ЗВІТ

про виконання лабораторної роботи №1

«Логіка»

з дисципліни «Програмування дискретних структур»

Студента групи КН-2327Б

Сухар Станіслав

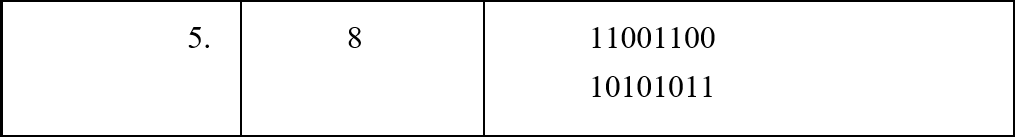
Мета роботи: Поглибити і закріпити розуміння теоретичних положень логіки висловлювань, формувати навички складання алгоритмів та програм опрацювання логічних величин

## Завдання: Скласти програми із зазначеними вхідними даними та результатами.

1. Задано значення істинності висловлювань *p* та *q*. Знайти значення істинності кон’юнкції, диз’юнкції, альтернативного «або», імплікації й еквівалентності цих висловлювань.

2. Задано два бітові рядки довжиною *n*. Знайти результати виконання порозрядних операцій OR, AND, XOR цих рядків.

Варіант:



1. Текст програми:

main.cpp

#include <iostream>

#include <string>

#include <limits>

using namespace std;

bool implication(bool p, bool q) {

return !p || q;

}

bool equivalence(bool p, bool q) {

return implication(p, q) && implication(q, p);

}

void bitwise\_operations(const string& a, const string& b, string& or\_result, string& and\_result, string& xor\_result) {

if (a.length() != 8 || b.length() != 8) {

throw invalid\_argument("Both strings must be exactly 8 characters long.");

}

or\_result = "";

and\_result = "";

xor\_result = "";

for (int i = 0; i < 8; ++i) {

if (a[i] == '1' || b[i] == '1') {

or\_result += '1';

}

else {

or\_result += '0';

}

if (a[i] == '1' && b[i] == '1') {

and\_result += '1';

}

else {

and\_result += '0';

}

if (a[i] != b[i]) {

xor\_result += '1';

}

else {

xor\_result += '0';

}

}

}

bool get\_input(const string& prompt) {

bool value;

while (true) {

cout << prompt;

if (cin >> value && (value == 0 || value == 1)) {

break;

}

else {

cout << "Помилка: введiть 0 або 1!" << endl;

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

}

}

return value;

}

string get\_bit\_string(const string& prompt) {

string bit\_string;

while (true) {

cout << prompt;

cin >> bit\_string;

bool is\_valid = true;

for (char c : bit\_string) {

if (c != '0' && c != '1') {

is\_valid = false;

break;

}

}

if (is\_valid && bit\_string.length() == 8) {

break;

}

else {

cout << "Помилка: рядок повинен мiстити тiльки 0 та 1, i мати рiвно 8 символiв." << endl;

}

}

return bit\_string;

}

int main() {

setlocale(LC\_CTYPE, "ukr");

bool p = get\_input("Введiть значення iстинностi для p (0 або 1): ");

bool q = get\_input("Введiть значення iстинностi для q (0 або 1): ");

bool conjunction = p && q;

cout << "Кон'юнкцiя (p i q): " << conjunction << endl;

bool disjunction = p || q;

cout << "Диз'юнкцiя (p або q): " << disjunction << endl;

bool exclusiveOr = p != q;

cout << "Альтернативне \"або\" (p xor q): " << exclusiveOr << endl;

bool impl\_pq = implication(p, q);

cout << "IмплiкацIя (p -> q): " << impl\_pq << endl;

bool impl\_qp = implication(q, p);

cout << "Iмплiкацiя (q -> p): " << impl\_qp << endl;

bool equiv = equivalence(p, q);

cout << "Еквiвалентнiсть (p ~ q): " << equiv << endl;

string a = get\_bit\_string("\nВведiть 1-й бiтовий рядок (8 бIт): ");

string b = get\_bit\_string("Введiть 2-й бiтовий рядок (8 бIт): ");

string or\_result, and\_result, xor\_result;

bitwise\_operations(a, b, or\_result, and\_result, xor\_result);

cout << "Результат OR: " << or\_result << endl;

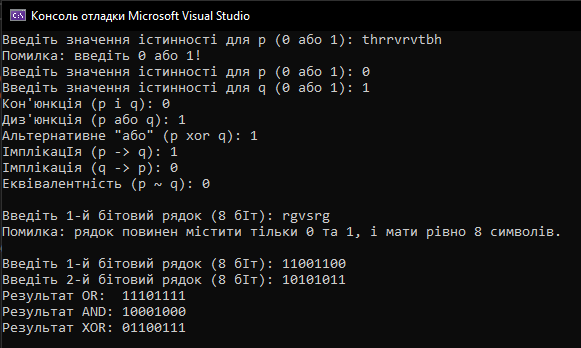
cout << "Результат AND: " << and\_result << endl;

cout << "Результат XOR: " << xor\_result << endl;

return 0;

}

1. Результат виконання коду:



1. UnitTest

#include "pch.h"

#include "CppUnitTest.h"

#include "../lab\_1\_PDS/main.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace unit1

{

TEST\_CLASS(unit1)

{

public:

TEST\_METHOD(TestImplication)

{

Assert::IsFalse(implication(1, 0));

Assert::IsTrue(implication(1, 1));

}

TEST\_METHOD(TestEquivalence)

{

Assert::IsTrue(equivalence(0, 0));

Assert::IsFalse(equivalence(0, 1));

Assert::IsFalse(equivalence(1, 0));

Assert::IsTrue(equivalence(1, 1));

}

TEST\_METHOD(TestBitwiseOperations) {

std::string a = "11001100";

std::string b = "10101011";

std::string or\_result, and\_result, xor\_result;

bitwise\_operations(a, b, or\_result, and\_result, xor\_result);

Assert::AreEqual(std::string("11101111"), or\_result);

Assert::AreEqual(std::string("10001000"), and\_result);

Assert::AreEqual(std::string("01100111"), xor\_result);

}

TEST\_METHOD(TestBitwiseOperations\_AllOnesAndZeros) {

std::string a = "11111111";

std::string b = "00000000";

std::string or\_result, and\_result, xor\_result;

bitwise\_operations(a, b, or\_result, and\_result, xor\_result);

Assert::AreEqual(std::string("11111111"), or\_result);

Assert::AreEqual(std::string("00000000"), and\_result);

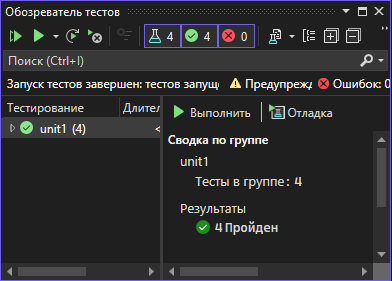
Assert::AreEqual(std::string("11111111"), xor\_result);

}

};

}

1. Результат UnitTest:



1. Побудова cтруктурної діаграми:

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Посилання на Git-репозиторій:

Висновок:На даній лабораторній роботі, я поглибив і закріпив розуміння теоретичних положень логіки висловлювань, сформував навички складання алгоритмів та програм опрацювання логічних величин.