ГУО “БГУИР”

Факультет компьютерного проектирования

Кафедра Инженерной психологии и эргономики

Отчёт по

Лабораторной работе №7

СРЕДА ПРОГРАММИРОВАНИЯ VISUAL C++. ПРОГРАММИРОВАНИЕ ЛИНЕЙНЫХ АЛГОРИТМОВ

Подготовил:

Студент гр.110101

Ладутько Я.Д.

Проверила:

Семижон Е.А.

Минск 2022

***Цель работы****:* изучить алгоритмы аппроксимации функций; освоить методику построения и использования алгебраических интерполяционных многочленов Лагранжа и Ньютона

Вариант 16.



Код:

**///MySpace.h**

#pragma once

#include <iostream>

template <typename T>

void input(T& a)

{

while (true)

{

std::cin >> a;

if (std::cin.fail() || (std::cin.peek() != '\n' && std::cin.peek() != ' ' && std::cin.peek() != '\t'))

{

std::cin.clear();

std::cin.ignore(10000, '\n');

std::cout << "Неверный ввод" << std::endl;

}

else

{

return;

}

}

}

void input(int& x, int a, int b);

void input(double& x, double a, double b);

void input(bool& b);

**///MySpace.cpp**

#include"MySpace.h"

using namespace std;

void input(int& x, int a, int b)

{

while (true)

{

input(x);

if (x < a || x > b)

{

cout <<

"Неверный ввод\n"

"Введите число на промежутке от " << a << " до " << b << endl;

}

else

return;

}

}

void input(double& x, double a, double b)

{

while (true)

{

input(x);

if (x < a || x > b)

{

cout <<

"Неверный ввод\n"

"Введите число на промежутке от " << a << " до " << b << endl;

}

else

return;

}

}

void input(bool& b)

{

char c;

while (true)

{

input(c);

if (c != 'y' && c != 'n')

{

cout <<

"Неверный ввод\n"

"Введите y или n\n";

}

else

{

b = (c == 'y' ? true : false);

return;

}

}

}

**///Lab8.cpp**

#include <iostream>

#include <cmath>

#include "MySpace.h";

using namespace std;

double func(double x)

{

return 0.1\*x\*x\*x + x\*x - 10\*sin(x);

}

double method(double xt, double\* m\_x, double\* m\_y, int m, bool &is\_ok)

{

is\_ok = true;

int i = 2;

if (xt < m\_x[0] || xt > m\_x[m - 1])

{

is\_ok = false;

return 0;

}

while (xt > m\_x[i])

i++;

i--;

double n1 = m\_y[i - 1] + (xt - m\_x[i - 1]) \* (m\_y[i] - m\_y[i - 1]) / (m\_x[i] - m\_x[i - 1]);

double n2 = n1 + (xt - m\_x[i - 1]) \* (xt - m\_x[i]) \*

((m\_y[i - 1] - m\_y[i]) / (m\_x[i - 1] - m\_x[i]) - (m\_y[i] - m\_y[i + 1]) / (m\_x[i] - m\_x[i + 1])) /

(m\_x[i - 1] - m\_x[i + 1]);

return n2;

}

int main()

{

int a = -4, b = 2, m, n;

double res1 = 0, res2 = 0;

setlocale(LC\_ALL, "RU");

cout << "Введите количество узлов [1, 10000]: ";

input(m, 1, 10000);

cout << "Введите количество точек аппроксимации [" << m <<", 10000]: ";

input(n, m, 10000);

double \*m\_x = new double[m];

double \*m\_y = new double[m];

double\* y\_t = new double[n];

double h = (b - a) / (double)(m-1);

double h1 = (b - a) / (double)(n-1);

double x = a;

for (int i = 0; i < m; i++)

{

m\_x[i] = x;

m\_y[i] = func(x);

x += h;

}

x = a;

double max = 0, f;

bool is\_ok;

for (int i = 0; i < n; i++, x += h1)

{

y\_t[i] = method(x, m\_x, m\_y, m, is\_ok);

f = func(x);

if(is\_ok)

{

cout << "xt = " << x << " f\*(x) = " << y\_t[i] << endl;

if (abs(f - y\_t[i]) > max)

{

max = abs(f - y\_t[i]);

}

}

}

cout << "Погрешность: " << max << endl;

system("pause");

}

***Вывод****:* изучил алгоритмы аппроксимации функций; освоил методику построения и использования алгебраических интерполяционных многочленов Лагранжа и Ньютона

