077);

Movie \*copy = new Movie(\*params);

cout << "Default/Param/Copy/Set (D/P/C/S))?" << endl;

cin >> order;

cin.ignore();

switch (order)

{

case 'd':

case 'D':

cout << "Defalut Title: " << defaultMovie->getTitle() << endl;

cout << "Defalut Year: " << defaultMovie->getYear() << endl;

cout << "Defalut Director: " << defaultMovie->getDirector() << endl;

cout << "Defalut Duration: " << defaultMovie->getDuration() << endl;

break;

case 'p':

case 'P':

cout << "Defalut Title: " << params->getTitle() << endl;

cout << "Defalut Year: " << params->getYear() << endl;

cout << "Defalut Director: " << params->getDirector() << endl;

cout << "Defalut Duration: " << params->getDuration() << endl;

break;

case 'c':

case 'C':

cout << "Defalut Title: " << copy->getTitle() << endl;

cout << "Defalut Year: " << copy->getYear() << endl;

cout << "Defalut Director: " << copy->getDirector() << endl;

cout << "Defalut Duration: " << copy->getDuration() << endl;

case 's':

case 'S':

Set set1("abcfgnrfhegwrghgtrhgdbfdbdf");

Set set2("cdefgwfgwfbfgnaaaaaaaaaaaaaaaaaffffpbn");

cout << "SET 1: ";

set1.print();

cout << "SET 2: ";

set2.print();

Set symDiff = buildSymmetricDifference(set1, set2);

cout << "SYMMETRIC DIFFERENCE: ";

symDiff.print();

}

cout << "DESTRUCT:" << endl << endl;

defaultMovie->~Movie();

copy->~Movie();

params->~Movie();

return 0;

}

**Висновок:**

В цій лабораторній роботі реалізував різні типи конструкторів: за замовчуванням, з параметрами та копіюючий конструктор. Вони дозволяють ініціалізувати об'єкти з різними значеннями та правильно копіювати дані між ними. Деструктор забезпечив коректне виведення об'єктів з пам'яті.

Також створив клас «Множина (символів)» і додали методи для перевірки елементів, перетину, об'єднання та різниці множин. Це зміцнило наші навички роботи з конструкторами та множинами в об'єктно-орієнтованому програмуванні.