**Пермский национальный исследовательский  
политехнический университет**

**О Т Ч Ё Т**

**по творческой работе №1**

Калькулятор

Дисциплина: информатика

Вариант: 19

Выполнил работу

студент группы РИС-20-1б

Рябов Никита Андреевич

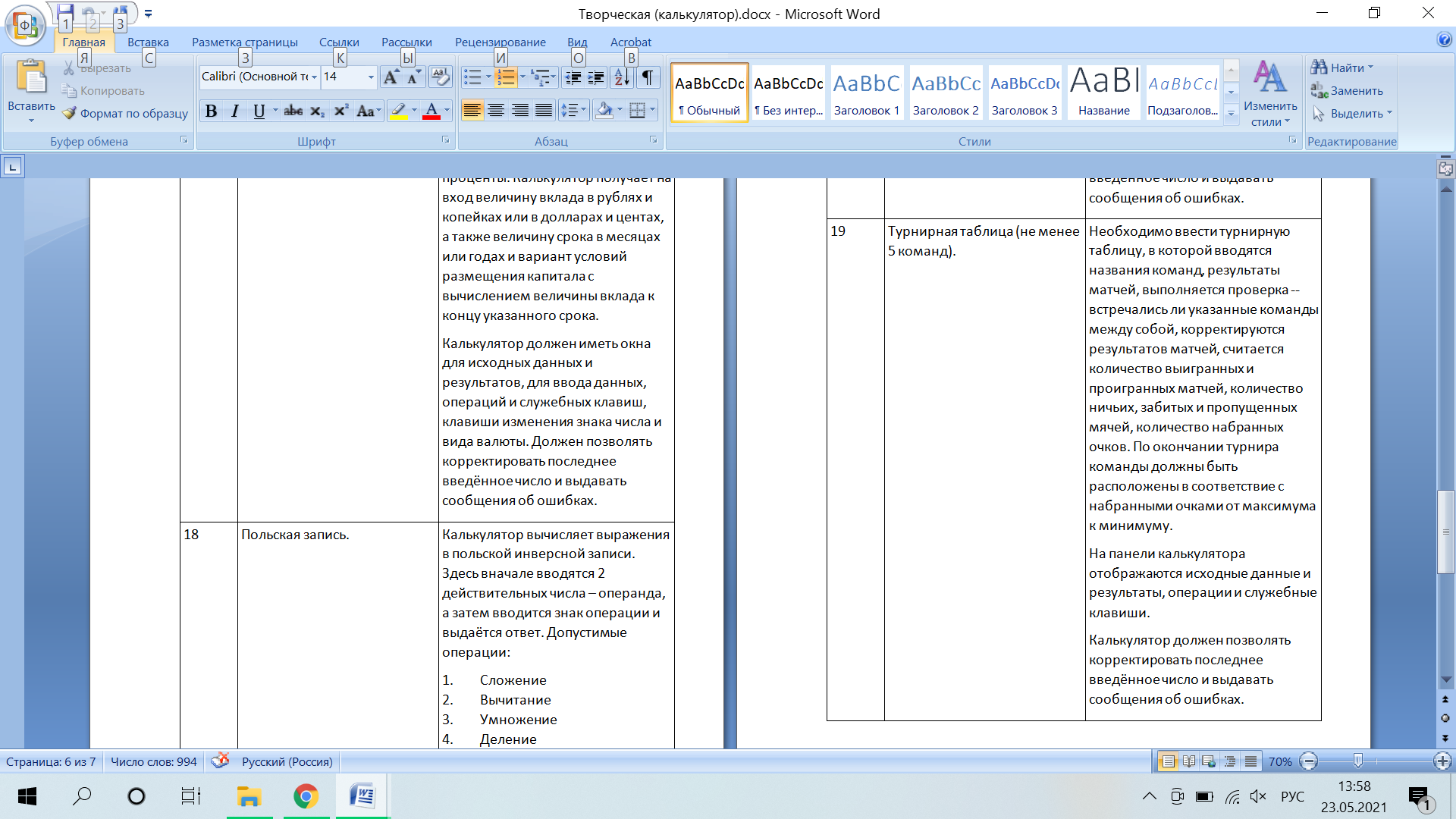
Проверила

Доцент кафедры ИТАС Полякова О.А.

Пермь, 2021

**Постановка задачи**

1. Разработать алгоритм калькулятора в соответствии с заданием.
2. Реализовать алгоритм в виде программы на алгоритмическом языке С++.
3. Разработать интерфейс средствами Qt.
4. Использование анимации, автоматизации, креативной визуализации повышает рейтинг автора проекта и влияет на экзаменационную оценку.



**Анализ задачи**

*Какие предстоит выполнить действия.*

Объединение виджетов в массивы для более удобного обращения к ним

//ввод названий

team\_l[0]=ui->lineEdit\_1; team\_r[0]=ui->lineEdit\_4;

team\_l[1]=ui->lineEdit\_6; team\_r[1]=ui->lineEdit\_5;

team\_l[2]=ui->lineEdit\_10; team\_r[2]=ui->lineEdit\_9;

team\_l[3]=ui->lineEdit\_15; team\_r[3]=ui->lineEdit\_21;

team\_l[4]=ui->lineEdit\_18; team\_r[4]=ui->lineEdit\_14;

team\_l[5]=ui->lineEdit\_19; team\_r[5]=ui->lineEdit\_24;

team\_l[6]=ui->lineEdit\_27; team\_r[6]=ui->lineEdit\_25;

team\_l[7]=ui->lineEdit\_31; team\_r[7]=ui->lineEdit\_30;

//ввод счета

score\_l[0]=ui->lineEdit\_2; score\_r[0]=ui->lineEdit\_3;

score\_l[1]=ui->lineEdit\_8; score\_r[1]=ui->lineEdit\_7;

score\_l[2]=ui->lineEdit\_12; score\_r[2]=ui->lineEdit\_11;

score\_l[3]=ui->lineEdit\_23; score\_r[3]=ui->lineEdit\_20;

score\_l[4]=ui->lineEdit\_17; score\_r[4]=ui->lineEdit\_16;

score\_l[5]=ui->lineEdit\_13; score\_r[5]=ui->lineEdit\_22;

score\_l[6]=ui->lineEdit\_26; score\_r[6]=ui->lineEdit\_28;

score\_l[7]=ui->lineEdit\_32; score\_r[7]=ui->lineEdit\_29;

//кнопки удаления

btn\_del[0]=ui->pushButton\_1;

btn\_del[1]=ui->pushButton\_2;

btn\_del[2]=ui->pushButton\_3;

btn\_del[3]=ui->pushButton\_4;

btn\_del[4]=ui->pushButton\_5;

btn\_del[5]=ui->pushButton\_6;

btn\_del[6]=ui->pushButton\_7;

btn\_del[7]=ui->pushButton\_8;

Настройка внешнего вида окна: установка иконок кнопок, сокрытие некоторых виджетов и так далее.

NumOfAct=1;

for (int i=0;i<8;i++)

{

Gdel->addButton(*btn\_del[i]*,i);

btn\_del[i]->setIcon(QIcon("C:/qtprojects/ArtProject1/icon1.png"));

btn\_del[i]->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

}

for (int i=1;i<8;i++)

{

team\_l[i]->hide();

score\_l[i]->hide();

score\_r[i]->hide();

team\_r[i]->hide();

btn\_del[i]->hide();

}

ui->menu\_save->setIcon(QIcon("C:/qtprojects/ArtProject1/icon2.png"));

ui->menu\_load->setIcon(QIcon("C:/qtprojects/ArtProject1/icon3.png"));

ui->result1->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

ui->btn1->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

ui->BtnPlus->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

//connection

Функция, которая показывает ранее скрытые виджеты для ввода результата матча.

void MainWindow::**Iplus**()

{

if (NumOfAct<8)

{

team\_l[NumOfAct]->show();

score\_l[NumOfAct]->show();

score\_r[NumOfAct]->show();

team\_r[NumOfAct]->show();

btn\_del[NumOfAct]->show();

NumOfAct++;

}

}

Функция, удаляющая запись матча. Скрывает часть виджетов. Если все скрыты, кроме последнего, то просто удаляет из него данные.

void MainWindow::**Idel**()

{

QPushButton\* button = qobject\_cast<QPushButton\*>(*sender()*);

int buttonID = Gdel->id(*button*);

if (NumOfAct!=1)

{

for (int i=buttonID;i<7;i++)

{

team\_l[i]->setText(team\_l[i+1]->text());

score\_l[i]->setText(score\_l[i+1]->text());

score\_r[i]->setText(score\_r[i+1]->text());

team\_r[i]->setText(team\_r[i+1]->text());

}

NumOfAct--;

team\_l[NumOfAct]->setText("");

score\_l[NumOfAct]->setText("");

score\_r[NumOfAct]->setText("");

team\_r[NumOfAct]->setText("");

team\_l[NumOfAct]->hide();

score\_l[NumOfAct]->hide();

score\_r[NumOfAct]->hide();

team\_r[NumOfAct]->hide();

btn\_del[NumOfAct]->hide();

}

else

{

team\_l[0]->setText("");

score\_l[0]->setText("");

score\_r[0]->setText("");

team\_r[0]->setText("");

}

}

Функция для построении рейтинга. Проходит по всем полям ввода и записывает количество забитых мячей в словарь. Затем данные перекладываются в список, сортируются и выводятся на экран.

void MainWindow::**Orating**()

{

QMap <QString,int> MAP;

QList <std::tuple<QString,int>> LIST;

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString d=team\_r[i]->text();

int b;

int c;

if (a==d && a!="")

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E007\n матч между одинаковыми командами");

return;

}

try

{

b=score\_l[i]->text().toInt();

c=score\_r[i]->text().toInt();

}

catch (...)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E001\n нецелочисленный счет");

return;

}

if (a.isEmpty() && d.isEmpty()) continue;

else if (!a.isEmpty() && !d.isEmpty())

{

if (MAP.find(a)==MAP.end()) MAP.insert(a,b);

else MAP.insert(a,MAP[a]+b);

if (MAP.find(d)==MAP.end()) MAP.insert(d,c);

else MAP.insert(d,MAP[d]+c);

}

else

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E002\n пустое имя команды");

return;

}

}

if (MAP.isEmpty())

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E003\n пустая таблица");

return;

}

for (auto it=MAP.begin();it!=MAP.end();it++)

{

std::tuple<QString,int> TUPLE(it.key(),*it.value()*);

LIST.push\_back(TUPLE);

}

std::sort(LIST.begin(),LIST.end(),comp);

QString text1="Рейтинг\n";

QString text2="🏆\n";

QString text3="⚽\n";

int count=1;

for (auto it=LIST.begin();it!=LIST.end();it++)

{

text1+=std::get<0>(*\*it*)+"\n";

text2+=QString::number(count)+"\n";

text3+=QString::number(std::get<1>(*\*it*))+"\n";

count++;

}

ui->textBrowser\_1->setText(text1);

ui->textBrowser\_2->setText(text2);

ui->textBrowser\_3->setText(text3);

}

Функция для вывода анализа игры какой-либо команды. Вначале определяется несколько переменных-счетчиков. Далее проверяются все поля ввода, и если нужная команда встретилась – счетчики обновляются.

void MainWindow::**Oanalis**()

{

int count\_win1=0;

int count\_win2=0;

int count\_win3=0;

int count\_iball=0;

int count\_oball=0;

int count\_games=0;

QString name=ui->ent1->text();

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString d=team\_r[i]->text();

int b;

int c;

if (a==d && a!="")

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E007\n матч между одинаковыми командами");

return;

}

try

{

b=score\_l[i]->text().toInt();

c=score\_r[i]->text().toInt();

}

catch (...)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E001\n нецелочисленный счет");

return;

}

if (a.isEmpty() && d.isEmpty()) continue;

else if (!a.isEmpty() && !d.isEmpty())

{

if (a!=name && d!=name) continue;

count\_games++;

if (d==name)

{

swap(*a*,*d*);

int e=c;

c=b; b=e;

}

if (b==c) count\_win3++;

if (b>c) count\_win1++;

if (b<c) count\_win2++;

count\_iball+=c;

count\_oball+=b;

}

else

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E002\n пустое имя команды");

return;

}

}

QString text1=name+"\n";

QString text2="\n";

QString text3="\n";

if (count\_games==0) text1+="не обнаружено";

else

{

text1=name+"\nматчи\nпобеды\nпоражения\nничьи\nзабитые\nпропущенные";

text2+=QString::number(count\_games)+"\n";

text2+=QString::number(count\_win1)+"\n";

text2+=QString::number(count\_win2)+"\n";

text2+=QString::number(count\_win3)+"\n";

text2+=QString::number(count\_oball)+"\n";

text2+=QString::number(count\_iball)+"\n";

text3+="--\n";

text3+=QString::number(qCeil(100.0\*count\_win1/count\_games))+"%\n";

text3+=QString::number(qCeil(100.0\*count\_win2/count\_games))+"%\n";

text3+=QString::number(qCeil(100.0\*count\_win3/count\_games))+"%\n";

text3+="--\n";

text3+="--\n";

}

ui->textBrowser\_1->setText(text1);

ui->textBrowser\_2->setText(text2);

ui->textBrowser\_3->setText(text3);

}

Функции для загрузки и сохранения турнира. Используется формат файла csv и стандартные файловые потоки для ввода/вывода.

void MainWindow::**MSave**()

{

QString path = QFileDialog::getSaveFileName(this,

QString::fromUtf8("Сохранить файл"),

QDir::currentPath(),

"Tables (\*.csv);;All files (\*.\*)");

QFile file(path);

if ( !file.*open*(QFile::WriteOnly | QFile::Text) )

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E004\n не удалось отрыть файл");

return;

}

else

{

QTextStream out(*&file*);

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString b=score\_l[i]->text();

QString c=score\_r[i]->text();

QString d=team\_r[i]->text();

out<<a<<","<<b<<","<<c<<","<<d<<"\n";

}

file.*close*();

}

ui->textBrowser\_1->setText("сохранено");

}

void MainWindow::**MLoad**()

{

QString path = QFileDialog::getOpenFileName(this,

QString::fromUtf8("Открыть файл"),

QDir::currentPath(),

"Tables (\*.csv);;All files (\*.\*)");

QFile file(path);

if ( !file.*open*(QFile::ReadOnly | QFile::Text) )

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E004\n не удалось отрыть файл");

return;

}

else

{

QTextStream in(*&file*);

int count=0;

while (!in.atEnd())

{

QStringList data = in.readLine().split(",");

if (data.size()!=4)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E005\n некорректный формат данных");

return;

}

team\_l[count]->setText(data[0]);

score\_l[count]->setText(data[1]);

score\_r[count]->setText(data[2]);

team\_r[count]->setText(data[3]);

team\_l[count]->show();

score\_l[count]->show();

score\_r[count]->show();

team\_r[count]->show();

btn\_del[count]->show();

count++;

if (count>7 && !in.atEnd())

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E006\n критический размер файла");

return;

}

}

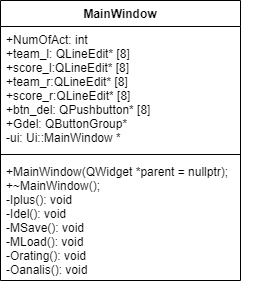
file.*close*();

}

NumOfAct=8;

ui->textBrowser\_1->setText("загружено");

**UML диаграмма**



**Код**

**Mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QLineEdit>

#include <QPushButton>

#include <QButtonGroup>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

int NumOfAct;

QLineEdit\* team\_l[8];

QLineEdit\* score\_l[8];

QLineEdit\* team\_r[8];

QLineEdit\* score\_r[8];

QPushButton\* btn\_del[8];

QButtonGroup\* Gdel =new QButtonGroup;

public:

**MainWindow**(QWidget \*parent = nullptr);

~***MainWindow***();

private slots:

void **Iplus**(); //input plus

void **Idel**(); //input delete

void **MSave**(); //menu save

void **MLoad**(); //menu download

void **Orating**();//output rating

void **Oanalis**();//output team analis

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Main.cpp**

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication a(*argc*, *argv*);

MainWindow w;

w.show();

return a.exec();

}

**Mainwindow.cpp**

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include <QMap>

#include <QList>

#include <QMessageBox>

#include <QtGui>

#include <QFileDialog>

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

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

//ввод названий

team\_l[0]=ui->lineEdit\_1; team\_r[0]=ui->lineEdit\_4;

team\_l[1]=ui->lineEdit\_6; team\_r[1]=ui->lineEdit\_5;

team\_l[2]=ui->lineEdit\_10; team\_r[2]=ui->lineEdit\_9;

team\_l[3]=ui->lineEdit\_15; team\_r[3]=ui->lineEdit\_21;

team\_l[4]=ui->lineEdit\_18; team\_r[4]=ui->lineEdit\_14;

team\_l[5]=ui->lineEdit\_19; team\_r[5]=ui->lineEdit\_24;

team\_l[6]=ui->lineEdit\_27; team\_r[6]=ui->lineEdit\_25;

team\_l[7]=ui->lineEdit\_31; team\_r[7]=ui->lineEdit\_30;

//ввод счета

score\_l[0]=ui->lineEdit\_2; score\_r[0]=ui->lineEdit\_3;

score\_l[1]=ui->lineEdit\_8; score\_r[1]=ui->lineEdit\_7;

score\_l[2]=ui->lineEdit\_12; score\_r[2]=ui->lineEdit\_11;

score\_l[3]=ui->lineEdit\_23; score\_r[3]=ui->lineEdit\_20;

score\_l[4]=ui->lineEdit\_17; score\_r[4]=ui->lineEdit\_16;

score\_l[5]=ui->lineEdit\_13; score\_r[5]=ui->lineEdit\_22;

score\_l[6]=ui->lineEdit\_26; score\_r[6]=ui->lineEdit\_28;

score\_l[7]=ui->lineEdit\_32; score\_r[7]=ui->lineEdit\_29;

//кнопки удаления

btn\_del[0]=ui->pushButton\_1;

btn\_del[1]=ui->pushButton\_2;

btn\_del[2]=ui->pushButton\_3;

btn\_del[3]=ui->pushButton\_4;

btn\_del[4]=ui->pushButton\_5;

btn\_del[5]=ui->pushButton\_6;

btn\_del[6]=ui->pushButton\_7;

btn\_del[7]=ui->pushButton\_8;

//еще переменные

NumOfAct=1;

for (int i=0;i<8;i++)

{

Gdel->addButton(*btn\_del[i]*,i);

btn\_del[i]->setIcon(QIcon("C:/qtprojects/ArtProject1/icon1.png"));

btn\_del[i]->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

}

for (int i=1;i<8;i++)

{

team\_l[i]->hide();

score\_l[i]->hide();

score\_r[i]->hide();

team\_r[i]->hide();

btn\_del[i]->hide();

}

ui->menu\_save->setIcon(QIcon("C:/qtprojects/ArtProject1/icon2.png"));

ui->menu\_load->setIcon(QIcon("C:/qtprojects/ArtProject1/icon3.png"));

ui->result1->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

ui->btn1->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

ui->BtnPlus->setStyleSheet("QPushButton { background-color: rgb(244,249,244); }\n");

//connection

connect(ui->menu\_save,&QAction::triggered,

this,&MainWindow::MSave);

connect(ui->menu\_load,&QAction::triggered,

this,&MainWindow::MLoad);

for (int i=0;i<8;i++)

{

connect(btn\_del[i],&QPushButton::clicked,

this,&MainWindow::Idel);

}

connect(ui->BtnPlus,&QPushButton::clicked,

this,&MainWindow::Iplus);

connect(ui->btn1,&QPushButton::clicked,

this,&MainWindow::Oanalis);

connect(ui->result1,&QPushButton::clicked,

this,&MainWindow::Orating);

}

void MainWindow::**Iplus**()

{

if (NumOfAct<8)

{

team\_l[NumOfAct]->show();

score\_l[NumOfAct]->show();

score\_r[NumOfAct]->show();

team\_r[NumOfAct]->show();

btn\_del[NumOfAct]->show();

NumOfAct++;

}

}

void MainWindow::**Idel**()

{

QPushButton\* button = qobject\_cast<QPushButton\*>(*sender()*);

int buttonID = Gdel->id(*button*);

if (NumOfAct!=1)

{

for (int i=buttonID;i<7;i++)

{

team\_l[i]->setText(team\_l[i+1]->text());

score\_l[i]->setText(score\_l[i+1]->text());

score\_r[i]->setText(score\_r[i+1]->text());

team\_r[i]->setText(team\_r[i+1]->text());

}

NumOfAct--;

team\_l[NumOfAct]->setText("");

score\_l[NumOfAct]->setText("");

score\_r[NumOfAct]->setText("");

team\_r[NumOfAct]->setText("");

team\_l[NumOfAct]->hide();

score\_l[NumOfAct]->hide();

score\_r[NumOfAct]->hide();

team\_r[NumOfAct]->hide();

btn\_del[NumOfAct]->hide();

}

else

{

team\_l[0]->setText("");

score\_l[0]->setText("");

score\_r[0]->setText("");

team\_r[0]->setText("");

}

}

bool **comp**(std::tuple<QString,int> a,std::tuple<QString,int> b)

{

return std::get<1>(*a*)>std::get<1>(*b*);

}

void MainWindow::**Orating**()

{

QMap <QString,int> MAP;

QList <std::tuple<QString,int>> LIST;

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString d=team\_r[i]->text();

int b;

int c;

if (a==d && a!="")

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E007\n матч между одинаковыми командами");

return;

}

try

{

b=score\_l[i]->text().toInt();

c=score\_r[i]->text().toInt();

}

catch (...)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E001\n нецелочисленный счет");

return;

}

if (a.isEmpty() && d.isEmpty()) continue;

else if (!a.isEmpty() && !d.isEmpty())

{

if (MAP.find(a)==MAP.end()) MAP.insert(a,b);

else MAP.insert(a,MAP[a]+b);

if (MAP.find(d)==MAP.end()) MAP.insert(d,c);

else MAP.insert(d,MAP[d]+c);

}

else

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E002\n пустое имя команды");

return;

}

}

if (MAP.isEmpty())

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E003\n пустая таблица");

return;

}

for (auto it=MAP.begin();it!=MAP.end();it++)

{

std::tuple<QString,int> TUPLE(it.key(),*it.value()*);

LIST.push\_back(TUPLE);

}

std::sort(LIST.begin(),LIST.end(),comp);

QString text1="Рейтинг\n";

QString text2="🏆\n";

QString text3="⚽\n";

int count=1;

for (auto it=LIST.begin();it!=LIST.end();it++)

{

text1+=std::get<0>(*\*it*)+"\n";

text2+=QString::number(count)+"\n";

text3+=QString::number(std::get<1>(*\*it*))+"\n";

count++;

}

ui->textBrowser\_1->setText(text1);

ui->textBrowser\_2->setText(text2);

ui->textBrowser\_3->setText(text3);

}

void MainWindow::**Oanalis**()

{

int count\_win1=0;

int count\_win2=0;

int count\_win3=0;

int count\_iball=0;

int count\_oball=0;

int count\_games=0;

QString name=ui->ent1->text();

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString d=team\_r[i]->text();

int b;

int c;

if (a==d && a!="")

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E007\n матч между одинаковыми командами");

return;

}

try

{

b=score\_l[i]->text().toInt();

c=score\_r[i]->text().toInt();

}

catch (...)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E001\n нецелочисленный счет");

return;

}

if (a.isEmpty() && d.isEmpty()) continue;

else if (!a.isEmpty() && !d.isEmpty())

{

if (a!=name && d!=name) continue;

count\_games++;

if (d==name)

{

swap(*a*,*d*);

int e=c;

c=b; b=e;

}

if (b==c) count\_win3++;

if (b>c) count\_win1++;

if (b<c) count\_win2++;

count\_iball+=c;

count\_oball+=b;

}

else

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E002\n пустое имя команды");

return;

}

}

QString text1=name+"\n";

QString text2="\n";

QString text3="\n";

if (count\_games==0) text1+="не обнаружено";

else

{

text1=name+"\nматчи\nпобеды\nпоражения\nничьи\nзабитые\nпропущенные";

text2+=QString::number(count\_games)+"\n";

text2+=QString::number(count\_win1)+"\n";

text2+=QString::number(count\_win2)+"\n";

text2+=QString::number(count\_win3)+"\n";

text2+=QString::number(count\_oball)+"\n";

text2+=QString::number(count\_iball)+"\n";

text3+="--\n";

text3+=QString::number(qCeil(100.0\*count\_win1/count\_games))+"%\n";

text3+=QString::number(qCeil(100.0\*count\_win2/count\_games))+"%\n";

text3+=QString::number(qCeil(100.0\*count\_win3/count\_games))+"%\n";

text3+="--\n";

text3+="--\n";

}

ui->textBrowser\_1->setText(text1);

ui->textBrowser\_2->setText(text2);

ui->textBrowser\_3->setText(text3);

}

void MainWindow::**MSave**()

{

QString path = QFileDialog::getSaveFileName(this,

QString::fromUtf8("Сохранить файл"),

QDir::currentPath(),

"Tables (\*.csv);;All files (\*.\*)");

QFile file(path);

if ( !file.*open*(QFile::WriteOnly | QFile::Text) )

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E004\n не удалось отрыть файл");

return;

}

else

{

QTextStream out(*&file*);

for (int i=0;i<8;i++)

{

QString a=team\_l[i]->text();

QString b=score\_l[i]->text();

QString c=score\_r[i]->text();

QString d=team\_r[i]->text();

out<<a<<","<<b<<","<<c<<","<<d<<"\n";

}

file.*close*();

}

ui->textBrowser\_1->setText("сохранено");

}

void MainWindow::**MLoad**()

{

QString path = QFileDialog::getOpenFileName(this,

QString::fromUtf8("Открыть файл"),

QDir::currentPath(),

"Tables (\*.csv);;All files (\*.\*)");

QFile file(path);

if ( !file.*open*(QFile::ReadOnly | QFile::Text) )

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E004\n не удалось отрыть файл");

return;

}

else

{

QTextStream in(*&file*);

int count=0;

while (!in.atEnd())

{

QStringList data = in.readLine().split(",");

if (data.size()!=4)

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E005\n некорректный формат данных");

return;

}

team\_l[count]->setText(data[0]);

score\_l[count]->setText(data[1]);

score\_r[count]->setText(data[2]);

team\_r[count]->setText(data[3]);

team\_l[count]->show();

score\_l[count]->show();

score\_r[count]->show();

team\_r[count]->show();

btn\_del[count]->show();

count++;

if (count>7 && !in.atEnd())

{

QMessageBox::critical(this, "Ошибка","Код ошибки: E006\n критический размер файла");

return;

}

}

file.*close*();

}

NumOfAct=8;

ui->textBrowser\_1->setText("загружено");

}

MainWindow::~***MainWindow***()

{

delete ui;

}

**Скриншот**

