МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ЛАБОРАТОРНАЯ РАБОТА №6  
на тему «Рекурсивные функции»

Выполнила: студентка группы РИС-23-2Б Жилина Анастасия Александровна

Проверила: доцент кафедры ИТАС О.А.Полякова

2024**Разработка алгоритма из варианта 18.**

1. **Постановка задачи №1:** Приближенно вычислить значение функции двумя способами: с помощью рекурсии, где x – перменная и n – количество членов ряда, и через прямое вычисление значения функции. Сравнить время выпонения разных способов.

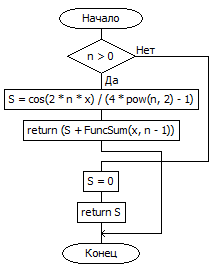
https://sun9-9.userapi.com/impg/IRpfif9wqC7r_HPzCyOirYw45XIHwHMszfwU8w/ZQ4YdwNYTgQ.jpg?size=228x45&quality=96&sign=ae081fae1addbf9e077b48dd50d95eab&type=album

1. **Анализ задачи:**

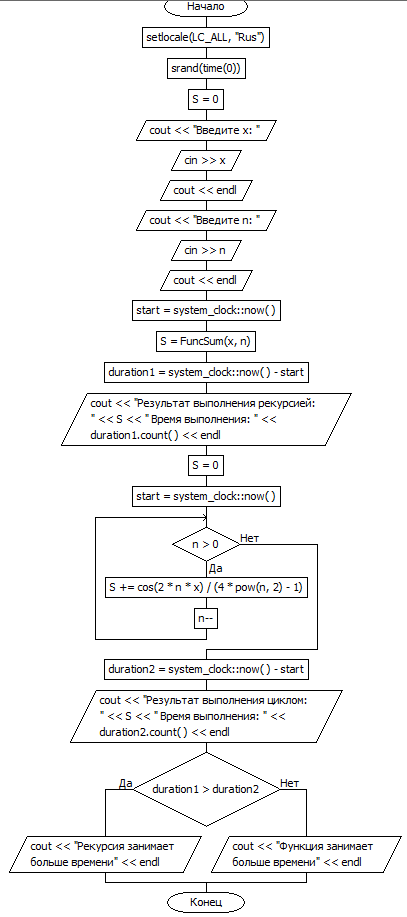
Задача состоит из двух частей: создания рекурсивной функции и цикла. Для создания рекурсивной функции необходимо создать цикл, в котором условием будет n>0. Каждый шаг цикла сопровождается вычислением суммы членов ряда S и вызовом повторно рекурсивной функции с уменьшением значения n на 1. Когда значение n станет равно 0, функция вернет 0.  
Для создания цикла необходимо создать цикл while с условием n > 0, в котором будет производиться подсчет суммы элементов ряда. Если n == 0, то цикл останавливается.

1. **Блок-схема**

Рекурсивная функция:

****

Функция main:

****

1. **Листинг программы**

#include <iostream>

#include <cmath>

#include <chrono>

using namespace std;

using namespace std :: chrono;

double FuncSum(int x, int n)

{

double S;

if (n > 0)

{

S = cos(2 \* n \* x) / (4 \* pow(n, 2) - 1);

return (S + FuncSum(x, n - 1));

}

S = 0;

return S;

}

int main()

{

setlocale(LC\_ALL, "Rus");

double S = 0;

int x, n;

cout << "Введите x: ";

cin >> x;

cout << endl;

cout << "Введите n: ";

cin >> n;

cout << endl;

system\_clock::time\_point start = system\_clock::now();

S = FuncSum(x, n);

duration<double> duration1 = system\_clock::now() - start;

cout << "Результат выполнения рекурсией: " << S << " Время выполнения: " << duration1.count() << endl;

S = 0;

start = system\_clock::now();

while (n > 0)

{

S += cos(2 \* n \* x) / (4 \* pow(n, 2) - 1);

n--;

}

duration<double> duration2 = system\_clock::now() - start;

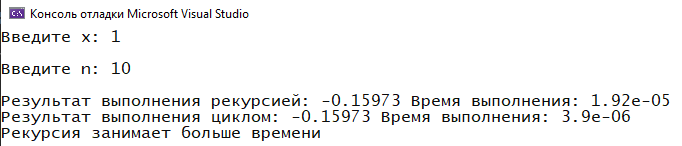
cout<< "Результат выполнения циклом: " << S << " Время выполнения: " << duration2.count() << endl;

if (duration1 > duration2) cout << "Рекурсия занимает больше времени" << endl;

else cout << "Функция занимает больше времени" << endl;

}

1. **Результаты работы**

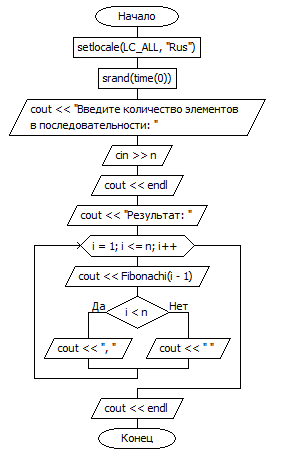
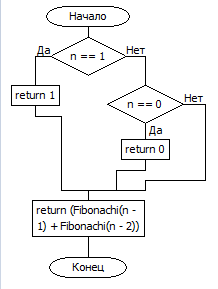
****

1. **Постановка задачи №2:** Реализовать последовательность чисел Фибоначчи с помощью рекурсии.
2. **Анализ задачи:**

Рекурсивная функция может вывести только одно значение – последнее число последовательности. Следовательно создаем цикл от 1 до n, где n – количество элементов в последовательности, который будет выводить каждый раз новый элемент, начиная с 1го. Числа Фибоначчи начинаются с 0, то есть значение первого элемента равно 0. Сделаем в самой рекурсивной функции n индексом элементов, для этого передадим в функцию параметр n-1.

1. **Блок-схема**

int Fibonachi(int n) и int main():

****

1. **Листинг программы**

#include <iostream>

using namespace std;

int Fibonachi(int n)

{

int S;

if (n == 1)

{

return 1;

}

else if (n == 0)

{

return 0;

}

return (Fibonachi(n - 1) + Fibonachi(n - 2));

}

int main()

{

setlocale(LC\_ALL, "Rus");

int n;

cout << "Введите количество элементов в последовательности: ";

cin >> n;

cout << endl;

cout<< "Результат: ";

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

{

cout << Fibonachi(i - 1);

if (i < n) cout << ", ";

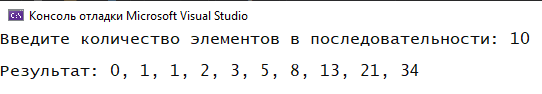
else cout << " ";

}

cout << endl;

}

1. **Результаты работы**

****

1. **Постановка задачи №3:**

Реализовать башню Ханоя с помощью рекурсии.

1. **Анализ задачи**

Анализ задачи для ханойской башни с 3 стержнями и 3 дисками:

1. У нас есть три стержня: 1, 2, 3. На стержне 1 изначально находятся 3 диска, уложенные по убыванию размера.

2. Задача состоит в том, чтобы переместить все диски с стержня 1 на стержень 3, соблюдая условия:

   - За один раз можно переместить только один диск.

   - Никакой диск не может быть положен на диск меньшего размера.

3. Для решения задачи используется рекурсивный подход:

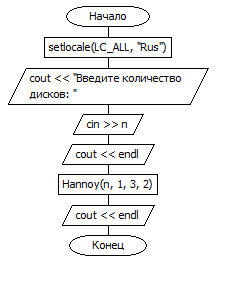
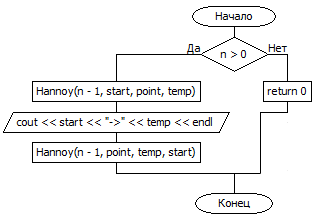
   - Первый шаг - переместить верхние n-1 дисков со стержня 1 на стержень 2, используя стержень 3 в качестве вспомогательного.

   - Второй шаг - переместить самый большой диск с стержня 1 на стержень 3.

   - Третий шаг - переместить n-1 дисков с стержня 2 на стержень 3, используя стержень 1 в качестве вспомогательного.

4. Каждый шаг рекурсивно вызывает функцию Hanoy(), пока не будут перемещены все диски на стержень 3.

1. **Блок-схема**



1. **Листинг программы**

#include <iostream>

using namespace std;

int Hannoy(int n, int start, int temp, int point)

{

if (n > 0)

{

Hannoy(n - 1, start, point, temp);

cout << start << "->" << temp<<endl;

Hannoy(n - 1, point, temp, start);

}

else

{

return 0;

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

int n;

cout << "Введите количество дисков: ";

cin >> n;

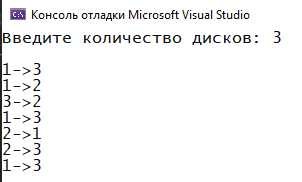
cout << endl;

Hannoy(n,1,3,2);

cout << endl;

}

1. **Результаты работы**



Разместим готовые исполнимые файлы в созданный репозиторий на GitHub

