 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Практичне заняття №7**

з дисципліни «Об’єктно-орієнтоване програмування»

на тему: «Шаблони»

Варіант №11

**Виконав:**

студент гр. БС-52

Розюк П.В.

**Перевірив:**

ст. викладач каф. БМК

Радогуз А.В.

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2017

**Завдання:**

1. Ознайомитися з тноретичними відомостями роботи обробки виняткових ситуацій мови С++.
2. Створити клас Exeption, що дозволяє обробляти зазначену в роботі виняткову ситуацію.
3. Розробити діагрмау класів в нотації UML та програмний додаток для демонстраціїї роботи програми в нормальному режимі та під час винекнення виняткової ситупції відповідно до свого варіанту.

**🞏 Практичне заняття без зауважень** **🞏 Практичне заняття має зауваження:**

**🞏 несвоєчасний захист**

**🞏 присутні зауваження до UML діаграми:**

**🞏 діаграма класу не відповідає коду**

**🞏 виконані не за стандартом:**

**🞏 атрибути**

**🞏 відношення**

**🞏 багатостатність**

**🞏 інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🞏 присутні зауваження до коду:**

**🞏 задача завдання вирішена хибно**

**🞏 код програми не компілюється**

**🞏 хибно задані специфікатори доступу**

**🞏 помилки у визначенні конструкторів / деструкторів**

**🞏 використано глобальні змінні**

**🞏 статичні змінні при роботі з масивами**

**🞏 оформлення коду**

**🞏 присутні зайві символи «{» та «}»**

**🞏 інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🞏 невірні відповіді на запитання:**

**🞏 №1 🞏 №2 🞏 №3 🞏 №4 🞏 №5**

**🞏 №6 🞏 №7 🞏 №8 🞏 №9 🞏 №10**

**🞏 маються інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Лістинг прогрмами:**

**Value.h**

#pragma once

class Value

{

private:

double a;

public:

Value(double a);

void multiplication(double result);

~Value() {};

};

**Value.cpp**

#include <iostream>

#include <fenv.h>

#include "Value.h"

using namespace std;

Value::Value(double a)

{

this->a = a;

double result;

result = a\*a;

multiplication(result);

}

void Value::multiplication(double result)

{

try

{

if (result = fetestexcept(FE\_OVERFLOW))

throw 1;

else

cout << a << " \* " << a << " = " << a\*a<<endl;

}

catch (int i)

{

cout << a << " \* " << a << " = " << "Error "<<i<<", overflow!" <<"\n\n";

}

}

**Main.cpp**

#include "Value.h"

#include <iostream>

void main()

{

Value A(1.0e100), B(1.0e200);

system("PAUSE");

}



**UML-діаграма:**

  
**Контрольні питання**

1. **Що називається винятковою ситуацією?**

Виняткова ситуація – це подія, що сталася під час виконання програми, в результаті чого подальше нормальне виконання програми стає неможливим.

1. **Що називається обробкою виняткової ситуації?**

Обробкою виняткової ситуації називається перевірка умов, що описують виняткову ситуацію, і реакція на її виникнення.

1. **Як здійснюється обробка виняткової ситуації в мові С++?**

Обробка виняткових ситуацій в мові С ++ використовує три ключових блока: *try*, *catch* та *throw*.

1. **Чи можна помістити блок *try* у функцію, залишивши блок *catch* для обробки виняткових ситуацій у функціїmain()?Відповідь пояснити.**

Ні, тому що блоки try та catch блоки нерозривні,тобто кожному блоку try має ставитись у відповідність блок catch.

Необхідно перемістити try- блок у main() або catch- блок у функцію.

1. **До чого призведе передавання винятку в блок *catch* за посиланням?**

При передачі об’єкту- винятку за посиланням catch буде посилатися на той самий об’єкт- виняток, а не створювати новий.

1. **Для чого використовують вираз throw, що не містить операндів?**

Щоб створити повторно оброблюваний в даний момент виняток.

1. **До чого призведе генерація похідної виняткової ситуації за умови наявності блоку *catch*, що призначений для обробки базової виняткової ситуації? Відповідь пояснити.**

Якщо catch- блок, що оброблює виняток базового типу, стоїть перед catch- блоком, що оброблює виняткок похідного типу, то завжди при генерації випадкової ситуації буде виконуватись catch- блок з винятком базового типу. Якщо спочатку стоїть catch- блок з винятком похідного типу, а за ним catch- блок з винятком базового типу, то при генерації виняткової ситуації похідного типу буде виконуватись catch- блок з винятком похідного типу, а при генерації випадкової ситуації базового типу буде виконуватись catch- блок з базовим типом винятку.

1. **Назвіть і опишіть стандартні типи винятків з бібліотеки *std*.**

Базовий клас exeption: bad\_exception, bad\_alloc, bad\_typeid, bad\_cast.

Базовий клас logic\_error: domain\_error, invalid\_argument, length\_error, out\_of\_range.

Базовий клас runtime\_error: overflow\_error, range\_error, underflow\_error.

1. **Для чого під час обробки виняткових ситуацій використовують призначені функції *terminate()*, *unexpected()*, *abort()*?**

terminate() виконується, якщо не знайдений потрібний для обробки блок catch. unexpected() – якщо функція створює виключення, якого немає в throw-списку.

abort() – вона викликається функцією terminate().

1. **Навести приклад коду, за допомогою якого можна вивести на екран текстове повідомлення з номером строчки, під час виконання якої сталася виняткова ситуація, та назвою файлу, що містить цю строчку.**

try

{

//some code

}

catch(Exception& ex)

{

cout<<ex.Message()<<endl;

}