**мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»**

**ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра ІСМ**

|  |
| --- |
| **2022** |



**ЗВІТ**

**Про виконання лабораторної роботи №5.1(C)**

**«Класи з опрацюванням виняткових ситуацій»**

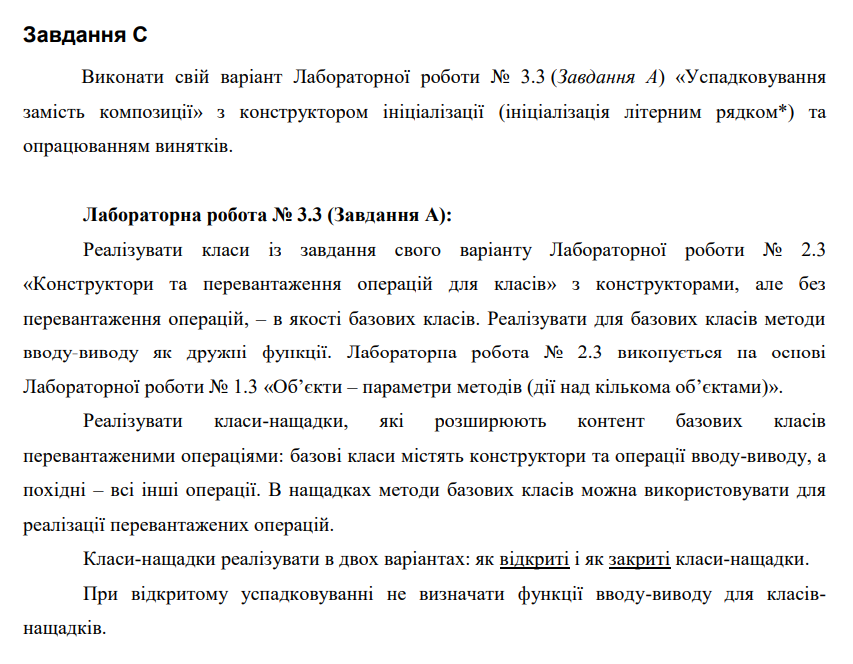
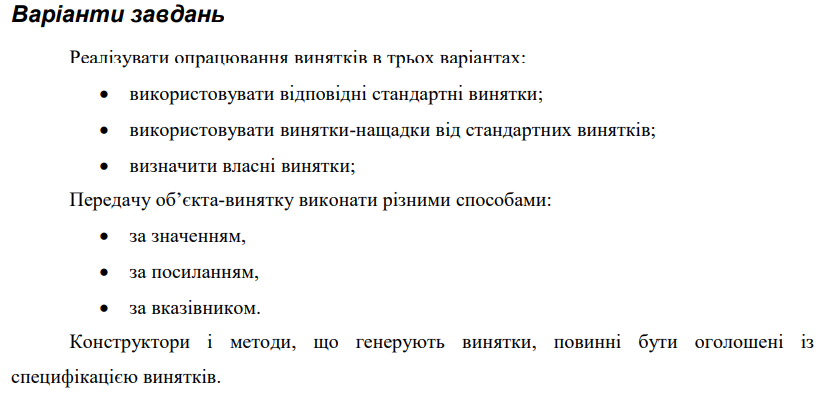
**З дисципліни**

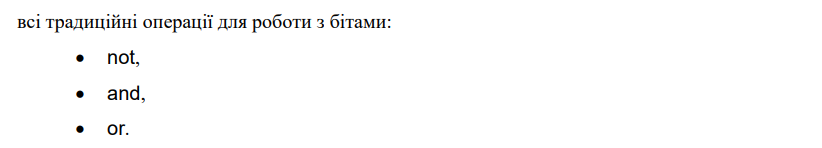
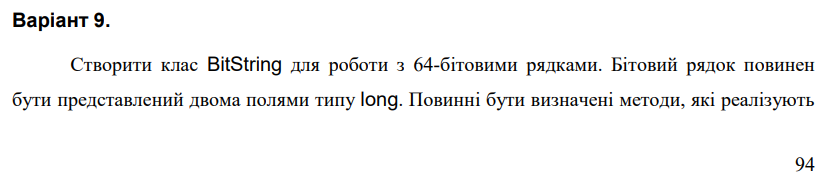
**«Об’єктно-орієнтоване програмування»**

**Студента групи ІТ-11**

**Лисецького Володимира Любомировича**

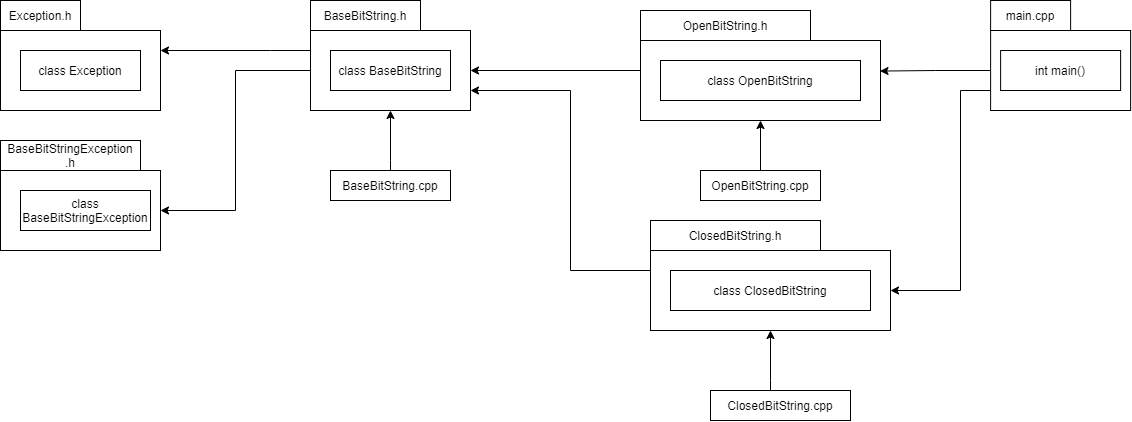
**Умова завдання:**

****

****

**UML-діаграма класів: **

**Структурна схема програми:**



**Текст програми:**

#pragma once

#include "BaseBitString.h"

class OpenBitString :

public BaseBitString

{

public:

OpenBitString operator ++();

OpenBitString operator --();

OpenBitString operator ++(int);

OpenBitString operator --(int);

OpenBitString& operator = (const OpenBitString& obj);

friend OpenBitString operator ! (OpenBitString obj);

friend OpenBitString operator | (OpenBitString obj1, OpenBitString obj2);

friend OpenBitString operator & (OpenBitString obj1, OpenBitString obj2);

};

#pragma once

#include "BaseBitString.h"

class ClosedBitString :

private BaseBitString

{

public:

ClosedBitString();

ClosedBitString(long x, long y);

ClosedBitString(char\* x, char\* y);

ClosedBitString(ClosedBitString& obj);

ClosedBitString operator ++();

ClosedBitString operator --();

ClosedBitString operator ++(int);

ClosedBitString operator --(int);

ClosedBitString& operator = (const ClosedBitString& obj);

operator string() const;

friend ostream& operator << (ostream& out, ClosedBitString& obj);

friend istream& operator >> (istream& in, ClosedBitString& obj);

friend ClosedBitString operator ! (ClosedBitString obj);

friend ClosedBitString operator | (ClosedBitString obj1, ClosedBitString obj2);

friend ClosedBitString operator & (ClosedBitString obj1, ClosedBitString obj2);

};

#pragma once

#include<iostream>

#include<string>

#include<exception>

using namespace std;

class Exception : public exception

{

string message;

public:

Exception(string message) : message(message) {}

string What() { return message; }

};

#pragma once

#include <sstream>

#include <string>

#include <iostream>

#include "Exception.h"

#include "BaseBitStringException.h"

using namespace std;

class BaseBitString

{

protected:

long a;

long b;

public:

long getA() const { return a; };

long getB() const { return b; };

void setA(long value);

void setB(long value);

operator string() const;

BaseBitString& operator = (const BaseBitString& obj);

BaseBitString();

BaseBitString(long first, long second);

BaseBitString(char\* first, char\* second);

BaseBitString(BaseBitString& obj);

friend long to\_Long(char\* value);

friend ostream& operator << (ostream& out, BaseBitString& obj);

friend istream& operator >> (istream& in, BaseBitString& obj);

};

// Lab5\_1C.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

#include "OpenBitString.h"

#include "ClosedBitString.h"

using namespace std;

int main()

{

OpenBitString b1;

try {

cin >> b1;

cout << endl;

}

catch (BaseBitStringException\* q) {

cout << "BaseBitStringException: " << q->What() << endl;

}

catch (BaseBitStringException& q) {

cout << "BaseBitStringException: " << q.What() << endl;

}

catch (Exception& q) {

cout << "Exception: " << q.What() << endl;

}

catch (Exception\* q) {

cout << "Exception: " << q->What() << endl;

}

catch (invalid\_argument q) {

cout << "invalid\_argument: " << q.what() << endl;

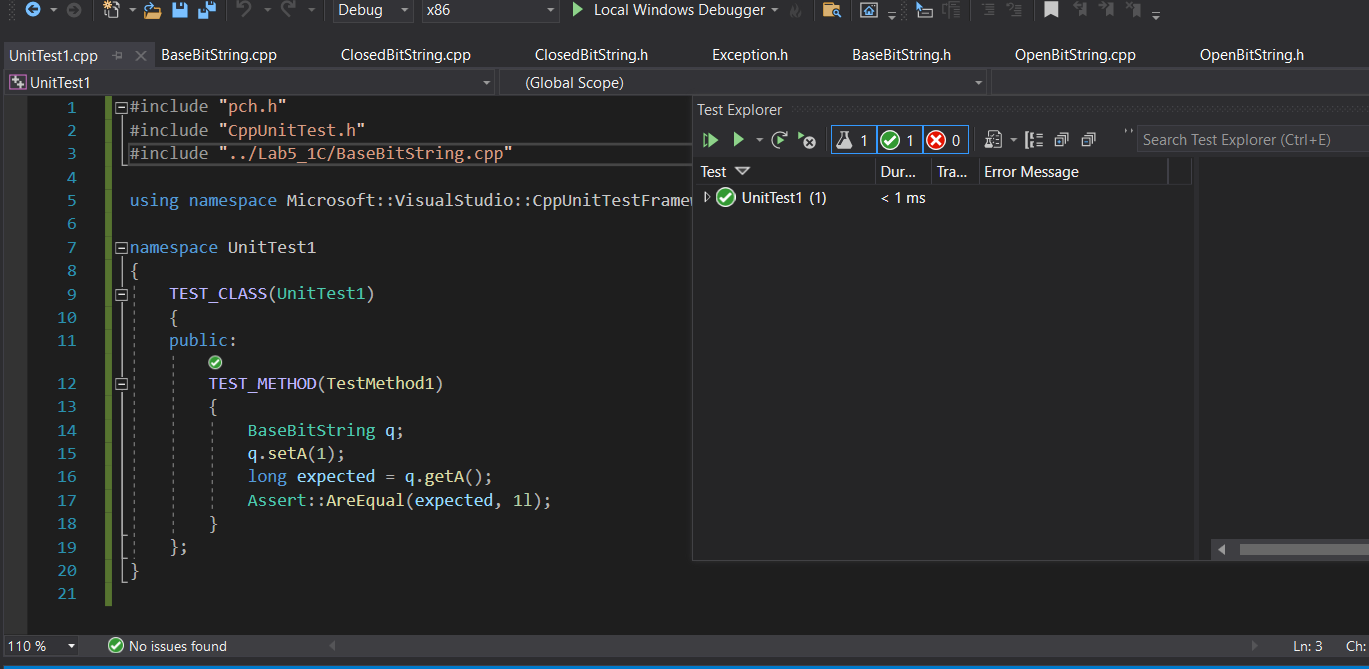
}

cout << b1 << endl;

}

**Посилання на git-репозиторій з проектом:**

**https://github.com/MarshmallowSoup/oop-lab-5-1**

**Результати unit-тесту:** ****