ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
О ПРАКТИЧЕСКОЙ РАБОТЕ №3

По дисциплине «Процедурное программирование»

Выполнил: ст. гр. ТКИ – 112

Зинченко Б.А.

Проверил: к.т.н., доц.

Васильева М.А.

Москва 2021

**Оглавление**

[ЗАДАНИЕ 3.1 3](#_Toc72570664)

[ЗАДАНИЕ 3.2 6](#_Toc72570665)

[ЗАДАНИЕ 3.3 9](#_Toc72570666)

ЗАДАНИЕ 3.1

1. Условие задачи

Протабулировать заданную в таблице функцию. Использовать данные в таблице значения шага и интервала в качестве ввода пользователя для решения тестового примера. При невозможности расчёта функции в конкретной точке выводить её значение и надпись, означающую отсутствие решения.

1. Блок-схема алгоритма

Рисунок 1 – Блок-схема 3.1

1. Текст программы на языке C++

#define \_USE\_MATH\_DEFINES

#include <cmath>

#include <iostream>

#include <iomanip>

using namespace std;

/\*\*

\* \brief Функциия для расчета y.

\* \param x Параметр x.

\* \return Возвращает значение y.

\*/

double Calculate(const double x);

/\*\*

\* \brief Точка входа в программу

\* \return Возвращает нуль, в случае успеха

\*/

int main()

{

const auto left = 2.0;

const auto right = 4.0;

const auto step = 0.2;

auto x = left;

cout << " x" << setw(15) << "y\n";

while ((x < right) || (abs(x - right) < step))

{

if (isCalculated(x))

{

const auto y = Calculate(x);

cout << setw(10) << setprecision(2) << x << setw(15)

<< setprecision(5) << y << endl;

}

else

{

cout << setw(10) << setprecision(2) << x << setw(15)

<< "not defined" << endl;

}

x += step;

}

return 0;

}

bool isCalculated(const double x) {

return x >= numeric\_limits<double>::min();

}

double Calculate(const double x) {

return 3 \* x + 4 \* log(x) - 5;

}

Результаты работы программы

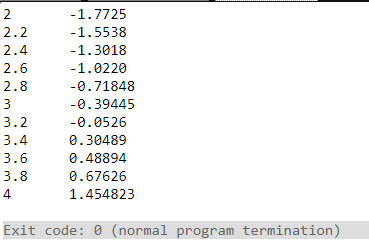


Рисунок 2 ˗ Вывод программы 3.1

1. Проверка расчётов в Excel

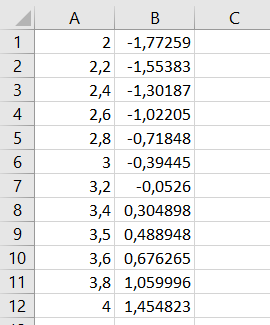


Рисунок 3 ˗ Проверка Расчёта программы 3.1

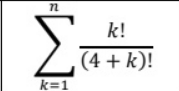
ЗАДАНИЕ 3.2

1. Условие задачи

Составьте две программы:

a) вычислить сумму первых членов последовательности

b) вычислить сумму всех членов последовательности, не меньших заданного числа



1. Блок-схема алгоритма



Рисунок 4 ˗ Блок-схема 3.2

1. Текст программы на языке C++

#include <iostream>

using namespace std;

/\*\*

\* \brief Функция расчета факториала числа

\* \param k число

\* \return значение факториала

\*/

double factorial(const int k);

/\*\*

\* \brief Функция расчета рекуррентного соотношения

\* \param index номер члена ряда

\* \return

\*/

double Recurrence(const size\_t index);

/\*\*

\* \brief Рассчитывает сумму ряда через рекуррентное соотношение

\* из n членов

\* \param size количество членов ряда

\* \return

\*/

double Sum(const size\_t n);

/\*\*

\* \brief Рассчитывает сумму ряда через рекуррентное соотношение

\* до тех пор, пока текущий член ряда не превосходит epsilon

\* \param epsilon точность расчета суммы ряда

\* \return

\*/

double Sum1(const double epsilon);

/\*\*

\* \brief Точка входа в программу.

\* \return Код ошибки (0 - успех).

\*/

int main()

{

int n;

cout << "Enter n » ";

cin >> n;

if (cin.fail()) {

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Вы ввели не число!" << endl;

}

else {

cout << "result for n = " << Sum(n) << endl;

}

double epsilon;

cout << "enter eps » ";

cin >> epsilon;

if (cin.fail()) {

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Вы ввели не число!" << endl;

}

else {

cout << "\n res for EPS = " << Sum1(epsilon) << endl;

}

return 0;

}

double factorial(const int k)

{

int res = 1;

for (int i = 1; i <= k; i++) {

res = res \* i;

}

return res;

}

double Recurrence(const size\_t index)

{

return factorial(index) / factorial(4 + index);

}

double Sum(const size\_t n)

{

double sum = 0;

size\_t k = 0;

while (k < n)

{

k++;

sum += factorial(k) / factorial(4 + k);

}

return sum;

}

double Sum1(const double epsilon)

{

auto previous = 1.0 / (2 \* 3 \* 4 \* 5);

auto sum = previous;

auto k = 1;

while (abs(previous) > epsilon)

{

const auto current = Recurrence(k) \* previous;

sum += current;

previous = current;

k++;

}

return sum;

}

1. Результаты работы программы

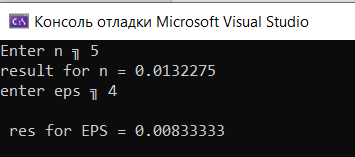


Рисунок 5 ˗ Вывод программы 3.2

1. Проверка расчётов в Excel

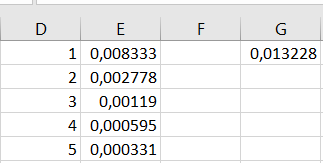
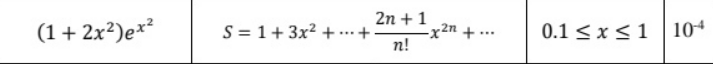


Рисунок 6 ˗ Проверка Расчёта программы 3.2

ЗАДАНИЕ 3.3

1. Условие задачи

Протабулировать заданную функцию и сумму функционального ряда разложения этой функции на интервале и с шагом (шаг и интервал задается в константах). Функциональнй ряд вычисляется по соответствующей рекуррентной формуле с заданной точностью . В результате показать три столбца: значение аргумента, значение функции в данной точке и значение суммы ряда, вычисленное с заданной точностью в данной точке. Два последних столбца должны иметь близкие результаты.



1. Блок-схема алгоритма

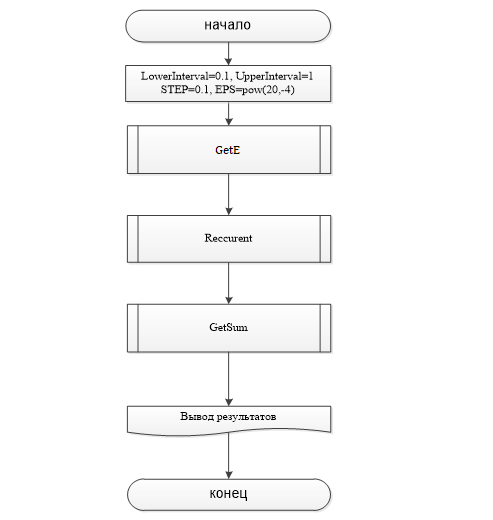


Рисунок 7 ˗ Блок-схема 3.3

1. Текст программы на языке C++

#define \_USE\_MATH\_DEFINES

#include <cmath>

#include <iostream>

#include <iomanip>

using namespace std;

/\*\*

\* \brief функция для расчета y

\*\param Параметр x

\*

\*\reurn

\*/

double Calculate(const double x);

/\*\*

\* \brief Функциия для расчета суммы.

\* \param x Параметр x,n2.

\* \return Возвращает значение sum.

\*/

double Reccurent(const double x, const int n2);

/\*\*

\* \brief Точка входа в программу

\* \return Возвращает нуль, в случае успеха

\*/

int main()

{

int n2 = 0;

auto sum = 0.0;

const auto left = 0.1;

const auto right = 0.9;

const auto step = 0.1;

cout << "Please print n\n";

cin >> n2;

auto x = left;

cout << " x" << setw(15) << "y" << setw(20) << "sum\n";

while ((x < right) || (abs(x - right) < step))

{

if (isCalculated(x))

{

const auto y = Calculate(x);

const auto z = Reccurent(x, n2);

cout << setw(10) << setprecision(2) << x << setw(15)

<< setprecision(5) << y << setw(20) << setprecision(4) << z << endl;

}

else

{

cout << setw(10) << setprecision(2) << x << setw(15)

<< "not defined" << endl;

}

x += step;

}

return 0;

}

bool isCalculated(const double x) {

return x >= numeric\_limits<double>::min();

}

double Calculate(const double x) {

return ((1 + 2 \* pow(x, 2)) \* pow(M\_E, pow(x, 2)));

}

double Reccurent(const double x, const int n2)

{

int n1;

float sum = 0;

int answer = 1;

for (n1 = 1; n1 <= n2; n1++)

{

answer = answer \* (n1);

sum = sum + ((2 \* x) \* pow(x, 2 \* n1)) / (answer);

}

return (sum);

}Результаты работы программы

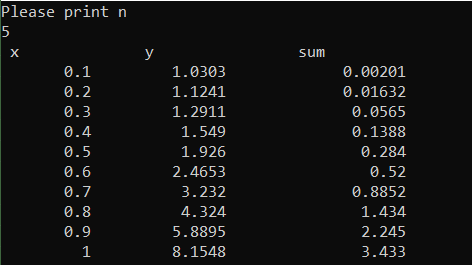


Рисунок 8 ˗ Вывод программы 3.3

1. Проверка расчётов в Excel

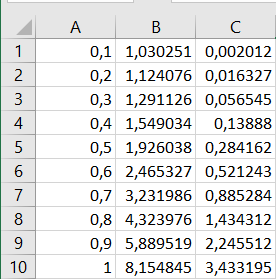


Рисунок 9 ˗ Проверка Расчёта программы 3.3