ГУАП

КАФЕДРА № 83

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ст. препод. |  |  |  | К.А. Жиданов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1 |
|  |
| по курсу: ЯЗЫКИ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 3145 |  |  |  | А.А. Одинцов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

**Вариант 2**

Вычислить значение косинуса с заданной точностью (ряд Тейлора)

**Цель работы**

Изучить методы разработки консольных приложений, способы их запуска и обработки кодов возврата.

**Ход работы**

1. Реализовал на языке Си функцию, выполняющую вычисление тангенса через формулы синуса и косинуса для ряда Тейлора:

double mycos(double x, double ep)

{

double S = 0, S\_1;

int zn = 1, n = 1, sign = 1;

//если х меньше 0, берем по модулю

if (x < 0) x = fabs(x);

//если x > 2Pi перемещаем в интервал [0..2pi]

x = fmod(x, 2 \* PI); //остаток от деления x/(2\*pi) , убираем период

//x > pi перемещаем в интервал [0..pi]

if (x > PI)

{

x = 2 \* PI - x; // чтобы получить угол от 0 до пи, вычитаем из 2пи найденный угол >пи

}

//x > (pi/2) перемещаем в интервал [0..pi/2]

if (x > (PI / 2))

{

x = PI - x;

sign = sign \* (-1); //т.к от 0 до пи/2 знак + а от пи/2 до пи знак -, то меняем знак

}

do

{

S\_1 = S; //предыдущее значение суммы

S = S + (zn \* pow(x, 2 \* n - 2) / fact(2 \* n - 2));

n = n + 1;

zn = zn \* (-1);

} while (fabs(S - S\_1) > ep);

S = S \* sign;

return S;

}

1. Проверка программы и вывод ошибки с использованием стандартной функции cos()

int test\_cos(void)

{

double g; // для подсчета угла

for (g = -2 \*PI; g <= 2 \* PI; g = g + PI / 10.) // с шагом 1/10 PI идем до 6PI от нуля

{

if (fabs(fmod(fabs(g), PI) - PI / 2)>=EP) //точки PI/2 убираем

{

printf("mycos(%8.5f) = %9.6f cos(%8.5f) =%9.6f\n", g, mycos(g, EP),g, cosl(g));

// печать двух вещ. чисел с заданной точностью

}

else

{

return 1; // Если угол близится к PI/2 то код ошибки 1

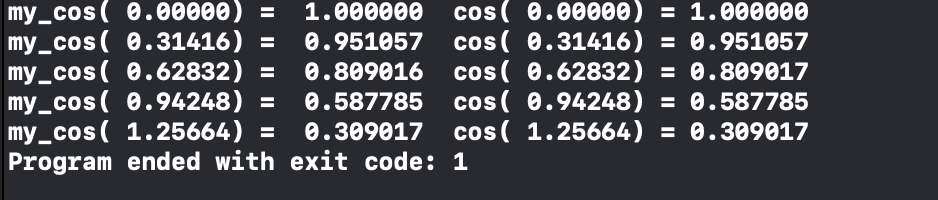
}

}

return 0;

}

При выводе от 0 до PI/2:

****

1. В консоль выводится код ошибки 1 если при вычитании от модуля угла без периода, мы получаем угол, близкий к Pi/2
2. Компиляция программы успешная, значения совпадают со стандартной функцией cos(), в конце получаем код ошибки «1»
3. Полный код программы:

#include <stdio.h>

#include <math.h>

#define PI 3.14159265358979323846

#define EP 0.0001

int fact(int k)

{

int f = 1, i;

for (i = 2; i <= k; i++)

f = f \* i;

return f;

}

double my\_cos(double x)

{

double S = 0, S\_1;

int zn = 1, n = 1, sign = 1;

//если х меньше 0, берем по модулю

if (x < 0) x = fabs(x);

//если x > 2PI перемещаем в интервал [0..2PI]

x = fmod(x, 2 \* PI);

//остаток от деления x/(2\*PI) , убираем период

//x > PI перемещаем в интервал [0..PI]

if (x > PI)

{

x = 2 \* PI - x;

// чтобы получить угол от 0 до пи, вычитаем из 2пи найденный угол >пи

}

//x > (PI/2) перемещаем в интервал [0..PI/2]

if (x > (PI / 2))

{

x = PI - x;

sign = sign \* (-1);

//т.к от 0 до пи/2 знак + а от пи/2 до пи знак -, то меняем знак

}

do

{

S\_1 = S; //предыдущее значение суммы

S = S + (zn \* pow(x, 2 \* n - 2) / fact(2 \* n - 2));

n = n + 1;

zn = zn \* (-1);

} while (fabs(S - S\_1) > EP);

S = S \* sign;

return S;

}

int test\_cos(void)

{

double g; // для подсчета угла

for (g = 0; g <= 2 \* PI; g = g + PI / 10.) // с шагом 1/10 PI идем до 2PI от нуля

{

if (fabs(fmod(fabs(g), PI) - PI / 2)>=EP) //точки PI/2 убираем

{

printf("my\_cos(%8.5f) = %9.6f cos(%8.5f) =%9.6f\n", g, my\_cos(g),g, cos(g));

// печать двух вещ. чисел с заданной точностью

}

else

{

return 1; // Если угол близится к PI/2 то код ошибки 1

}

}

return 0;

}

int main() {

return test\_cos();

}

**Вывод**

Изучил методы разработки консольного приложения, способ его запуска и обработки кодов возврата