**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

**«Приближение функций с помощью интерполяционного полинома»**

**Студент**

Тев Никита Михайлович

2 курс, 2 группа

**Преподаватель**

Никифоров Иван Васильевич

**Минск 2019**

**1. Постановка задачи**

На отрезке [0; 1] задана функция . Разбить исходный отрезок на n частей, на каждом из которых приблизить функцию полиномом Лагранжа третьей степени, построенным по чебышевским узлам. Построить график.

**2. Входные данные**

n1 = 10, n2 = 20, n3 = 40.

**3. Листинг программы**

#include "stdafx.h"

#include <iostream>

#include <fstream>

#include <iomanip>

#include <vector>

#include <cmath>

#define PI 3.14159265359

using namespace std;

double A = 0;

double B = 1;

ofstream fout1("output\_11.txt");

ofstream fout2("output\_21.txt");

ofstream fout3("output\_41.txt");

//интерполируемая функция

double f(double x) {

return x \* exp(x) + x \* x;

}

//вспомогательная функция для построения интерполяционного многочлена

double L(int k, double x, vector<double> &v){

double ans = 1.0;

for (int i = 1; i < v.size(); i++) {

if (i != k) {

ans \*= (x - v[i]);

ans /= (v[k] - v[i]);

}

}

return ans;

}

// отрезок [A,B] делим на n меньших отрезков, на каждом из которых отдельно производится интерполяция по Чебышевским узлам

void perform(int n, ofstream& fout) {

vector< vector<double>> v(n);

for (int i = 0; i < n - 1; i++) {

vector<double> tmp;

v.push\_back(tmp);

v[i].push\_back(-1.0);

//рассчитываем границы отрезка

double a = i \* (B - A) / (n - 1.0);

double b = (i + 1) \* (B - A) / (n - 1.0);

//рассчитываем 4 узла Чебышева для отрезка

for (int j = 1; j <= 4; j++)

{

v[i].push\_back((a + b) / 2 + ((b - a) / 2) \* cos((2 \* j - 1) \* PI / 8.0));

}

//на отрезке выбираем 10 точек, затем считаем в них значения интерполяционного многочлена и непосредственно интреполируемой функции

vector<double> x;

for (int p = 0; p < 10; p++) {

x.push\_back(a + p \* (b - a) / 9.0);

}

for (int k = 0; k < x.size(); k++) {

double ans = 0;

for (int j = 1; j <= 4; j++) {

ans += f(v[i][j])\*L(j, x[k], v[i]);

}

fout << setprecision(8) << fixed << x[k] << " " << f(x[k]) << " " << ans << endl;

}

}

}

int main()

{

int n1 = 11;

int n2 = 21;

int n3 = 41;

perform(n1, fout1);

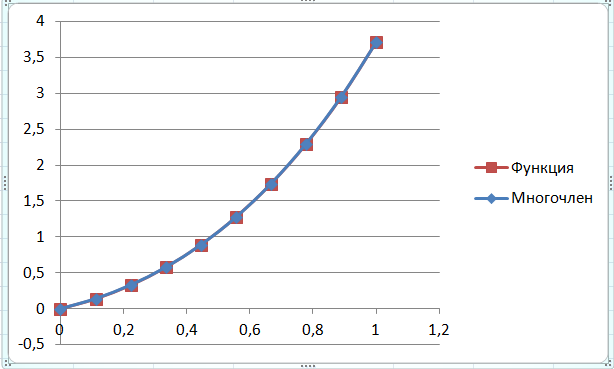
perform(n2, fout2);

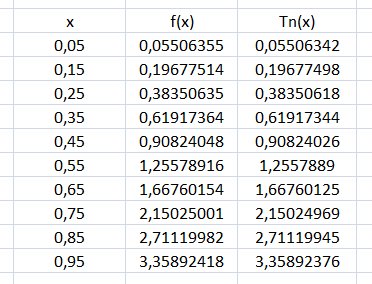
perform(n3, fout3);

return 0;

}

**4. Выходные данные**

Даже при делении отрезка на 10 частей, точность интерполяции настолько высока, что на графике не представляется возможным увидеть разницу между исходной функцией и интреполяционным полиномом.

Поэтому я приведу полученные значения в виде таблицы (для n=10).