#### МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение   
высшего профессионального образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №2

КУРСОВАЯ РАБОТА (ПРОЕКТ)   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ (ПРОЕКТУ) |
| Этап 3  **Тема**: «АППРОКСИМАЦИЯ ФУНКЦИИ  МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ» |
| по дисциплине: ИНФОРМАТИКА |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4710 |  |  |  | Хмелев Н.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

**Текст программы**

Mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <stdio.h>

MainWindow::MainWindow(QWidget \*parent) :QMainWindow(parent),

ui(new Ui::MainWindow){

ui->setupUi(this);

}

MainWindow::~MainWindow(){delete ui;}

void MainWindow::split(QString string, float \*array){

QStringList tmp = string.split(' ');

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

array[i] = tmp.at(i).toFloat();

}

void MainWindow::mainWork(){

coeff(arrayOfX, arrayOfY, matrix);

stringMatrix(matrix);

if(!convergence(matrix)){

ui->unswerLabel->setText("No convergence");

return;}

zedelMethod(matrix , accuracy, C);

QString arrayOfC = "\n";

for(int i = 0; i < 3; i++)

arrayOfC += "C" + QString::number(i) +" = " +

QString::number(transform(C[i])) + "\n";

output += arrayOfC;

fi(funcFi, arrayOfX, C);

QString func = "\nValue of the approximated function :\n";

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

func += QString::number(transform(funcFi[i])) + "\t";

output += func;

diff(funcFi, difference, arrayOfY);

QString diference = "\nDifference of function : \n";

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

diference += QString::number(transform(difference[i])) + "\t";

output += diference + "\n";

unsver = sq(difference, sqare);

output += "Unswer is : ";

output += QString::number(transform(unsver)) + "\n";

ui->unswerLabel->setText(output);}

void MainWindow::coeff(float \*x,float \*y, float \*matrix){

for( int i = 0; i < WIDTH\*3;i += 4){

for(int j = 0; j < WIDTH-1; j++){

matrix[i + j] = 0;

for(int k = 0; k < LENGTH;k++)

matrix[i + j] += fi(x[k], i / 4) \* fi(x[k], j);}

matrix[i + WIDTH]=0;

for(int k = 0;k < LENGTH; k++)

matrix[i + 3] += y[k] \* fi(x[k],i / 4);

}

}

float MainWindow::fi(float x,int i){

switch(i){

case 0 :

return 1;

case 1:

return 2 \* x;

default :

return 4 \* x \* x - 2;

}

}

void MainWindow::stringMatrix(float \*matrix){

output = "";

for(int i = 0; i < 3; i ++){

for( int j = 0; j < 4; j++)

output += QString::number(transform(matrix[i \* 4 + j])) + "\t";

output += "\n";

}

}

void MainWindow::zedelMethod(float \*matrix, float accuracy, float \*C){

float tmp1;

int n = WIDTH;

float tmp[HIGH];

do{

for( int i = 0; i < HIGH;i ++){

tmp[i] = C[i];

C[i] = 0;

for(int j = 0; j < WIDTH; j++){

if(j == 3)

;

else if(i != j)

C[i] -= matrix[i \* n + j] \* C[j];

}

C[i] /= matrix[i \* n + i];

tmp1 = abs(abs(tmp[i]) - abs(C[i]));

}

}while(tmp1 >= accuracy);

}

float MainWindow::abs(float x){

if(x < 0)

x = -1 \* x;

return x;

}

float MainWindow::diff(float \*fucFi, float \*difference, float \*y){

float result = 0;

for(int i = 0; i < LENGTH; i++){

difference[i] = fucFi[i] - y[i];

result += difference[i];

}

return result;

}

float MainWindow::fi(float \*fucFi, float \*x, float \*C){

float result = 0;

for(int i = 0; i < LENGTH; i++){

for(int j = 0; j<HIGH; j++)

switch (j){

case 0:

fucFi[i] += C[j] \* 1;

break;

case 1:

fucFi[i] += C[j] \* x[i];

break;

case 2:

fucFi[i] += (4 \* x[i] \* x[i] - 2) \* C[j];

break;

}

result = fucFi[i];

}

return result;

}

float MainWindow::sq(float \*difference, float \*sqare){

float result = 0;

for(int i = 0; i < LENGTH; i++){

sqare[i] = difference[i] \* difference[i];

result += sqare[i];

}

return result;

}

void MainWindow::on\_calcButton\_clicked()

{

QString tmp = ui->editX->toPlainText();

split(tmp, arrayOfX);

tmp = ui->editY->toPlainText();

split(tmp, arrayOfY);

tmp = ui->editAccuracy->toPlainText();

accuracy = tmp.toFloat();

mainWork();

}

float MainWindow::transform(float &a){

int b = a \* 1000;

a = (float) b / 1000;

return a;

}

bool MainWindow::convergence(float \*matrix){

for( int i = 0; i < WIDTH\*3;i += 4){

float tmp = 0, tmp2;

for(int j = 0; j < WIDTH-1; j++)

if(i/4 != j)

tmp += abs(matrix[i + j]);

else

tmp2 = abs(matrix[i + j]);

if(tmp >= tmp2)

return false;

}

return true;

}

Mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#define LENGTH 5

#define HIGH 3

#define WIDTH 4

#include <QMainWindow>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~MainWindow();

float diff(float \*fucFi, float \*difference, float \*y);//yes

float fi(float \*fucFi, float \*x, float \*C);

float sq(float \*difference, float \*sqare);

void zedelMethod(float \*matrix, float accuracy, float \*C);//yes

void coeff(float \*x,float \*y, float \*matrix);//yes

void split(QString string, float \*array);//yes

void stringMatrix(float \*matrix);//yes

void mainWork();//yes

float abs(float x);//yes

float transform(float &a);

float fi(float x,int i);

bool convergence(float \*matrix);

private slots:

void on\_calcButton\_clicked();

private:

QString output;

float unsver;

float C[HIGH] = {0,0,0};

float arrayOfX[LENGTH], arrayOfY[LENGTH], matrix[3\*4], X, Y, accuracy;

float funcFi[LENGTH], difference[LENGTH], sqare[LENGTH];

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

Main.cpp

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow window;

window.setWindowTitle("Least-squares method");

window.show();

return a.exec();

}

