**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

**Факультет комп'ютерних наук**

**Кафедра інженерії програмного забезпечення**

**ЗВІТ**

з лабораторної роботи № 9

**«Виключення»**

**Варіант 5**

Дисципліна «Об'єктно-орієнтоване програмування»

Спеціальність: **Інженерія програмного забезпечення**

121-ЛР.09-208.22120802

***Студент\_\_Д. М.  Дзундза***

*(підпис)*

*\_\_11.01.2021* \_\_

*(дата)*

***Викладач \_\_\_\_\_\_\_\_С. Ю. Боровльова***

*(підпис)*

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

*(дата)*

Миколаїв – 2021

**Лабораторна робота №9**

**Завдання:**

1. Приклади виникнення та перехоплення exception

2. Використання сatch без параметрів

3. Повторну генерація exception в блоці catch та спосіб розкрутки стеку(отримати початкову точку виникнення виключення після повторної генерації виключення)

4. Навести приклади коду з використанням блоку finally. Пояснити алгоритм виконання цього блоку в різних режимах.

5. Створіть свій клас для генерації exception та використайте його в коді.

6. Наведіть в коді приклади генерації наступних виключень (використовувати throw заборонено, виключення повинно генеруватись CLR автоматично):

6.1. NullReferenceException

6.2. ArgumentNullException

6.3. InvalidOperationException

**Лістинг коду:**

using System;

using System.Collections.Generic;

namespace ConsoleApp1

{

class MyException : Exception

{

int errorCode;

public MyException(string mess, int errorCode) : base(mess)

{

this.errorCode = errorCode;

}

public int ErrorCode() { return errorCode; }

}

class Program

{

static void Main(string[] args)

{

int first = 5, second = 0;

try

{

try

{

//int zeroDivision = first / second;

//object o = null;

//o.ToString();

//throw new MyException("my error", 22);

String[] s = null;

String sep = " ";

String j = String.Join(sep, s);

}

catch (DivideByZeroException ex)

{

Console.WriteLine("Division by 0 occured\n" + ex.StackTrace);

throw new DivideByZeroException("Message", ex);

}

catch (ArgumentNullException ex) {

Console.WriteLine("Argument Null Exception Occured");

}

catch (NullReferenceException ex)

{

Console.WriteLine("Null Reference Exception Occured");

}

catch (Exception ex)

{

Console.WriteLine("Base catch worked: " + ex.Message);

}

finally

{

var date1 = new DateTime(2008, 5, 1, 8, 30, 52);

Console.WriteLine(date1);

}

var numbers = new List<int>() { 1, 2, 3, 4, 5 };

foreach (var number in numbers)

{

int square = (int)Math.Pow(number, 2);

Console.WriteLine("{0}^{1}", number, square);

Console.WriteLine("Adding {0} to the collection...\n", square);

numbers.Add(square);

}

}

catch (DivideByZeroException ex)

{

Console.WriteLine("Second division by 0 occured\n" + ex.InnerException);

}

catch (InvalidOperationException ex)

{

Console.WriteLine("Invalid Operation Exception Occured");

}

catch {

Console.WriteLine("Exception upper level");

}

}

}

}

**Питання до захисту:**

1. Базовий клас для всіх виключень

Exception

1. Специфіка використання сatch без параметрів

Використовувати його в останню чергу після всіх похідних виключень.

1. Розкрутка стеку

Використовувати StackTrace та InnerException.

1. Блок finally

Виконується завжди, окрім випадку коли програма аварійно завершує роботу.

1. Вкладені блоки try

Try{

…code…

Try{

…code…

Catch{}

}

Catch{}