ЗВІТ

про виконання лабораторної роботи № < 5.2 >

*«Обчислення суми ряду Тейлора за допомогою функцій »*

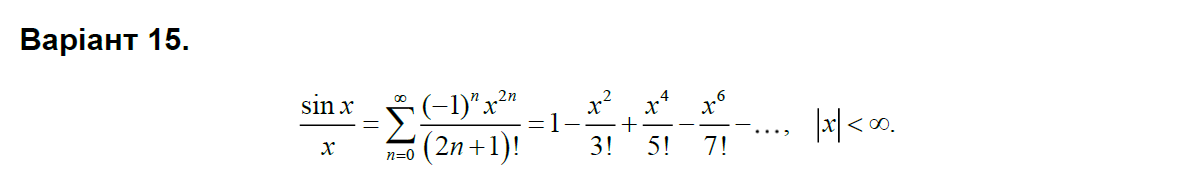
з дисципліни

«Алгоритмізація та програмування»

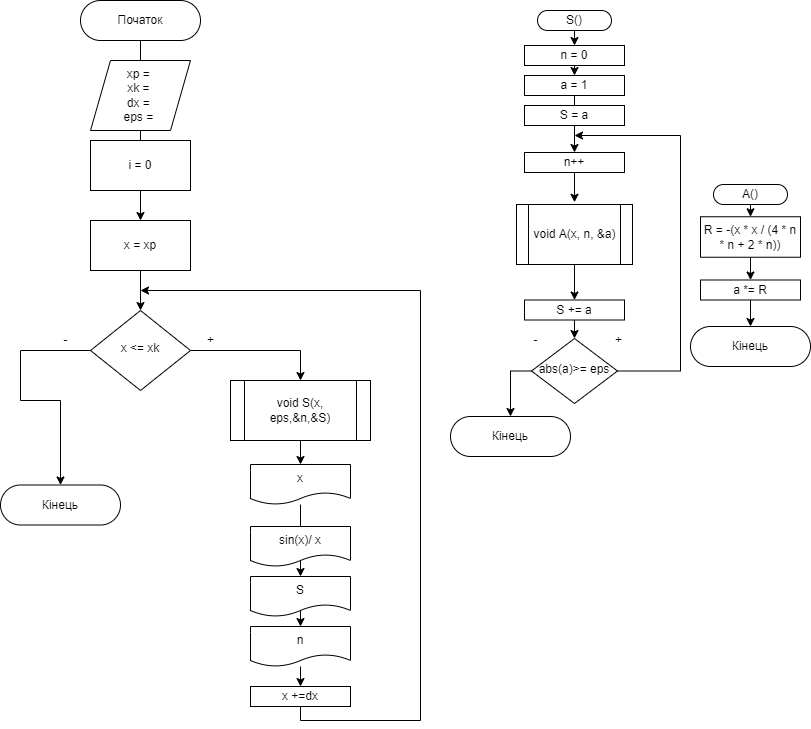
студента групи ІК-11

< *Снігура Стефана Андрійовича* >

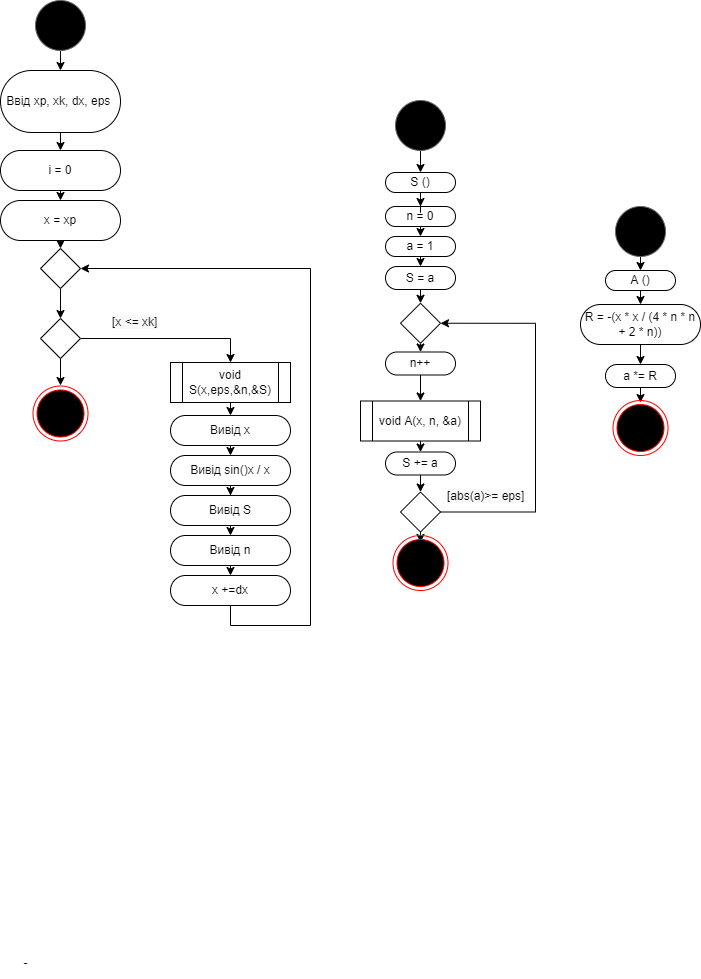
**Умова завдання:**



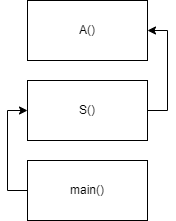
**Блок-схема:**

****

**UML-activity діаграма :**

****

**Структурна схема програми :**

****

**Текст програми:**

**VOID-спосіб**

// Лабораторна робота № 5.2

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

void S(const double x, const double eps, int& n, double& s);

void A(const double x, const int n, double& a);

int main()

{

double xp, xk, x, dx, eps, s = 0;

int n = 0;

cout << "xp = "; cin >> xp;

cout << "xk = "; cin >> xk;

cout << "dx = "; cin >> dx;

cout << "eps = "; cin >> eps;

cout << fixed;

cout << "-------------------------------------------------" << endl;

cout << "|" << setw(5) << "x" << " |"

<< setw(10) << "exp(x)" << " |"

<< setw(7) << "S" << " |"

<< setw(5) << "n" << " |"

<< endl;

cout << "-------------------------------------------------" << endl;

x = xp;

while (x <= xk)

{

S(x, eps, n, s);

cout << "|" << setw(7) << setprecision(2) << x << " |"

<< setw(10) << setprecision(5) << sin(x \* 1.0)/x << " |"

<< setw(10) << setprecision(5) << s << " |"

<< setw(5) << n << " |"

<< endl;

x += dx;

}

cout << "-------------------------------------------------" << endl;

return 0;

}

void S(const double x, const double eps, int& n, double& s)

{

n = 0; // вираз залежить від умови завдання варіанту

double a = 1; // вираз залежить від умови завдання варіанту

s = a;

do {

n++;

A(x, n, a);

s += a;

} while (abs(a) >= eps);

}

void A(const double x, const int n, double& a)

{

double R = -(x \* x / (4 \* n \* n + 2 \* n)); // вираз залежить від умови завдання варіанту

a \*= R;

}

**DOUBLE-спосіб**

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

double S(const double x, const double eps, int& n, double s);

double A(const double x, const int n, double a);

int main()

{

double xp, xk, x, dx, eps, s = 0;

int n = 0;

cout << "xp = "; cin >> xp;

cout << "xk = "; cin >> xk;

cout << "dx = "; cin >> dx;

cout << "eps = "; cin >> eps;

cout << fixed;

cout << "-------------------------------------------------" << endl;

cout << "|" << setw(5) << "x" << " |"

<< setw(10) << "sin(x)/x" << " |"

<< setw(7) << "S" << " |"

<< setw(5) << "n" << " |"

<< endl;

cout << "-------------------------------------------------" << endl;

x = xp;

while (x <= xk)

{

s = S(x, eps, n, s);

cout << "|" << setw(7) << setprecision(2) << x << " |"

<< setw(10) << setprecision(5) << sin(x) / x << " |"

<< setw(10) << setprecision(5) << s << " |"

<< setw(5) << n << " |"

<< endl;

x += dx;

}

cout << "-------------------------------------------------" << endl;

return 0;

}

double S(const double x, const double eps, int& n, double s)

{

n = 0; // вираз залежить від умови завдання варіанту

double a = 1; // вираз залежить від умови завдання варіанту

s = a;

do {

n++;

a = A(x, n, a);

s += a;

} while (abs(a) >= eps);

return s;

}

double A(const double x, const int n, double a)

{

double R = -(x \* x / (4 \* n \* n + 2 \* n)); // вираз залежить від умови завдання варіанту

a \*= R;

return a;

}

**UNIT-тест(void)**

#include "pch.h"

#include "CppUnitTest.h"

#include "/Users/User/source/repos/lab5.2/Source.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest1

{

TEST\_CLASS(UnitTest1)

{

public:

TEST\_METHOD(TestMethod\_A)

{

double a = 1;

A(6, 1, a);

Assert::AreEqual(a, -6.);

}

TEST\_METHOD(TestMethod\_S)

{

int n = 2;

double s = 1;

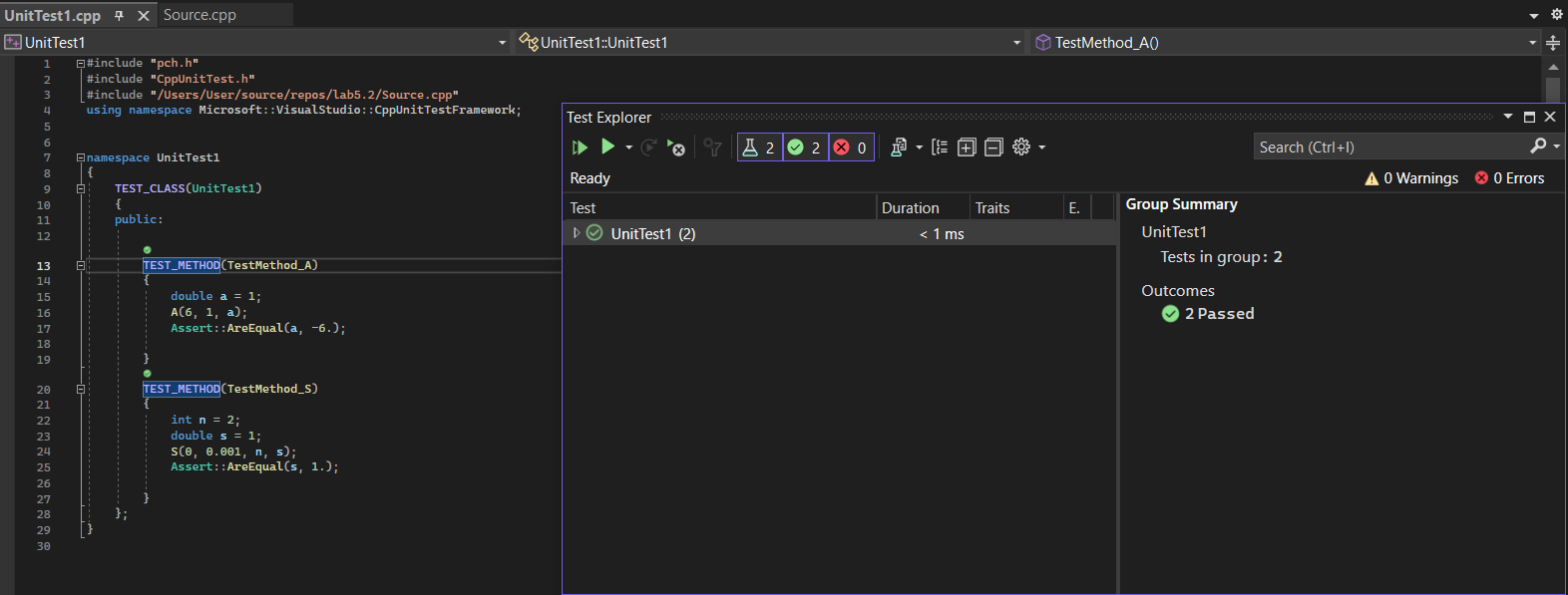
S(0, 0.001, n, s);

Assert::AreEqual(s, 1.);

}

};

}

****

**UNIT-тест(double)**

#include "pch.h"

#include "CppUnitTest.h"

#include "C:\Users\User\source\repos\lab5.2(return)\Source.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest1

{

TEST\_CLASS(UnitTest1)

{

public:

TEST\_METHOD(TestMethod\_A)

{

int test\_n = 1;

double test\_A = A(6, test\_n, 1);

Assert::AreEqual(test\_A, -6.);

}

TEST\_METHOD(TestMethod\_S)

{

int test\_n2 = 1;

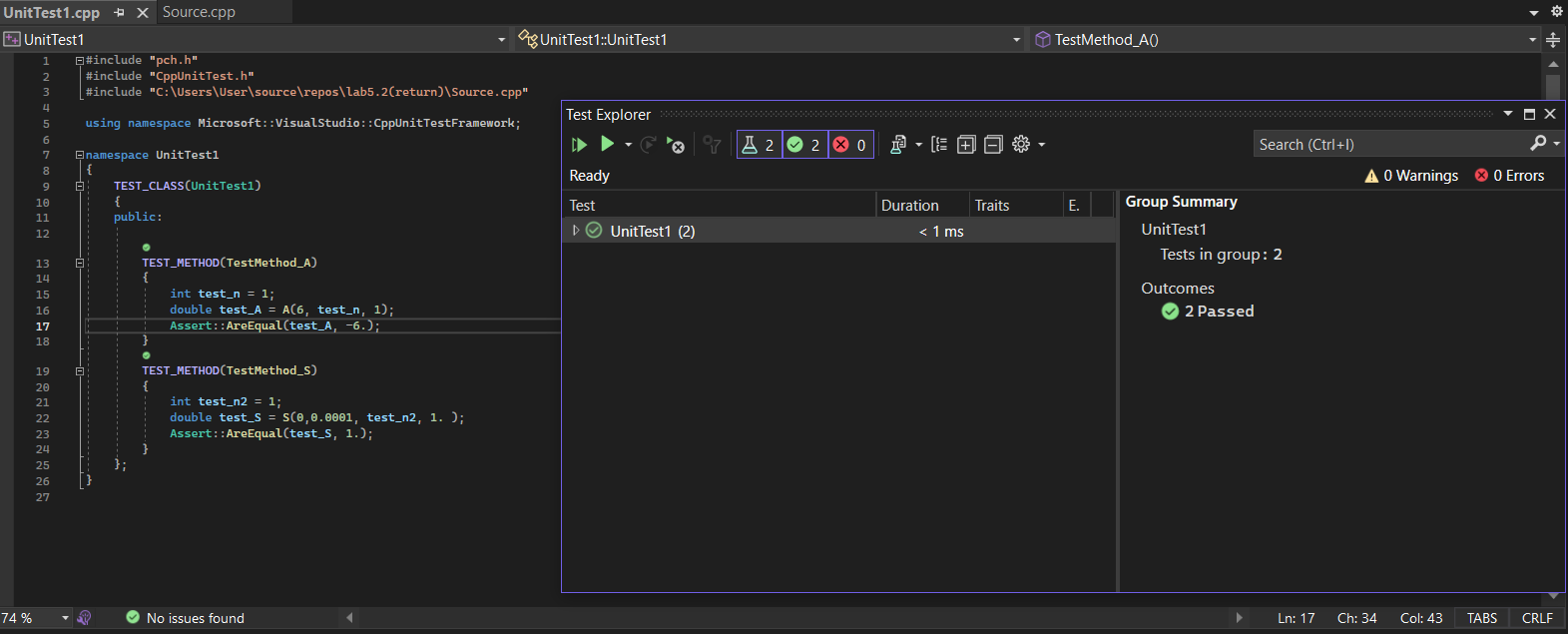
double test\_S = S(0,0.0001, test\_n2, 1. );

Assert::AreEqual(test\_S, 1.);

}

};

}

****

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

<https://github.com/BigTrouble-Git/ashtray.git>

**Висновки**:

Виконавши цю лабораторну роботу я навчився використовувати функції, формати виводу та рекурентні співвідношення.