|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное автономное образовательное учреждение высшего образования  «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ» |

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | Синёв Н. И. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ЦИКЛИЧЕСКИЕ АЛГОРИТМЫЛАБОРАТОРНАЯ РАБОТА №2 |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1941 |  |  |  | Князюк Р.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2020

1. Постановка задачи

Вычислить сумму первых N элементов ряда:

1. Формализация

* **Список** **вводимых переменных:**

num\_elem **–** количество элементов ряда;

x – переменная для вычисления ряда;

* **Список промежуточных переменных:**

factor – факториал;

i – счетчик цикла;

res – результат;

x\_temp - переменная для рекуррентного подсчета переменной x;

* Переменные res, x, x\_temp –типа float , так как эти переменные могут иметь дробную часть;
* Первым элементом ряда будем считать 1;
* num\_elem, factor, i – целочисленный тип, так как не имеют дробной части;
* Данные вводятся с клавиатуры. Вводятся только цифры.
* res=1.0 ,factor=1 и i=0. res=1, потому что первый элемент ряда равен 1. factor=1 ,потому что 0!=1. i=0, для рекуррентной зависимости факториала;
* Программа должна проверять, что num\_elem>0 , x<>0, иначе если num\_elem>0,то вывести результат, иначе вывести ошибку;
* Чтобы не считать факториал каждый раз заново, будем хранить предыдущее значение факториала. Рекуррентная зависимость факториала для этого ряда factor=factor\*(i\*2+1)\*(i\*2+2); (i - переменная для цикла, увеличивающаяся на 1 в конце итерации);
* Результат вывести с точностью до 6 знаков после запятой;
* Из-за сохранения знака перехода каретки на новую строку getchar() не “задерживает” окно консоли, для решения проблемы используем \_flushall();

Тестовые примеры.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пример | №1 | №2 | №3 | №4 |
| Данные | num\_elem=1  x=0 | num\_elem=-1  x=0 | num\_elem=3  x=4 | num\_elem=-1  x=1 |
| Результат | 1.000000 | Error. | 3,666667 | Error. |

1. Алгоритмизация. Схема алгоритма

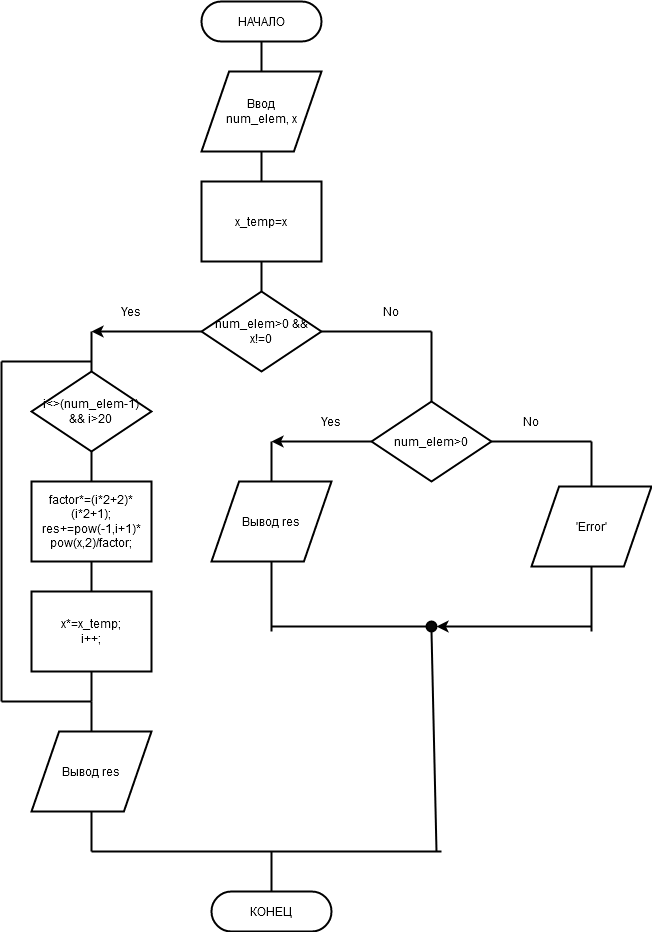


Рисунок 1 – схема алгоритма

Лист 1

1. Код программы на Си

#include <stdio.h>//printf и scanf

#include <conio.h>//для getchar

#include <math.h>//для функции pow()

int main(void){

    int num\_elem,factor=1,i=0;

    float res=1.0,x,x\_temp;

    printf("Nums of elements: ");

    scanf("%d",&num\_elem);

    printf("Enter X: ");

    scanf("%f",&x);

    x\_temp=x;

    if(num\_elem>0 && x!=0){

        while(i!=(num\_elem-1) && i<20){

            factor\*=(i\*2+2)\*(i\*2+1);

            res+=pow(-1,i+1)\*pow(x,2)/factor;

            x\*=x\_temp;

            i++;

        }

        printf("Result: %.6f",res);

    }else if(num\_elem>0)

        printf("Result: %f",res);

    else

        printf("Error.");

    \_flushall();//для очищения потока ввода

    getchar();

    return 0;

}

1. Проверка работы программы на тестовых примерах

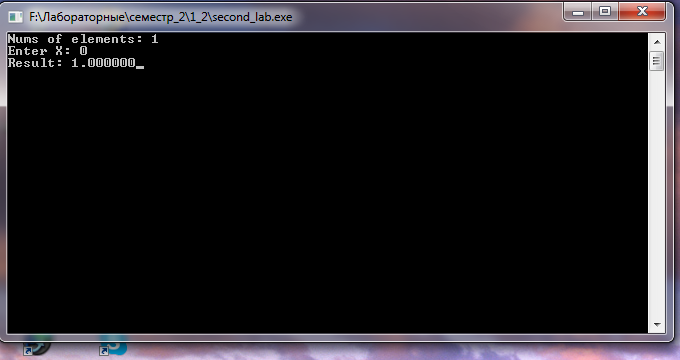


Рисунок 2 – Тест №1

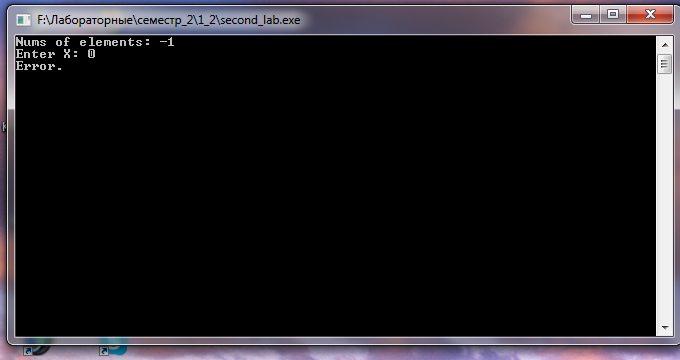


Рисунок 3 – Тест №2

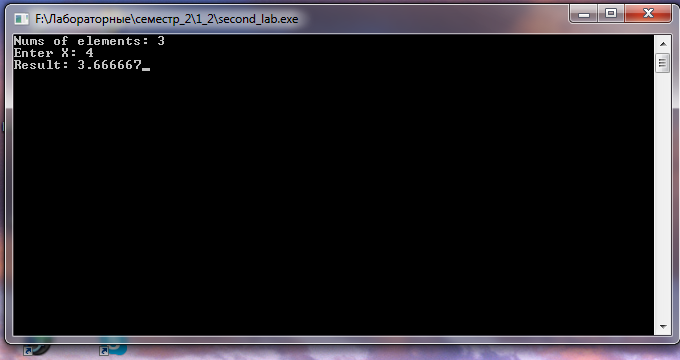


Рисунок 4 – Тест №3

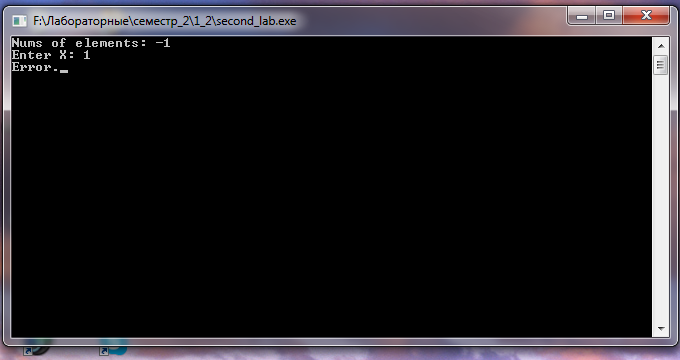


Рисунок 5 – Тест №4

1. Вывод.

Мы убедились ,что алгоритм верен и работает корректно на тестовых примерах.

Программа учитывает ввод некорректных данных (например, ввод отрицательного количества элементов).