**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

**Лабораторна робота №2**

з дисципліни  
«Алгоритми і структури даних»

Виконав: Перевірила:

студент групи ІП-04 Сергієнко А. А.  
Пащенко Дмитро Олексійович  
номер у списку групи: 19

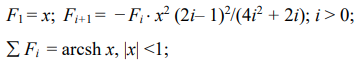
Київ 2021

**Постановка задачі**

Дане натуральне число n. Знайти суму перших n членів ряду чисел, заданого рекурентною формулою. Розв’язати задачу трьома способами (написати три програми): 1) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному спуску; 2) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному поверненні; 3) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

Програми повинні працювати коректно для довільного натурального n включно з n = 1.

**Варіант 19**



**Текст програми 1 (спуск)**

#include <stdio.h>

#include <math.h>

int N = 1;

double first(double x, double F, int n, double s) {

double i = -1;

if (n == -1)

i = s;

else {

s = s + F;

double t = (2 \* (N - n) - 1), p = (4 \* (N - n) \* (N - n) + 2 \* (N - n)), k = -1 \* t \* t / p;

F = x \* k \* x \* F;

i = first(x, F, n - 1, s);

}

return i;

}

int main() {

double S = 0, F, x;

N = 5;

x = 0.2;

if (fabs(x) < 1) {

F = x;

S = first(x, F, N - 1, S);

printf("%.10f\n", S);

}

else {

printf("Module x must be less than 1\n");

}

}

**Тестування**



**Текст програми 2 (повернення)**

#include <stdio.h>

#include <math.h>

int N = 1;

double S = 0;

double second(double x, int n) {

double i;

if (n == 1) {

i = x;

S = S + i;

}

else {

double t = (2 \* (n - 1) - 1), p = (4 \* (n - 1) \* (n - 1) + 2 \* (n - 1)), k = -1 \* t \* t / p;

i = x \* k \* x \* second(x, n - 1);

S = S + i;

}

return i;

}

int main() {

double F, x;

S = 0;

N = 5;

x = 0.2;

if (fabs(x) < 1) {

F = x;

x = second(x, N);

printf("%.10f\n", S);

}

else {

printf("Module x must be less than 1\n");

}

}

**Тестування**



**Текст програми 3 (спуск + повернення)**

#include <stdio.h>

#include <math.h>

int N = 1;

double third(double x, double F, int n) {

double i = -1;

if (n == -1)

return x;

else {

double t = (2 \* (N - n) - 1), p = (4 \* (N - n) \* (N - n) + 2 \* (N - n)), k = -1 \* t \* t / p;

F = x \* k \* x \* F;

return F + third(x, F, n - 1);

}

}

int main() {

double S = 0, F, x;

N = 5;

x = 0.2;

if (fabs(x) < 1) {

F = x;

S = third(x, x, N - 1);

printf("%.10f\n", S);

}

else {

printf("Module x must be less than 1\n");

}

}

**Тестування**



**Текст програми 4 (цикл)**

#include <stdio.h>  
#include <math.h>  
  
int main() {  
 int N;  
 double S = 0, x, F;  
 N = 5;  
 x = 0.2;  
  
 if (fabs(x) < 1) {  
 for (unsigned int i = 1; i <= N; i++) {  
 if (i == 1) {  
 F = x;  
 S += F;  
 } else {  
 F \*= (-1) \* x \* x \* (2 \* (i-1) - 1) \* (2 \* (i-1) - 1) / (4 \* (i-1) \* (i-1) + 2 \* (i-1));  
 S += F;  
 }  
 }  
 printf("%.10f\n", S);  
 }  
 else {  
 printf("Module x must be less than 1\n");  
 }  
}

**Тестування**

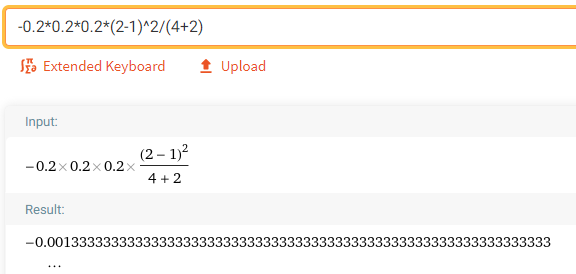


**Тестування (калькулятор)**

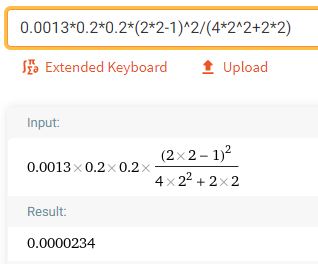
N = 5, x = 0.2

F1 = 0.2

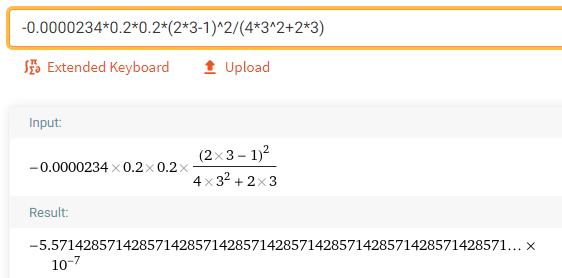
F2 = -0.0013



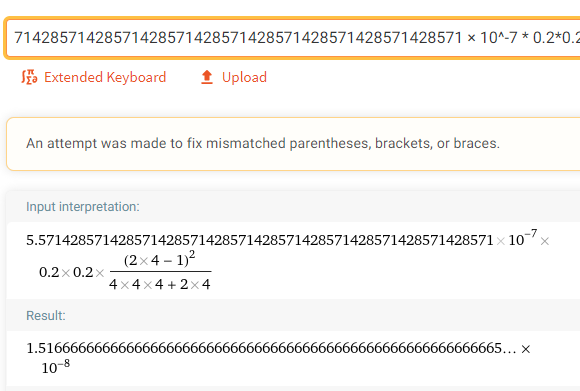
F3 = 0.0000234



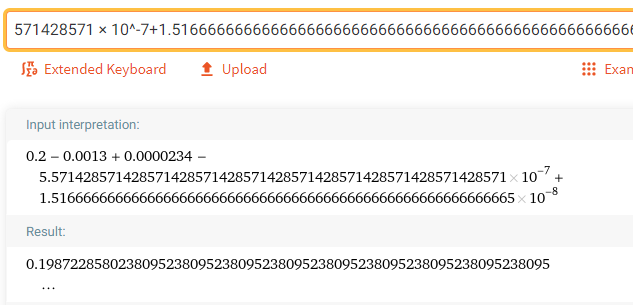
F4 = -5.571428571428571428571428571428571428571428571428571428571 × 10^-7



F5 = 1.5166666666666666666666666666666666666666666666666666666665 × 10^-8



sum = ~ 0.1987228580



**Графік похибки**