Мiнiстерство освiти i науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського» Iнститут прикладного системного аналiзу

**Лабораторна робота № 5**

з курсу «Чисельнi методи 1»

з теми «Інтерполювання фцнкцій»

Варіант № 16

Виконав студент 2 курсу групи КА-91

Панченко Єгор Станіславович

перевiрила старший викладач

Хоменко Ольга Володимирiвна

Київ-2021

**Завдання: дана функція . Побудувати інтерполяційний многочлен Лагранжа, перший та другий інтерполяційні поліноми Ньютона.**

1. **Текст програм**

#include <vector>  
#include <math.h>  
#include <cstdio>  
#include <assert.h>  
  
using namespace std;  
  
namespace Laba5 {  
 double f(double x) {  
 return (x + 8.0 / (1 + exp(x / 4)));  
 }  
  
  
}  
  
double gnLangrange(double x, vector < double > points) {  
 double res = 0;  
 for (int i = 0; i < points.size(); i++) {  
  
 double mlt = 1;  
 for (int j = 0; j < points.size(); j++) {  
  
 if (i == j) {  
 continue;  
 }  
  
 mlt \*= (x - points[j]) / (points[i] - points[j]);  
 }  
  
 res += Laba5::f(points[i]) \* mlt;  
 }  
  
 return res;  
}  
  
void PolLangrange() {  
  
 printf("Interval is [-10, 10]\n");  
 printf("Langrange\n");  
  
 vector < double > points = {-4, -2, 0, 2, 4};  
  
 vector < double > another\_points = {-10, -5, -3, -1, 1, 3, 5, 10};  
  
 printf("Values in points: f g\n");  
 for (auto i : points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnLangrange(i, points));  
 }  
  
 printf("Values in another\_points: f g\n");  
 for (auto i : another\_points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnLangrange(i, points));  
 }  
  
}  
  
double diff(vector < double > x) {  
 double res = 0;  
 for (int j = 0; j < x.size(); j++) {  
  
 double dn = 1;  
 for (int i = 0; i < x.size(); i++) {  
  
 if (i == j) {  
 continue;  
 }  
 dn \*= x[j] - x[i];  
 }  
  
 res += Laba5::f(x[j]) / dn;  
 }  
  
 return res;  
}  
  
double delta\_y(double i, double k, vector < double > &points) {  
 if (k == 0 /\*&& i >= 0 && i < points.size()\*/) {  
 return Laba5::f(points[i]);  
 }  
 /\*if (i < 0 || i >= points.size()) {  
 return 0;  
 }\*/  
 return (delta\_y(i + 1, k - 1, points) - delta\_y(i, k - 1, points));  
}  
  
double gnNewtonLection(double x, vector < double > points) {  
 /\*for (int i = 0; i <= 5; i++) {  
 printf("%lf\n", delta\_y(0, i, points));  
 }\*/  
 double res = 0;  
 double mlt = 1;  
  
 for (int i = 0; i < points.size(); i++) {  
 res += mlt \* delta\_y(0, i, points);  
 if (i + 1 == points.size()) {  
 break;  
 }  
 mlt \*= x - points[i];  
 mlt /= i + 1;  
 mlt /= 2;  
 }  
  
 return res;  
}  
  
double gnNewtonLectionSecond(double x, vector < double > points) {  
 /\*for (int i = 0; i <= 5; i++) {  
 printf("%lf\n", delta\_y(0, i, points));  
 }\*/  
 double res = 0;  
 double mlt = 1;  
  
 for (int i = 0; i < points.size(); i++) {  
 res += mlt \* delta\_y(points.size() - 1 - i, i, points);  
 if (i + 1 == points.size()) {  
 break;  
 }  
 mlt \*= x - points[points.size() - 1 - i];  
 mlt /= i + 1;  
 mlt /= 2;  
 }  
  
 return res;  
}  
  
double gnNewton(double x, vector < double > points) {  
 double res = 0;  
  
 vector < double > curr;  
 double mlt = 1;  
 for (int i = 0; i < points.size(); i++) {  
  
 curr.push\_back(points[i]);  
 res += diff(curr) \* mlt;  
 if (i == points.size()) {  
 break;  
 }  
 mlt \*= x - points[i];  
  
 }  
  
 return res;  
}  
  
void PolNewtonFirst() {  
 printf("Interval is [-10, 10]\n");  
 printf("Newton\n");  
  
 vector < double > points = {-4, -2, 0, 2, 4};  
  
 vector < double > another\_points = {-10, -5, -3, -1, 1, 3, 5, 10};  
  
 printf("Values in points: f g\n");  
 for (auto i : points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLection(i, points));  
 }  
  
 printf("Values in another\_points: f g\n");  
 for (auto i : another\_points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLection(i, points));  
 }  
}  
  
void PolNewtonSecond() {  
 printf("Interval is [-10, 10]\n");  
 printf("Newton\n");  
  
 vector < double > points = {-4, -2, 0, 2, 4};  
  
 vector < double > another\_points = {-10, -5, -3, -1, 1, 3, 5, 10};  
  
 printf("Values in points: f g\n");  
 for (auto i : points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLectionSecond(i, points));  
 }  
  
 printf("Values in another\_points: f g\n");  
 for (auto i : another\_points) {  
 printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLectionSecond(i, points));  
 }  
}

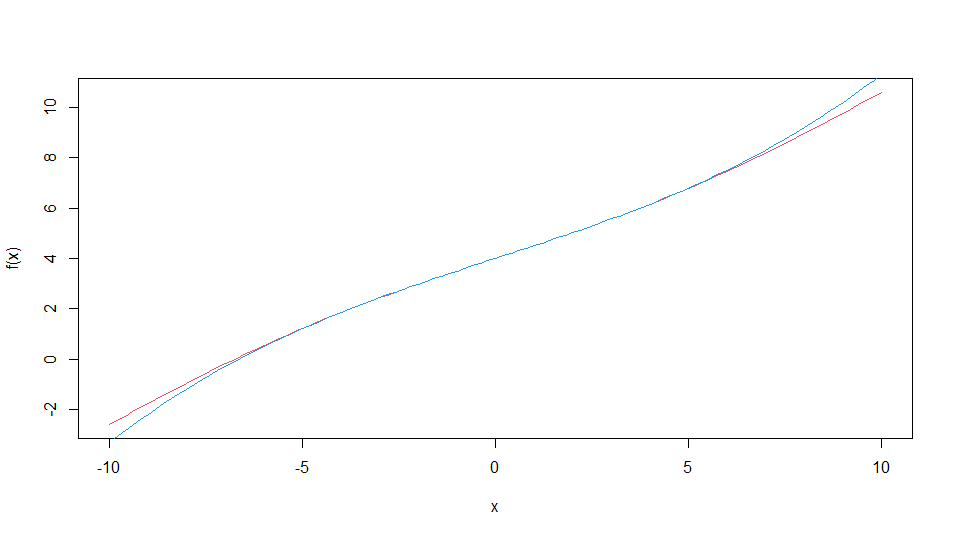
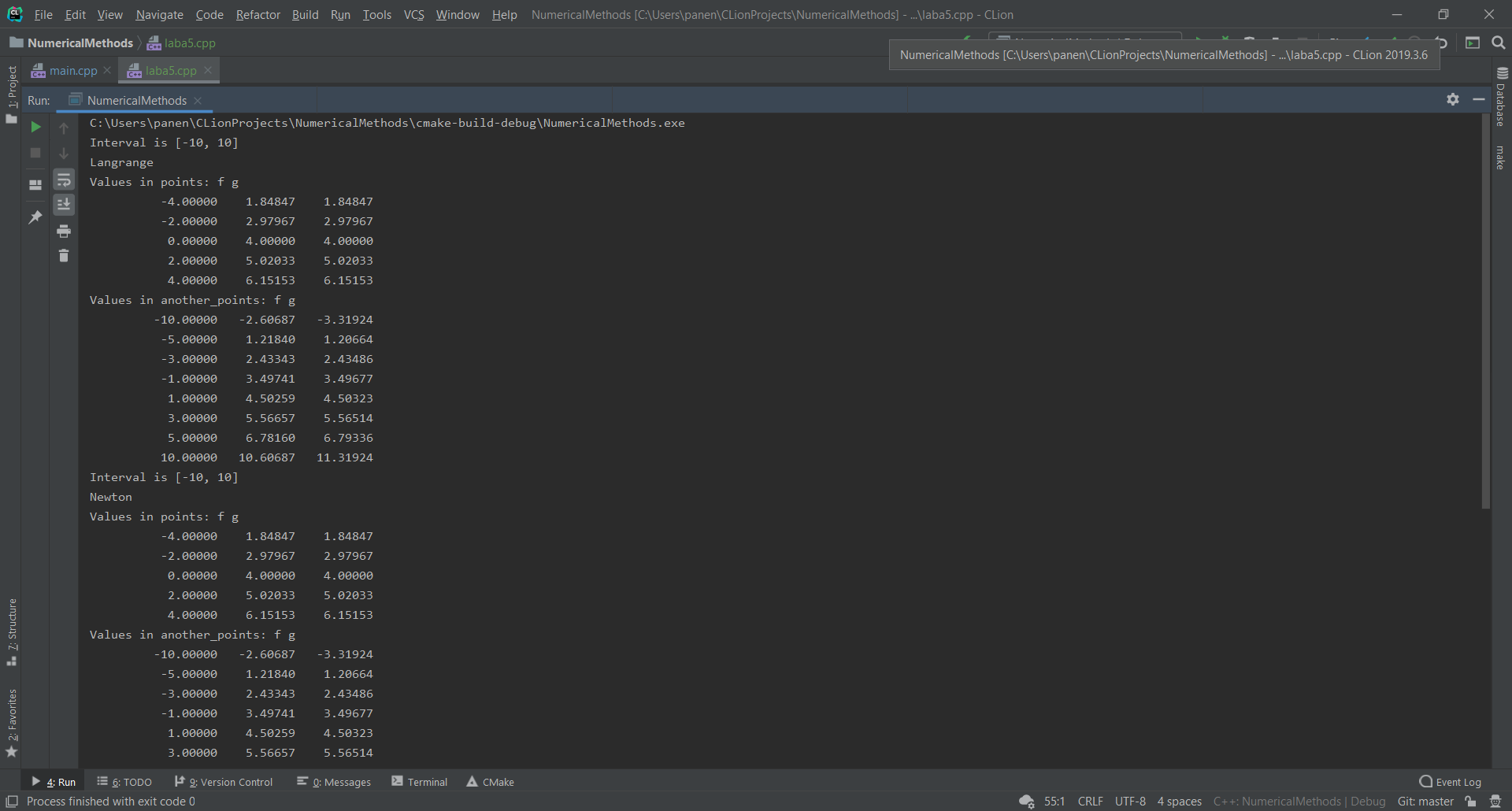
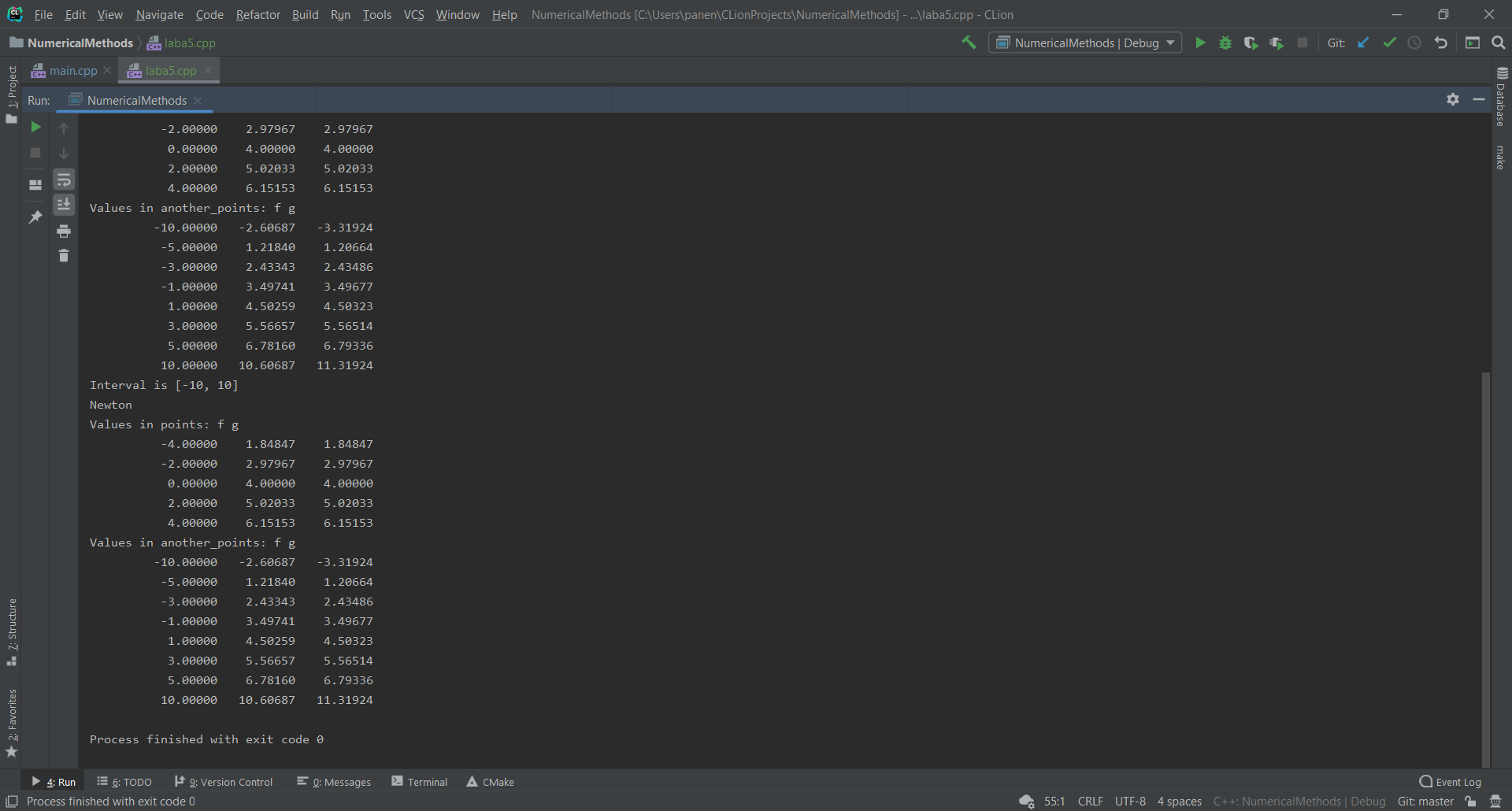
1. **Графіки поліномів**

Рис.3.3 Червона крива - f(x); Синя крива - інтерполяційні многочлени

1. **Результат роботи програми**



1. **Висновок**

У ході виконання лабораторної роботи було програмно реалізовано метод побудови інтерполяційного поліному Лагранжа, першого та другого інтерполяційних многочленів Ньютона.

Таблиця скінченних різниць побудована не була – замість неї реалізовано функцію , яка дозволяє вирахувати скінченну різницю -го порядку з індексом лише за числами та

З графіків можна зробити висновок, що функція дуже гарно апроксимується в околі нуля. При більших значеннях функції починають відрізнятися все більше і більше.

Письмова частина

При обрахунках було допущено округлення з метою спрощення розрахунків. Виявилося, що кубічні сплайни та інтерполяційний многочлен Лагранжа приймають один й той самий вигляд.

