ЗВІТ

про виконання лабораторної роботи № < 5.4 >

*«Обчислення сум та добутків за допомогою рекурсії »*

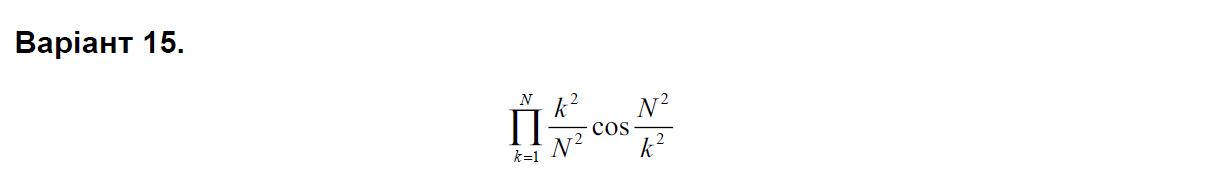
з дисципліни

«Алгоритмізація та програмування»

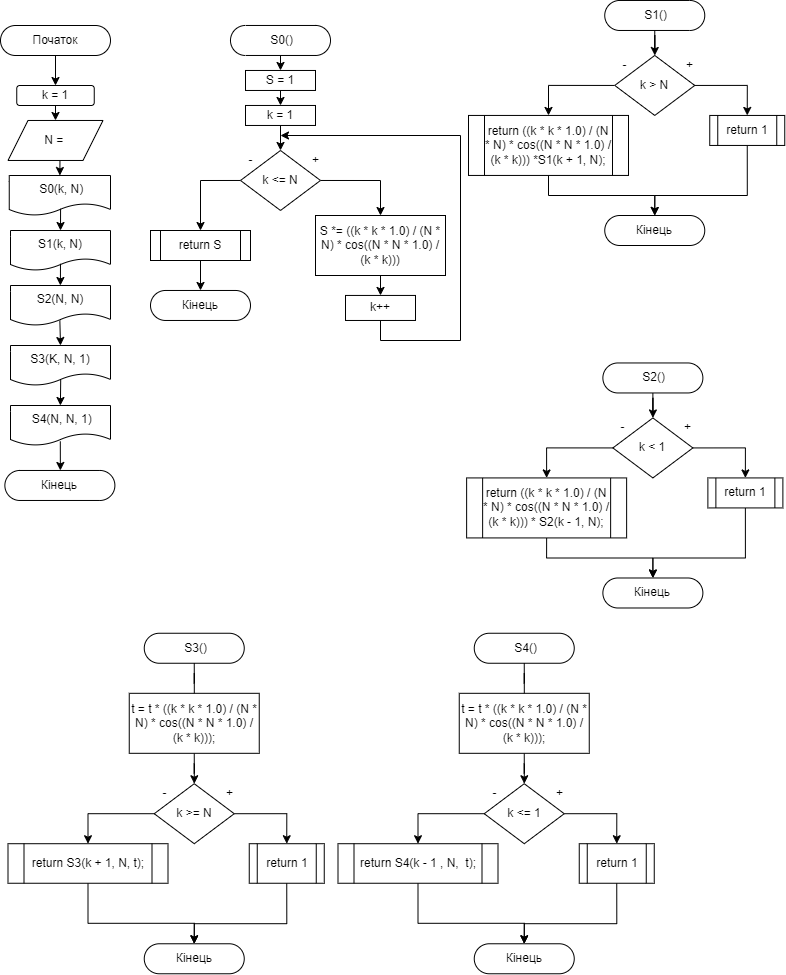
студента групи ІК-11

< *Снігура Стефана Андрійовича* >

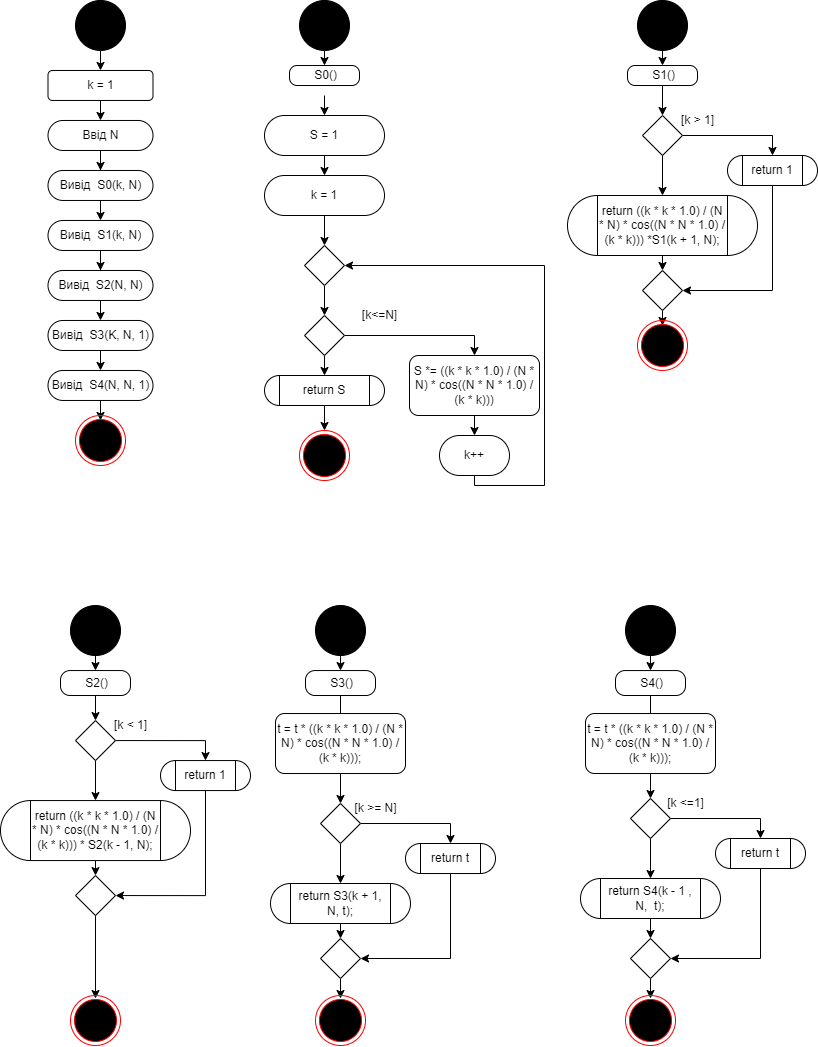
**Умова завдання:**

****

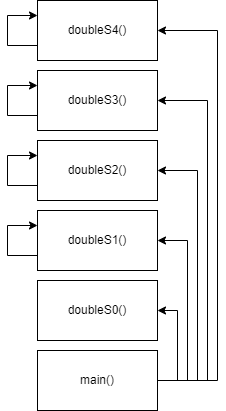
**Блок-схема:**

****

**UML-activity діаграма :**

****

**Структурна схема програми :**

****

**Текст програми:**

#include <iostream>

#include <cmath>

#include <stdio.h>

#include <math.h>

using namespace std;

double S0(const int k, const int N)

{

double S = 1;

for (int k = 1; k<= N; k++)

S \*= ((k \* k \* 1.0) / (N \* N) \* cos((N \* N \* 1.0) / (k \* k)));

return S;

}

double S1(const int k, const int N)

{

if (k > N)

return 1;

else

return ((k \* k \* 1.0) / (N \* N) \* cos((N \* N \* 1.0) / (k \* k))) \*S1(k + 1, N);

}

double S2(const int k, const int N)

{

if (k < 1)

return 1;

else

return ((k \* k \* 1.0) / (N \* N) \* cos((N \* N \* 1.0) / (k \* k))) \* S2(k - 1, N);

}

double S3(const int k, const int N, double t)

{

t = t \* ((k \* k \* 1.0) / (N \* N) \* cos((N \* N \* 1.0) / (k \* k)));

if (k >= N)

return t;

else

return S3(k + 1, N, t);

}

double S4(const int k, const int N, double t)

{

t = t \* ((k \* k \* 1.0) / (N \* N) \* cos((N \* N \* 1.0) / (k \* k)));

if (k <=1)

return t;

else

return S4(k - 1 , N, t);

}

int main()

{

int k = 1, N;

cout << "N = "; cin >> N;

cout << "(iter) S0 = " << S0( k, N) << endl;

cout << "(rec up ++) S1 = " << S1(k, N) << endl;

cout << "(rec up --) S2 = " << S2(N, N) << endl;

cout << "(rec down ++) S3 = " << S3(k, N, 1) << endl;

cout << "(rec down --) S4 = " << S4(N, N, 1) << endl;

return 0;

}

**UNIT-test:**

#include "pch.h"

#include "CppUnitTest.h"

#include "C:\Users\User\source\repos\lab5.4\Source.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest1

{

TEST\_CLASS(UnitTest1)

{

public:

TEST\_METHOD(TestMethodS0)

{

double test\_S0 = S0(1, 2);

Assert::AreEqual(test\_S0, 0,540302);

}

TEST\_METHOD(TestMethodS1)

{

double test\_S0 = S1(1, 2);

Assert::AreEqual(test\_S0, 0, 540302);

}

};

}

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

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

**Висновки**:

Виконавши цю лабораторну роботу я навчився використовувати рекурсивні функції.