МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕХНІЧНИЙ КОЛЕДЖ ТНТУ ІМЕНІ ІВАНА ПУЛЮЯ

ВІДДІЛЕННЯ ЕЛЕКТРОННИХ АПАРАТІВ

Циклова комісія програмних систем і комплексів

**ЗВІТ**

про виконання лабораторних робітз дисципліни:

**«ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ»**

Студента  3  курсу групи КН-321 спеціальності 122 «Комп’ютерні науки»

Поживотенко В.П.

(прізвище та ініціали)

Перевірив: Р.О. Слободян

(підпис)

Тернопіль – 2020

**Лабораторна робота №11**

**Тема:** Вивчення механізмів оброблення виняткових ситуацій.

**Мета:** навчитись обробляти ситуації появлення виняткових ситуацій, вивчити механізми їх оброблення.

**Завдання 1:**

Додайте клас винятків до програми ARROVER таким чином, щоб індекси, що виходять за межі масиву, викликали генерацію винятку. Блокпастка catch може виводити користувачеві повідомлення про помилку.

**КОД ПРОГРАМИ**

#include <iostream>

#include <stdio.h>

#include <string>

#include <math.h>

#include <string.h>

#define LIMIT 100

using namespace std;

class **Waiver**

{

public:

char str[128];

**Waiver**(char\* text=0)

{

strcpy(str,text);

}

};

class **Array**

{

int arr[LIMIT];

public:

int& operator [](int k)

{

try

{

if (k < 0 || k>= LIMIT) throw Waiver("index out of bounds");

return arr[k];

}

catch(Waiver e)

{

cout << e.str << endl;

system("pause");

exit(1);

}

}

};

int **main**()

{

Array save;

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

save[i] = cos (i-6)\* sin (2 \* pow(i,5))\* 60;

for (int i = 83; i<1000;i++)

cout << "element" << ' ' << i <<' ' << "is" << ' ' << save[i] << '\n';

system("pause");

return 0;

}

**РЕЗУЛЬТАТ ВИКОНАННЯ ПРОГРАМИ**

****

**Завдання 2:**

Модифікуйте програму з попереднього завдання таким чином, щоб в повідомлення про помилку входила інформація про значення індексу, який призвів до збою.

**КОД ПРОГРАМИ**

#include <iostream>

#include <stdio.h>

#include <string>

#include <math.h>

#include <string.h>

#define LIMIT 100

using namespace std;

class **Waiver**

{

public:

string str;

**Waiver**(string text)

{

str = text;

}

};

class **Array**

{

int arr[LIMIT];

public:

int& operator [](int k)

try

{

if (k < 0 || k>= LIMIT) throw Waiver("index" + to\_string(k) + "out of bounds");

return arr[k];

}

catch (Waiver e)

{

cout << e.str << endl;

system("pause");

exit(1);

}

};

int **main**()

{

Array save;

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

save[i] = cos(i-5) \* sin (2\*pow(i,3)) \* 50;

for (int i = 80; i < 1000; i++)

cout << "element" << ' ' << i << ' ' << "is" << ' ' << save[i] << endl;

system("pause");

return 0;

}

**РЕЗУЛЬТАТ ВИКОНАННЯ ПРОГРАМИ**

****

**Завдання 3:**

Модифікуйте текст програми, розробленої згідно індивідуального завдання лабораторної роботи №3, добавивши до нього (в місця можливого виникнення помилок) процедури оброблення виняткових ситуацій, які будуть генерувати об’єкти класу, який міститиме такі атрибути: місце виникнення помилки; значення, яке призвело до помилки; параметризований конструктор; перевизначену операції виводу, яка виводитиме значення на екран (або зберігатиме виняткову ситуацію у файл).

Створити клас – вектор, який має у закритій частині вказівник на дані цілого типу та кількість елементів. Визначити необхідні конструктори, деструктор. Перевантажити потокові операції введення і виведення елементів вектора, операції + , – , \* , = та [ ] .

**КОД ПРОГРАМИ**

#include <iostream>

#include <stdio.h>

#include <string>

#include <math.h>

#include <string.h>

using namespace std;

class **Waiver**

{

public:

string place;

string str;

**Waiver**(string k,string m)

{

place = k;

str = m;

}

};

class **Vector**{

int \*dataPointer;

int itemCount;

int items[5];

void **valid** (string place, int val)

{

try

{

if (val <=0)

{

Waiver e (place,to\_string(val) + "is not valid");

throw e;

}

}

catch (Waiver e)

{

cout << "Waiver in " << e.place << endl;

cout << "message" << e.str << endl;

exit(1);

}

}

public:

**Vector**()

{

\*dataPointer = 0;

int items[10];

}

int operator [] (int i) {

return items [i];

}

Vector operator + (int \*data) {

int sum = \*dataPointer + \*data;

Vector result = Vector();

result.SetDataPointer(&sum);

return result;

}

Vector operator \* (int \*data) {

int sum = \*dataPointer \* \*data;

Vector result = Vector();

result.SetDataPointer(&sum);

return result;

}

Vector operator - (int \*data) {

int sum = \*dataPointer - \*data;

Vector result = Vector();

result.SetDataPointer(&sum);

return result;

}

Vector operator = (int \*data) {

int sum = \*dataPointer = \*data;

Vector result = Vector();

result.SetDataPointer(&sum);

return result;

}

Vector operator / (int \*data) {

int sum = \*dataPointer / \*data;

Vector result = Vector();

result.SetDataPointer(&sum);

return result;

}

int **GetDataPointer** () {

return \*dataPointer;

}

void **SetDataPointer**(int \*data) {

\*dataPointer = \*data;

}

};

int **main**()

{

int val1 = 6;

int val2 = 9;

Vector value1 = Vector();

value1.SetDataPointer(&val1);

Vector value2 = Vector();

value2.SetDataPointer(&val2);

Vector sumResult = value1 + value2 ;

cout << "Sum is:" << endl;

cout << sumResult.GetDataPointer() << endl;

Vector multiplyResult = value1 \* value2;

cout << "Multiply is:" << endl;

cout << multiplyResult.GetDataPointer() << endl;

Vector devideResult = value1 / value2;

cout << "Devide is:" << endl;

cout << devideResult.GetDataPointer() << endl;

Vector minusResult = value1 - value2;

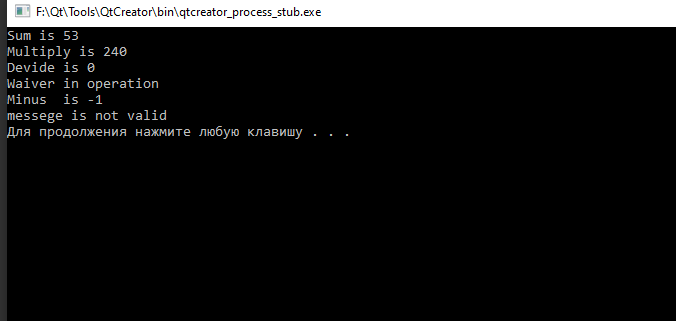
cout << "Minus is:" << endl;

cout << minusResult.GetDataPointer() << endl;

return 0;

}

**РЕЗУЛЬТАТ ВИКОНАННЯ ПРОГРАМИ**



**Висновок:** навчився обробляти ситуації появлення виняткових ситуацій, вивчив механізми їх оброблення.