#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QMessageBox>

Tree t;

list e, m, h;

Nodemath\* pe = e.head;

Nodemath\* pm = m.head;

Nodemath\* ph = h.head;

QString ptr;

int round = 0;

int marks = 0;

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

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

ui->pushButton\_5->setEnabled(false);

ui->lineEdit->setEnabled(false);

}

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

{

delete ui;

}

void list::**createEasy**()

{

head->qt = "Easy: What is 1 + 1 ? (answer in integer or fraction)";

head->ans = "2";

point = new Nodemath;

head->next = point;

point->qt = "Easy: What is 2 + 1 ? (answer in integer or fraction)";

point->ans = "3";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: What is 2 + 2 ?(answer in integer or fraction)";

point->ans = "4";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: What is 3 + 2 ? (answer in integer or fraction)";

point->ans = "5";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: What is 3 + 3 ? (answer in integer or fraction)";

point->ans = "6";

point->next = NULL;

point = point->next;

//displayList();

//for checking purposes only

}

void list::**createMed**()

{

head->qt = "Med: What is 5% of Rs.100 ? (answer in integer or fraction)";

head->ans = "5";

point = new Nodemath;

head->next = point;

point->qt = "Med: What is 1 + 4 / 2 \* 5 ? (answer in integer or fraction)";

point->ans = "11";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: What is the ratio in the series 5, 15, 45, 135 (answer in integer or fraction)";

point->ans = "3";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: Two angles of a triangle are 90 and 30. What is the third? (answer in integer or fraction)";

point->ans = "60";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: If two lines are perpendicular to each other the angle between then is? (answer in integer or fraction)";

point->ans = "90";

point->next = NULL;

point = point->next;

}

void list::**createHard**()

{

head->qt = "Hard: Solve for x-> x+y=5, x-y=3 (answer in integer or fraction)";

head->ans = "4";

point = new Nodemath;

head->next = point;

point->qt = "Hard: The ratio in the following series is ? 15, 45, 135, 405... (answer in integer or fraction)";

point->ans = "3";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: If the square of x is y and x is half of y, y is ? (for y>0)? (answer in integer or fraction)";

point->ans = "4";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: If Sara is half as old as her mother and next year her mother will be 25 years older than her, right now Sara is ? (answer in integer or fraction) ";

point->ans = "25";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: The mean of the following data set is ? : 1, 4, 6, 7, 8, 10 (answer in integer or fraction)";

point->ans = "6";

point->next = NULL;

point = point->next;

//displayList();cout<<endl; //for checking purposes only

}

void list::**createEasy1**()

{

head->qt = "Easy: What is the opposite of the word enter?";

head->ans = "exit";

point = new Nodemath;

head->next = point;

point->qt = "Easy: What is opposite of the word close?";

point->ans = "open";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: What is the opposite of the word toddler?";

point->ans = "adult";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: The word wander means to walk slowly and aimlessly (True/False)?";

point->ans = "true";

point->next = new Nodemath;

point = point->next;

point->qt = "Easy: the word fragile means strong (true/false)?";

point->ans = "false";

point->next = NULL;

point = point->next;

//displayList();cout<<endl; //for checking purposes only

}

void list::**createMed1**()

{

head->qt = "Med: She could not go either by bus \_\_\_\_\_\_ by train. \na)no\n b)or\n c)nor\n d)and\n";

head->ans = "or";

point = new Nodemath;

head->next = point;

point->qt = "Med: He \_\_\_\_ take the exam next year.";

point->ans = "will";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: As you \_\_\_, so shall you reap";

point->ans = "sow";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: Don't laugh \_\_\_ the beggars.'";

point->ans = "at";

point->next = new Nodemath;

point = point->next;

point->qt = "Med: could is the past form of \_\_\_";

point->ans = "can";

point->next = NULL;

point = point->next;

//displayList();cout<<endl; //for checking purposes only

}

void list::**createHard1**()

{

head->qt = "Hard: 'The green-eyed monster' means:\na)hatred\nb)love\nc)life\nd)jealousy";

head->ans = "jealousy";

point = new Nodemath;

head->next = point;

point->qt = "Hard: Retype the correct spelling: \na)affedevit\nb)affidavit\nc)afidevit\nd)affidevit";

point->ans = "affidavit";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: A prepositional phrase consists of a preposition and its object.\n (True/False)?";

point->ans = "false";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: Phrases are grammatical units that consist of one or more words.\n (True/False)? ";

point->ans = "true";

point->next = new Nodemath;

point = point->next;

point->qt = "Hard: 'The national anthem is being sung by Jason this time.' \nThe above sentence is Active or Passive?";

point->ans = "passive";

point->next = NULL;

point = point->next;

//displayList();cout<<endl; //for checking purposes only\*/

}

void list::**displayList**()

{

Nodemath\* ptrr = head;

while(ptrr != NULL)

{

//cout<<ptrr->qt<<endl;

ptrr = ptrr->next;

}

}

void Tree::**insert**(int d)

{

leaf\* ptr = root;

leaf\* prev = NULL;

if(root == NULL)

{

root = new leaf;

root->data = d;

return;

}

while(ptr != NULL)

{

prev = ptr;

if(d < ptr->data)

{

ptr = ptr->left;

}

else

{

ptr = ptr->right;

}

}

if(d<prev->data)

{

prev->left = new leaf;

prev->left->data = d;

}

else

{

prev->right = new leaf;

prev->right->data = d;

}

}

Tree::**Tree**() //constructor to initialize a tree with the below mentioned values

{

insert(15);

insert(8);

insert(23);

insert(4);

insert(11);

insert(19);

insert(27);

insert(2);

insert(5);

insert(9);

insert(13);

insert(17);

insert(21);

insert(25);

insert(29);

insert(1);

insert(3);

insert(6);

insert(7);

insert(8);

insert(10);

insert(12);

insert(14);

insert(16);

insert(18);

insert(20);

insert(22);

insert(24);

insert(26);

insert(28);

insert(30);

//displayT();cout<<endl; //for checking purposes only

root->data = 1;

filler(*root*);

}

void Tree::**filler**(leaf\* fill) //function to change values of tree to level 1,2,3

{

if(fill->left == NULL)

{

return;

}

else{

switch(fill->data)

{

case 1:{

fill->left->data = 1;

fill->right->data = 2;

break;

}

case 2:{

fill->left->data = 1;

fill->right->data = 3;

break;

}

case 3:{

fill->left->data = 2;

fill->right->data = 3;

break;

}

}

filler(*fill->left*);

filler(*fill->right*);}

}

void Tree::**displayT**()

{

displayTree(*root*);

return;

}

void Tree::**displayTree**(leaf \*ptr)

{

if(ptr==NULL)

return;

displayTree(*ptr->left*);

//cout<<ptr->data<<" ";

displayTree(*ptr->right*);

}

/\*PROOF THAT THE TREE COULD NOT BE CREATED THROUGH RECURSION :(

void createTree(leaf\* ptr)

{

ptr->left = new leaf;

ptr->right = new leaf;

if(ptr->data == 1)

{

ptr->left->data = 1;

ptr->right->data = 2;

}

else

{

if(ptr->data == 2)

{

ptr->left->data = 1;

ptr->right->data = 3;

}

else

{

if(ptr->data ==3)

{

ptr->left->data = 2;

ptr->right->data = 3;

}

}

}

while(count < 3 )

{

count++;

createTree(ptr->left);

createTree(ptr->right);

}

}

\*/

void MainWindow::**on\_pushButton\_2\_clicked**()

{

ui->pushButton\_3->setEnabled(false);

ui->pushButton\_2->setEnabled(false);

ui->pushButton\_5->setEnabled(true);

ui->label->setText("Math Mode Selected");

t.Pl = t.root;

e.createEasy();

m.createMed();

h.createHard();

}

void MainWindow::**on\_pushButton\_3\_clicked**()

{

ui->pushButton\_2->setEnabled(false);

ui->pushButton\_3->setEnabled(false);

ui->pushButton\_5->setEnabled(true);

ui->label->setText("English Mode Selected");

t.Pl = t.root;

e.createEasy1();

m.createMed1();

h.createHard1();

}

void MainWindow::**on\_pushButton\_5\_clicked**()

{

if(round<1)

{ ui->lineEdit->setEnabled(true);

switch(t.Pl->data)

{

case 1:{

ui->label\_2->setText(pe->qt);

ptr= pe->ans;

pe = pe->next;

break;

}

case 2:{

ui->label\_2->setText(pm->qt);

ptr= pm->ans;

pm = pm->next;

break;

}

case 3:{

ui->label\_2->setText(ph->qt);

ptr= ph->ans;

ph = ph->next;

break;

}

}

}

else{

if(round<5){

ui->lineEdit->setEnabled(true);

QString Answer = ui->lineEdit->text();

if(ptr == Answer)

{

marks++;

t.Pl = t.Pl->right;

}

else

{

t.Pl = t.Pl->left;

}

switch(t.Pl->data)

{

case 1:{

ui->label\_2->setText(pe->qt);

ptr= pe->ans;

pe = pe->next;

break;

}

case 2:{

ui->label\_2->setText(pm->qt);

ptr= pm->ans;

pm = pm->next;

break;

}

case 3:{

ui->label\_2->setText(ph->qt);

ptr= ph->ans;

ph = ph->next;

break;

}

}

}

else{

ui->label->setText("Finished! Total Marks:");

ui->label\_2->setText(QString::number(marks));

}

}

round++;

}

void MainWindow::**on\_pushButton\_4\_clicked**()

{

QMessageBox::information(this, "Help", "Instructions.... For more help consult a teacher.");

}

void MainWindow::**on\_pushButton\_6\_clicked**()

{

ui->lineEdit->clear();

}