МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №9**

**з дисципліни «Основи програмування»**

**на тему: *«*Рекурсія та ітерація.»**

Виконав: студент гр. ПЗ2011

Кулик Сергій Вадимович

Прийняла: ас. Нежуміра О.І.

Дніпро, 2020

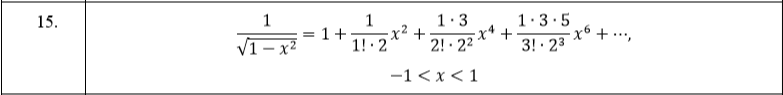
**Тема.** Рекурсія та ітерація.

**Мета.** Навчитися описувати повторюванні обчислювальні процеси у рекурсивної та ітераційної формах.

**Завдання**

Для заданого індивідуального завдання розробити програму обчислення значення функції рекурсивним та ітераційним способом. Порівняти результати обчислення функції рекурсивним та ітераційним способом з результатом обчислення лівої частині виразу, якщо це можливо.

Варіант 15



**1. Зовнішні специфікації**

Вхідні дані: кількість членів ряду, параметр ‘x’ для обчислення ряду.

Вихідні дані: сума ряду.

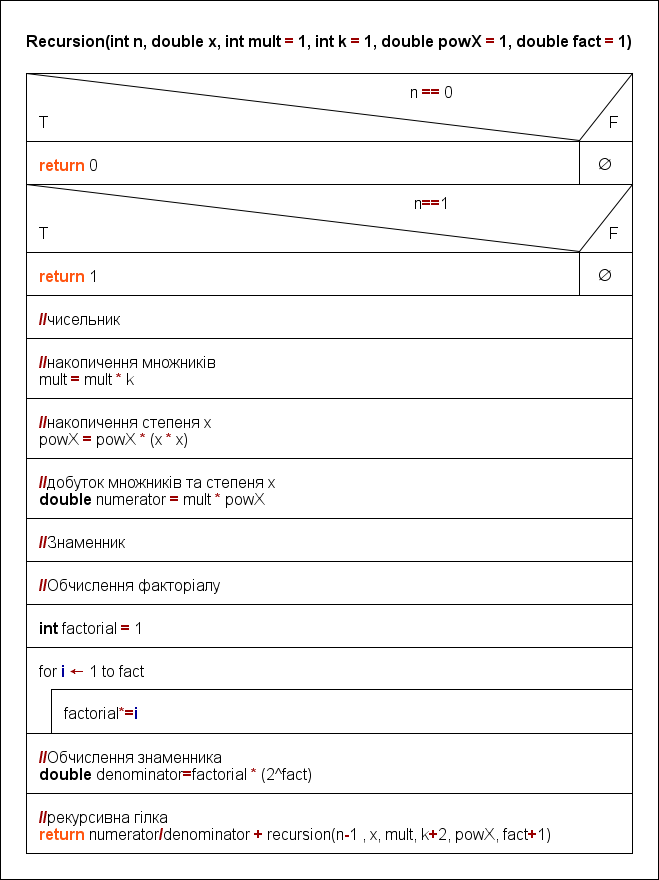
Функціональні вимоги: програма повинна підраховувати суму ряду рекурсивним та ітераційним методом.

**2. Тести**

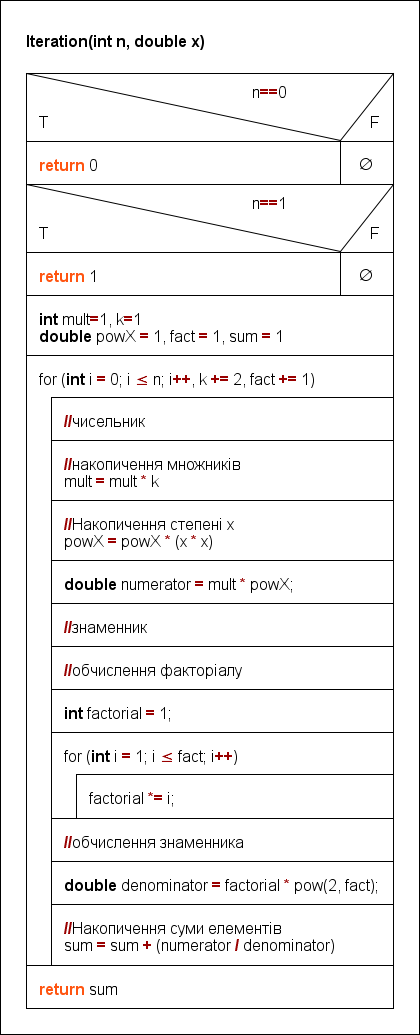
|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва тесту | Вхідні дані | Очікувані результати |
| 1 | Параметри дорівніють нулю | X = 0;  n = 0; | Вираз = 1  Результат рекурсії = 0  Результат ітерції = 0 |
| 2 | Довільні параметри | X = 0,5  n = 10 | Вираз = 1.1547  Результат рекурсії = 1.1547  Результат ітерції = 1.1547 |

**3. Алгоритм**

3.1 Алгоритм для рекурсивного методу



3.2 Алгоритм для ітераційного методу



**4. Текст програми**

//Програма рахує суму ряду рекрсивним та ітераційним методом.

//Результат повинен збігатися з результатом обчислення виразу

//Вхід: кількість елементів ряду

//Вихід: сума ряду

#include <iostream>

#include<math.h>

using namespace std;

/// <summary>

/// Обчислення суми ряду

/// </summary>

/// <param name="n">кількість членів ряду</param>

/// <param name="x">змінна ряду</param>

/// <param name="mult">накопичення множників чисельника</param>

/// <param name="k"></param>

/// <param name="powX">степінь x</param>

/// <param name="fact">змінна для факторіалу</param>

/// <returns>сумма ряду</returns>

double recursion(int n, double x, int mult = 1, int k = 1, double powX = 1, double fact = 1)

{

//термінальна гілка

if (n == 0)

{

return 0;

}

if (n==1)

{

return 1;

}

//чисельник

mult = mult \* k;

powX = powX \* (x \* x);

double numerator = mult \* powX;

//знаменник

//обчислення факторіалу

int factorial = 1;

for (int i = 1; i <= fact; i++)

{

factorial\*= i;

}

//обчислення знаменника

double denominator = factorial \* pow(2, fact);

//рекурсивна гілка

return numerator/denominator + recursion(n-1 , x, mult, k+2, powX, fact+1);

}

/// <summary>

/// Обчислення суми ряду

/// </summary>

/// <param name="n">кількість членів ряду</param>

/// <param name="x">змінна ряду</param>

/// <returns></returns>

double iteration(int n, double x)

{

if (n == 0)

{

return 0;

}

if (n == 1)

{

return 1;

}

int mult=1, k=1;

double powX = 1, fact = 1, sum = 1;

for (int i = 0; i <= n; i++, k += 2, fact += 1)

{

//чисельник

mult = mult \* k;

powX = powX \* (x \* x);

double numerator = mult \* powX;

//знаменник

//обчислення факторіалу

int factorial = 1;

for (int i = 1; i <= fact; i++)

{

factorial \*= i;

}

//обчислення знаменника

double denominator = factorial \* pow(2, fact);

sum = sum + (numerator / denominator);

}

return sum;

}

int main()

{

setlocale(LC\_ALL, "ru");

int n;

double x;

cout << "Введите количество членов ряда: "; cin >> n;

while (n < 0)

{

cout << "Количество не может быть отрицательным, введите еще раз: ";

cin >> n;

}

cout << "Введите значение парметра 'X' для ряда. -1 < X < 1 : "; cin >> x;

while (x >= 1 || x <= -1)

{

cout << "Ошибка X не принимает таких значений! Попробуйте еще раз : ";

cin >> x;

}

cout << "==========================================" << endl;

cout <<"Результат вычисления примера: "<< 1 / sqrt(1 - pow(x, 2)) << endl;;

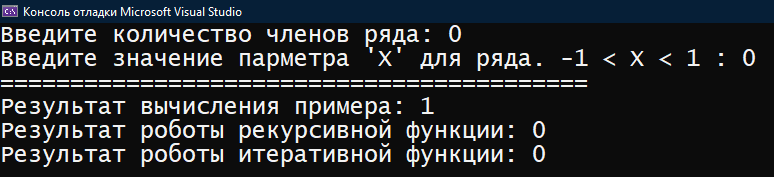
cout <<"Результат роботы рекурсивной функции: "<< recursion(n, x) << endl;;

cout << "Результат роботы итеративной функции: " << iteration(n, x) << endl;

}

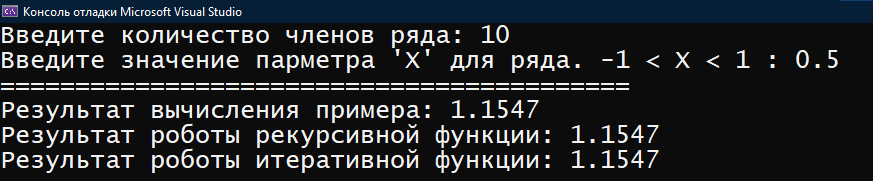
**5. Результати тестуваннь.**

1. Параметри дорівніють нулю



Підрахунок корректний.

2. Довільні параметри



Підрахунок корректний.

**6. Висновок**

Під час виконання лабораторної роботи писав функції, які підраховують суму ряда рекурсивним та ітеративним способом. Для використання рекурсії треба добре обдумати, які параметри буде приймати функція, знайти тривіальне рішення для опису термінальної гілки, та правильно прописати крок рекурсії, інакше може бути перепонення стеку. Ітеративна функція приймає менше параметрів, адже використовує змінні оголошені всередині функції. На мою думку рекурсивні функції треба використовувати в окремих випадках, адже ітеративний метод працює швидше, проте швидкістю можна знехтуати, якщо ваший код буде простіший у розумінні та у подальшій підримці.