**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра РАПС**

Курсовая РАБОТА

**по дисциплине «Программирование и основы алгоритмизации»**

Тема: **Электрооборудование пассажирского вагона**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3406 |  | Бушуева Д.А. |
| Преподаватель |  | Армашев А.А. |

Санкт-Петербург

2024

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 3 |
| 1. | Описание файлов программы | 4 |
| 2.  3. | Блок-схема проекта  Интерфейс программы и описание функций | 5  6 |
| 4. | Заключение | 8 |
| 5. | Список использованных источников | 9 |
| 6. | Приложение 1. Код программы. | 10 |

**введение**

**Цель работы:**

Разработать визуальный интерфейс для базы данных электрооборудования пассажирского вагона.

**Основные задачи:**

Разработка классов и методов, необходимых для работы с компонентами программы и создания методов:

* Чтение из файла;
* Редактирование строки;
* Добавление строки;
* Удаление строки;
* Сортировка;
* Поиск в массиве;
* Сохранение данных.

**Описание файлов программы**

Программа включает в себя четыре модуля:

1. В модуле Unit2.cpp представлен код для работы с графическим интерфейсом. Также в файле определены методы, такие, как чтение из файла, добавление строки в таблицу, редактирование строки в таблице, поиск в массиве, удаление строки из таблицы, сортировка массива, сохранение данных в файл.
2. В модуле Unit2.h представлен код базового и наследуемых классов, определены методы для работы с ними, конструкторы для создания объектов, класс формы и его методы.

**Блок-схема проекта**

Рисунок 1 – Блок-схема проекта

**Интерфейс программы**

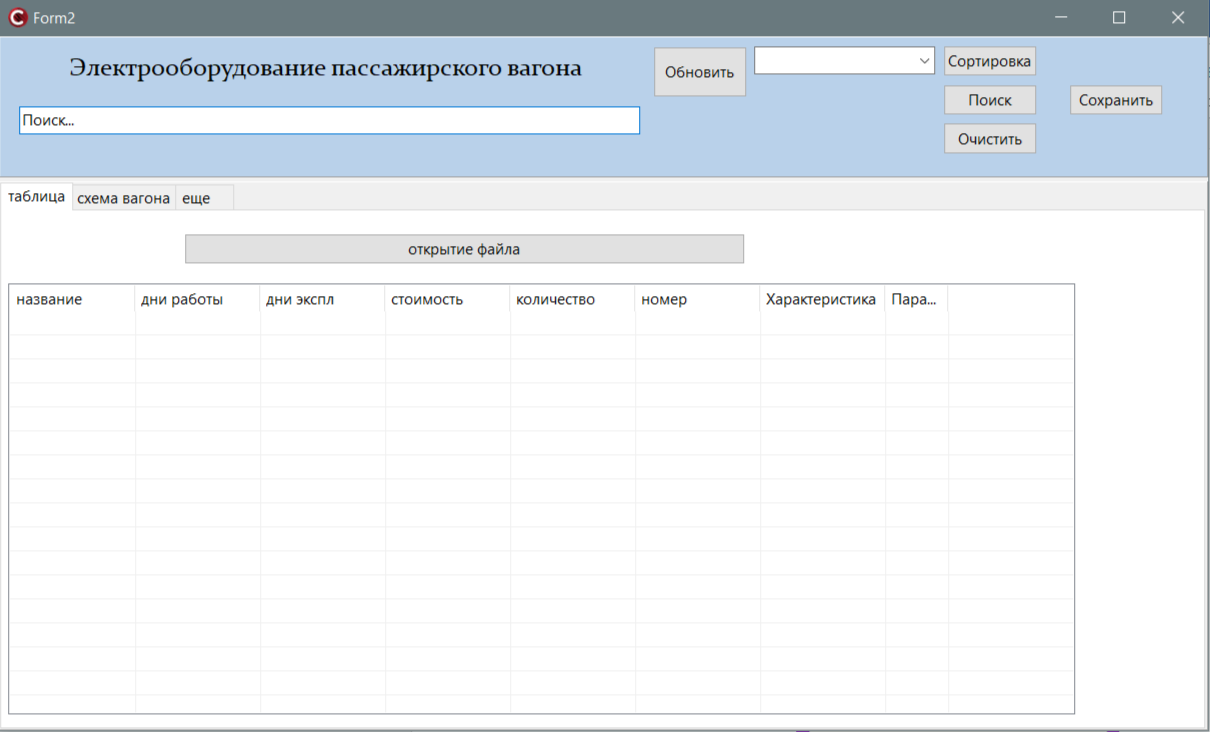
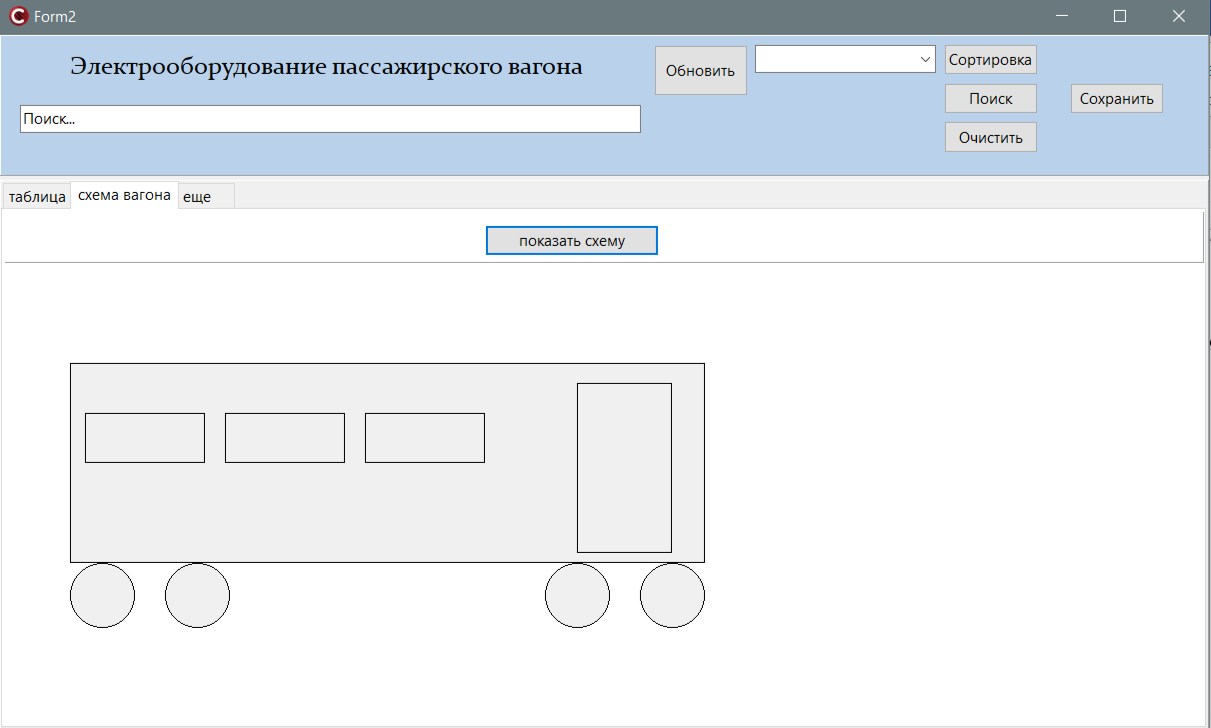


Рисунок 2 – Главная форма проекта



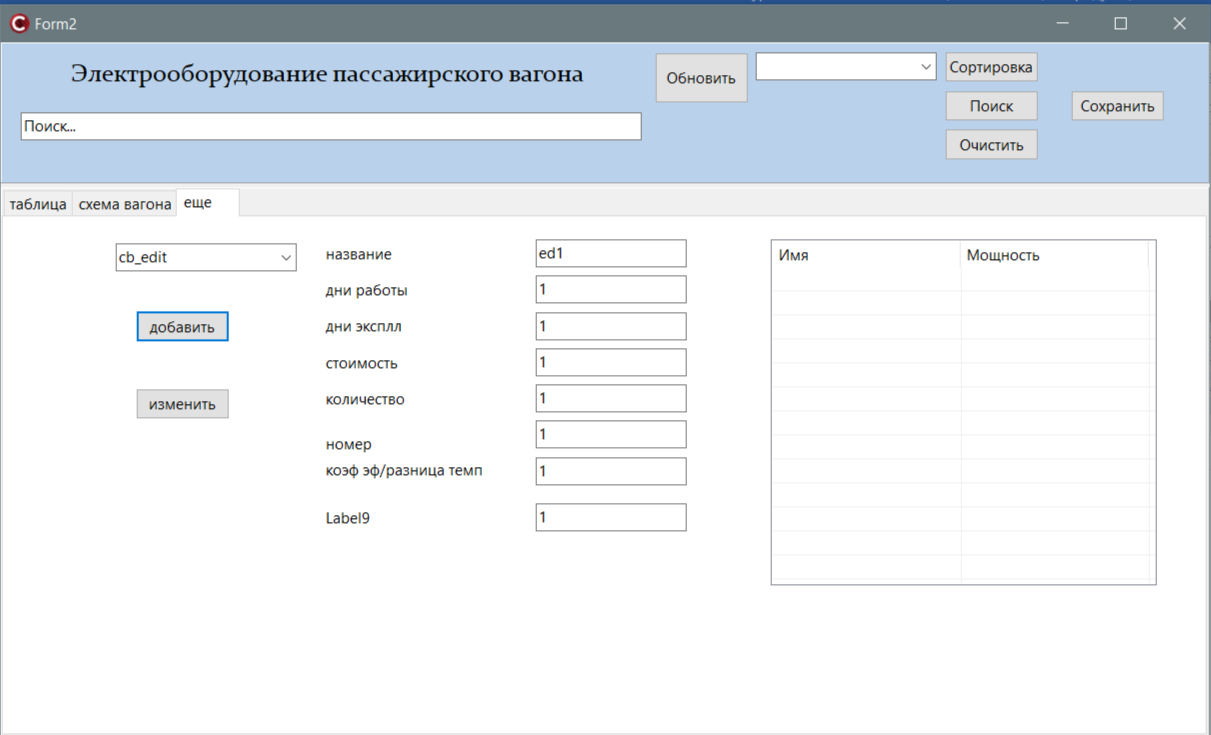


Рисунок 3 – Дополнительные элементы главной формы

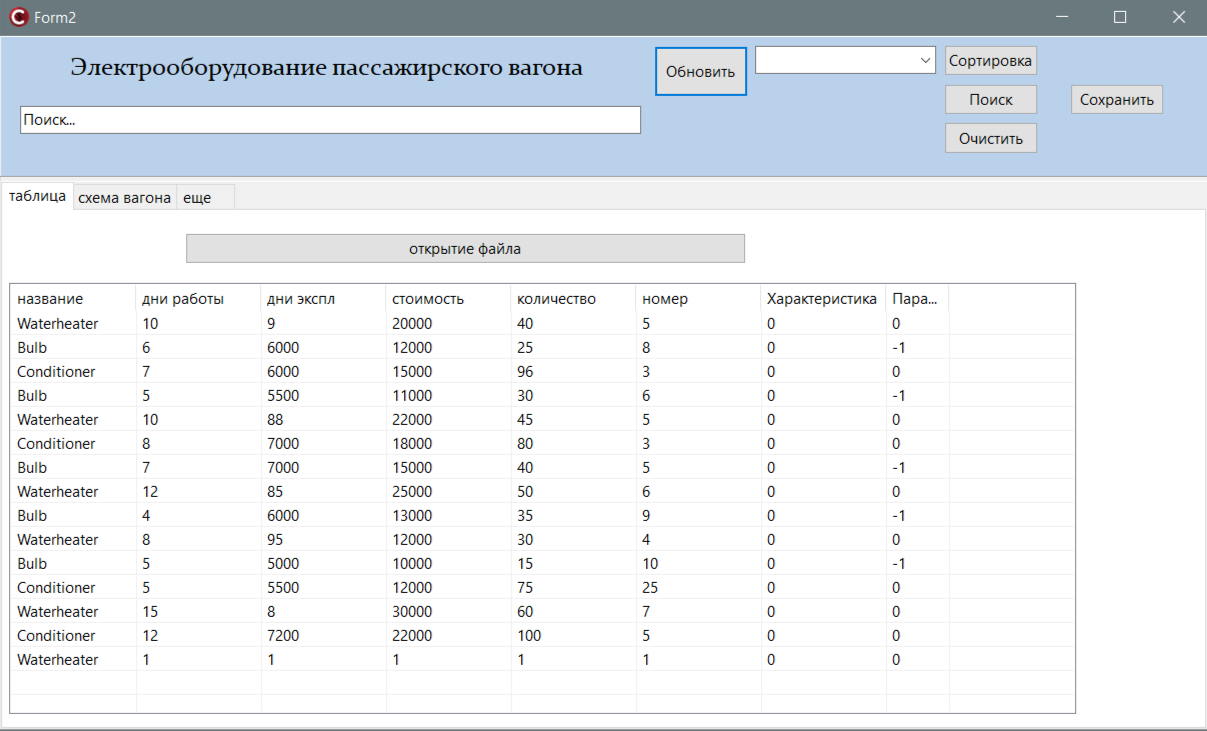


Рисунок 4 – Дополнительная форма

Функция UpdateEl обновляет данные в таблицах.

Функция DeleteElement удаляет строку таблицы.

Функция QuickSort сортирует элементы массива по имени, по стоимости и по времени работы. Используется метод быстрой сортировки.

Функция EditEnter редактирует строку в таблице.

Функция поиска элемента в массиве searchClick.

Функция сохранения элемента SaveElement.

Функция редактирования элемента EditEnter.

**заключение**

В данной работе был разработан интерфейс и программный код интерфейса курсового проекта. Этот интерфейс отражает выполнение всех действий, которые являлись задачами проекта. Таким образом, выполняются функции чтения, удаления, редактирования, рисования, добавления, сортировки массива по названию элемента, его стоимости и времени работы, поиска строки по ее элементу и записи данных в файл.

**список использованных источников**

1. Учебное пособие «Программирование и основы алгоритмизации». А.В. Шевченко, 2018.

**приложение 1. Код программы.**

Unit2.cpp

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

Equip\* Node[30];

int Equip\_count = 0;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button1Click(TObject \*Sender)

{

if (FileOpenDialog1->Execute())

{

AnsiString FileName = FileOpenDialog1->FileName;

FILE\* file = fopen(FileName.c\_str(), "rt");

strcpy( filevirtual , FileName.c\_str());

if (file == NULL)

{

ShowMessage("Не удалось открыть файл :(");

return;

}

char line[1024];

AnsiString allData = ""; // Переменная для накопления данных

while (fgets(line, sizeof(line), file))

{

// Убираем лишние пробелы из строки

AnsiString rawLine = AnsiString(line).Trim();

strcpy(line, rawLine.c\_str());

// Переменные для разбора строки

char type[75], name[75] = "";

int num = 0, cost = 0, count = 0, start\_ex = 0, oper\_time = 0;

float lumens = 0, efficiency = 0, baseEnergy = 0, temperatureDifference = 0;

// Разбор строки

if (strncmp(line, "Bulb", 4) == 0) // Лампа

{

int parsedCount = sscanf(line, " Bulb,%d,%d,%d,%d,%d,%f",

&num, &count, &cost, &start\_ex, &oper\_time, &lumens);

if (parsedCount == 6)

{

Node[Equip\_count++] = new Bulb("Bulb", start\_ex, oper\_time, cost, count, num, lumens);

allData += "Bulb: Num=" + IntToStr(num) + ", Cost=" + IntToStr(cost) +

", Count=" + IntToStr(count) + ", Lumens=" + FloatToStr(lumens) + "\n";

}

else

{

ShowMessage("Ошибка разбора строки: " + rawLine);

}

}

else if (strncmp(line, "Waterheater", 11) == 0) // Водонагреватель

{

int parsedCount = sscanf(line, "Waterheater,%d,%d,%d,%d,%d,%f,%f",

&num, &count, &cost, &start\_ex, &oper\_time, &efficiency, &baseEnergy);

if (parsedCount == 7)

{

Node[Equip\_count++] = new Waterheater("Waterheater", start\_ex, oper\_time, cost, count, num, efficiency, baseEnergy);

allData += "Waterheater: Num=" + IntToStr(num) + ", Cost=" + IntToStr(cost) +

", Efficiency=" + FloatToStr(efficiency) + ", BaseEnergy=" + FloatToStr(baseEnergy) + "\n";

}

else

{

ShowMessage("Ошибка разбора строки: " + rawLine);

}

}

else if (strncmp(line, "Conditioner", 11) == 0) // Кондиционер

{

float temperatureDifferenceF = 0;

int parsedCount = sscanf(line, "Conditioner,%d,%d,%d,%d,%d,%f,%f",

&num, &count, &cost, &start\_ex, &oper\_time, &efficiency, &temperatureDifferenceF);

temperatureDifference = (int)temperatureDifferenceF;

if (parsedCount == 7)

{

Node[Equip\_count++] = new Conditioner("Conditioner", start\_ex, oper\_time, cost, count, num, temperatureDifference, efficiency);

allData += "Conditioner: Num=" + IntToStr(num) + ", Cost=" + IntToStr(cost) +

", TempDiff=" + FloatToStr(temperatureDifference) + ", Efficiency=" + FloatToStr(efficiency) + "\n";

}

else

{

ShowMessage("Ошибка разбора строки: " + rawLine);

}

}

else

{ ShowMessage("Неизвестный тип оборудования: " + rawLine);

}

if (Equip\_count >= 30)

{

ShowMessage("Достигнут предел хранения (30 элементов).");

break;

}

}

fclose(file);

// Выводим все загруженные данные

if (allData.IsEmpty())

{

ShowMessage("Данные не были загружены.");

}

else

{

ShowMessage("Загруженные данные:\n" + allData);

}

}

}

void TForm2::UpdateEl(){

lv\_main->Items->BeginUpdate();

lv\_main->Items->Clear();

lv\_power->Items->BeginUpdate();

lv\_power->Clear();

for(int i=0; i<Equip\_count;i++)

{

TListItem\* item = lv\_main->Items->Add();

item -> Caption =Node[i] -> getName();

item-> SubItems -> Add(Node[i]->getStartEx());

item-> SubItems -> Add(Node[i]->getOperTime());

item-> SubItems -> Add(Node[i]->getCost());

item-> SubItems -> Add(Node[i]->getCount());

item-> SubItems -> Add(Node[i]->getNum());

item-> SubItems -> Add(Node[i]->Characters());

item-> SubItems -> Add(Node[i]->Parameters());

TListItem\* itemm = lv\_power->Items->Add();

itemm-> Caption = Node[i] -> getName();

itemm-> SubItems -> Add(Node[i]->CalculatePower());

}

lv\_main->Items->EndUpdate();

lv\_power->Items->EndUpdate();

}

void TForm2::SaveElement()

{

char str1[]="Bulb";

char str2[]="Waterheater";

char str3[]="Conditioner";

int Buf[5];

double Temp[2];

sscanf( AnsiString(ed2->Text).c\_str() ,"%d",&Buf[0]);

sscanf( AnsiString(ed3->Text).c\_str() ,"%d",&Buf[1]);

sscanf( AnsiString(ed4->Text).c\_str() ,"%d",&Buf[2]);

sscanf( AnsiString(ed5->Text).c\_str() ,"%d",&Buf[3]);

sscanf( AnsiString(ed6->Text).c\_str() ,"%d",&Buf[4]);

sscanf( AnsiString(ed7->Text).c\_str() ,"%lf",&Temp[0]);

sscanf( AnsiString(ed8->Text).c\_str() ,"%lf",&Temp[1]);

switch ( cb\_edit->ItemIndex)

{

case 0:

Node[Equip\_count]= new Bulb(

str1 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0]);

break;

case 1:

Node[Equip\_count]= new Waterheater(

str2 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0],Temp[1]);

break;

case 2:

Node[Equip\_count]= new Conditioner(

str3 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0],Temp[1]);

break;

}

Equip\_count++;

ShowMessage("Успех записи!");

}

void TForm2::DeleteElement(){

int index = lv\_main->ItemIndex;

if (index == -1) {

return;

}

lv\_main->Items->Delete(index);

for (int i = index; i < Equip\_count - 1; i++) {

Node[i] = Node[i + 1];

}

Node[Equip\_count - 1] = NULL;

Equip\_count--;

}

void TForm2::QuickSort(Equip\* node[],int L, int R)

{

int i=L, j=R;

Equip\* x = Node[(L+R)/2];

do {

switch (cb\_sort\_search->ItemIndex)

{

case 0: //имя

{

while (strcmp(Node[i]->getName(), x->getName()) < 0) i++;

while (strcmp(Node[j]->getName(), x->getName()) > 0) j--;

break;

}

case 1: //цена

{

while (Node[i]->getCost() < x->getCost()) i++;

while (Node[j]->getCost() > x->getCost()) j--;

break;

}

case 2: //время

{

while (Node[i]->getStartEx() < x->getStartEx()) i++;

while (Node[j]->getStartEx() > x->getStartEx()) j--;

break;

}

default: {

ShowMessage("Кнопочку забыл.");

break;

}

}

if (i <= j) {

Equip\* buf = Node[i];

Node[i] = Node [j];

Node[j] = buf;

i++;

j--;

}

}while (i <= j);

if(L<j) QuickSort(Node,L,j);

if(i<R) QuickSort(Node,i,R);

}

void TForm2::EnterElement(int i)

{

TListItem\* item = lv\_main->Items->Add();

item -> Caption =Node[i] -> getName();

item-> SubItems -> Add(Node[i]->getStartEx());

item-> SubItems -> Add(Node[i]->getOperTime());

item-> SubItems -> Add(Node[i]->getCost());

item-> SubItems -> Add(Node[i]->getCount());

item-> SubItems -> Add(Node[i]->getNum());

item-> SubItems -> Add(Node[i]->Characters());

}

void TForm2::EditEnter()

{

int index = lv\_main->ItemIndex;

if (index == -1) {

return;

}

ed1->Text = String(Node[index]->getName());

ed2->Text = String(Node[index]->getStartEx());

ed3->Text = String(Node[index]->getOperTime());

ed4->Text = String(Node[index]->getCost());

ed5->Text = String(Node[index]->getCount());

ed6->Text = String(Node[index]->getNum());

ed7->Text = String(Node[index]->Characters());

ed8->Text = String(Node[index]-> Parameters());

}

///////////////////////////////////////////////////////////////////////////

void \_\_fastcall TForm2::bt\_updateClick(TObject \*Sender)

{

UpdateEl();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button2Click(TObject \*Sender)

{

SaveElement();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::lv\_mainDblClick(TObject \*Sender)

{

DeleteElement();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::bt\_searchClick(TObject \*Sender)

{

char buf[52];

strcpy(buf,AnsiString(ed\_search->Text).c\_str());

lv\_main->Items->BeginUpdate();

lv\_main->Items->Clear();

switch (cb\_sort\_search->ItemIndex) {

case 0: {

for (int i = 0; i < Equip\_count; i++) {

if(strcmp(buf,Node[i] -> getName())==0)EnterElement( i);

}

break;

}

case 1: {

int x;

sscanf(buf,"%d",&x);

for (int i = 0; i < Equip\_count; i++) {

if(x== Node[i]->getCost())EnterElement( i);

}

break;

}

case 2: {

int x;

sscanf(buf,"%d",&x);

for (int i = 0; i < Equip\_count; i++) {

if(x==Node[i]->getStartEx())EnterElement( i);

}

break;

}

default: {

ShowMessage("Некорректный индекс поиска.");

break;

}

}

lv\_main->Items->EndUpdate();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::bt\_sortClick(TObject \*Sender)

{

QuickSort(Node , 0,Equip\_count-1);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::bt\_clearClick(TObject \*Sender)

{

lv\_main->Items->Clear();

Equip\_count = 0;

for(int i;i<50;i++)

{

Node[i] == NULL;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button3Click(TObject \*Sender)

{

int index = lv\_main->ItemIndex;

if (index == -1) {

return;

}

char str1[]="Bulb";

char str2[]="Waterheater";

char str3[]="Conditioner";

int Buf[5];

double Temp[2];

sscanf( AnsiString(ed2->Text).c\_str() ,"%d",&Buf[0]);

sscanf( AnsiString(ed3->Text).c\_str() ,"%d",&Buf[1]);

sscanf( AnsiString(ed4->Text).c\_str() ,"%d",&Buf[2]);

sscanf( AnsiString(ed5->Text).c\_str() ,"%d",&Buf[3]);

sscanf( AnsiString(ed6->Text).c\_str() ,"%d",&Buf[4]);

sscanf( AnsiString(ed7->Text).c\_str() ,"%lf", &Temp[0]);

sscanf( AnsiString(ed8->Text).c\_str() ,"%lf", &Temp[1]);

switch ( cb\_edit->ItemIndex)

{

case 0:

Node[index]= new Bulb(

str1 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0]);

break;

case 1:

Node[index]= new Waterheater(

str2 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0],Temp[1]);

break;

case 2:

Node[index]= new Conditioner(

str3 ,Buf[4],Buf[3],Buf[2],Buf[1],Buf[0],Temp[0],Temp[1]);

break;

}

ShowMessage("Успех редактирования!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::lv\_mainKeyPress(TObject \*Sender, System::WideChar &Key)

{

if (Key == VK\_RETURN) {

EditEnter();

Key = 0;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::bt\_savetofileClick(TObject \*Sender)

{

FILE\* file = fopen( filevirtual, "w");

for (int i = 0; i < Equip\_count; i++) {

if(strcmp(Node[i]->getName(),"Bulb")==0) fprintf(file, "%s,%d,%d,%d,%d,%d,%.2lf\n",

Node[i]->getName(),

Node[i]->getNum(),

Node[i]->getCount(),

Node[i]->getCost() ,

Node[i]->getOperTime() ,

Node[i]->getStartEx(),

Node[i]->Characters() );

else fprintf(file, "%s,%d,%d,%d,%d,%d,%.2lf,%.2lf\n",

Node[i]->getName(),

Node[i]->getNum(),

Node[i]->getCount(),

Node[i]->getCost() ,

Node[i]->getOperTime() ,

Node[i]->getStartEx(),

Node[i]->Characters(),

Node[i]->Parameters());

};

fclose(file);

ShowMessage("Запись выполнена успешно");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::PaintBox1Paint(TObject \*Sender)

{

TCanvas\* cs = PaintBox1->Canvas;

cs->Pen->Width = 1;

cs->Pen->Color = clBlack;

cs->Pen->Style = psSolid;

cs->Rectangle(65, 100, 700, 300);

cs->Pen->Width = 1;

cs->Pen->Color = clBlack;

cs->Pen->Style = psSolid;

cs->Rectangle(80, 150, 200, 200);

cs->Pen->Width = 1;

cs->Pen->Color = clBlack;

cs->Pen->Style = psSolid;

cs->Rectangle(220, 150, 340, 200);

cs->Pen->Width = 1;

cs->Pen->Color = clBlack;

cs->Pen->Style = psSolid;

cs->Rectangle(360, 150, 480, 200);

cs->Pen->Width = 1;

cs->Pen->Color = clBlack;

cs->Pen->Style = psSolid;

cs->Rectangle(572.5, 120, 667.5, 290);

cs->Ellipse(65, 300 , 130, 365);

cs->Ellipse(160, 300 , 225, 365);

cs->Ellipse(635, 300 , 700, 365);

cs->Ellipse(540, 300 , 605, 365);

}

//---------------------------------------------------------------------------

Unit2.h

#include <cstring>

#include <stdexcept>

#include <iostream>

#ifndef Unit2H

#define Unit2H

#include <System.Classes.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.Dialogs.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ComCtrls.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Dialogs.hpp>

class Equip {

private:

char name[52]; // Указатель на название

int num; // Номер оборудования

int cost; // Стоимость

int count; // Количество

int start\_ex; // Время ввода в эксплуатацию

int oper\_time; // Время работы в днях

public:

// Сделать доступ к полям наследникам

char\* getName() { return name; }

int getStartEx() const { return start\_ex; }

int getCount() const { return count; }

int getOperTime() const { return oper\_time; }

int getCost() const { return cost; }

Equip( char\* Name, int Num, int Count, int Cost, int Time, int Start)

{

num=Num;

cost = Cost;

count =Count;

oper\_time=Time;

start\_ex=Start;

strcpy(name, Name);

}

int getNum() const { return num; }

virtual double Characters(){return -1;};

virtual double Parameters(){return -1;};

virtual double CalculatePower() {return -1;};

};

class Bulb : public Equip {

private:

double lumens; // Яркость лампочки

public:

Bulb( char\* Name, int Num, int Count, int Cost, int Time, int Start, double Lumens)

: Equip(Name, Num, Count, Cost, Time, Start)

{

lumens=Lumens;

}

double Characters() { return lumens; }

double CalculatePower() {

if (getOperTime() + getStartEx() == 0) {

throw std::runtime\_error("Время работы и эксплуатации не может быть равно нулю.");

}

return static\_cast<double>(lumens \* getCount()) / (getOperTime() + getStartEx());

}

};

class Waterheater : public Equip {

private:

double efficiency; // Коэффициент эффективности

double baseEnergy; // Энергия

public:

Waterheater( char\* Name, int Num, int Count, int Cost, int Time, int Start, double Eff, double Energy)

: Equip(Name, Num, Count, Cost, Time, Start)

{

efficiency= Eff;

baseEnergy= Energy;

}

double Characters() { return efficiency; }

double Parameters(){ return baseEnergy; }

double CalculatePower() {

if (getOperTime() == 0) {

throw std::runtime\_error("Время работы не может быть равно нулю.");

}

return (efficiency \* baseEnergy) / getOperTime();

}

};

class Conditioner : public Equip {

private:

double temperatureDifference; // Разница температур

double efficiency; // Коэффициент эффективности

public:

Conditioner( char\* Name, int Num, int Count, int Cost, int Time, int Start, double TempDiff, double Eff)

: Equip(Name, Num, Count, Cost, Time, Start)

{

temperatureDifference = TempDiff;

efficiency =Eff;

}

double Parameters() { return temperatureDifference; } //сама

double Characters() { return efficiency; } //сама

double CalculatePower() {

if (getOperTime() == 0) {

throw std::runtime\_error("Время работы не может быть равно нулю.");

}

return (getCount() \* getCost() \* temperatureDifference \* efficiency) / getOperTime();

}

};

//---------------------------------------------------------------------------

class TForm2 : public TForm

{

\_\_published: // IDE-managed Components

TPanel \*Panel1;

TSplitter \*Splitter1;

TPanel \*Panel2;

TEdit \*ed\_search;

TPageControl \*PageControl1;

TTabSheet \*TabSheet1;

TTabSheet \*TabSheet2;

TPanel \*Panel3;

TPaintBox \*PaintBox1;

TTabSheet \*TabSheet3;

TListView \*lv\_main;

TFileOpenDialog \*FileOpenDialog1;

TLabel \*Label1;

TButton \*Button2;

TButton \*Button3;

TEdit \*ed1;

TEdit \*ed2;

TEdit \*ed3;

TEdit \*ed4;

TEdit \*ed5;

TEdit \*ed6;

TEdit \*ed7;

TLabel \*Label2;

TLabel \*Label3;

TLabel \*Label4;

TLabel \*Label5;

TLabel \*Label6;

TLabel \*Label7;

TLabel \*Label8;

TButton \*Button5;

TButton \*Button1;

TButton \*bt\_update;

TComboBox \*cb\_edit;

TEdit \*ed8;

TLabel \*Label9;

TComboBox \*cb\_sort\_search;

TButton \*bt\_sort;

TButton \*bt\_search;

TButton \*bt\_clear;

TListView \*lv\_power;

TButton \*bt\_savetofile;

TPopupMenu \*PM\_Paint;

TMenuItem \*N1;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall bt\_updateClick(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

void \_\_fastcall lv\_mainDblClick(TObject \*Sender);

void \_\_fastcall bt\_searchClick(TObject \*Sender);

void \_\_fastcall bt\_sortClick(TObject \*Sender);

void \_\_fastcall bt\_clearClick(TObject \*Sender);

void \_\_fastcall Button3Click(TObject \*Sender);

void \_\_fastcall lv\_mainKeyPress(TObject \*Sender, System::WideChar &Key);

void \_\_fastcall bt\_savetofileClick(TObject \*Sender);

void \_\_fastcall PaintBox1Paint(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TForm2(TComponent\* Owner);

void \_\_fastcall ZapolnenieLiseView();//это добавила может не нужно

Equip\*\* Eqp = new Equip\*[50];

int Equip\_count;

char filevirtual[90];

int clickX; // Координата X клика

int clickY; // Координата Y клика

void UpdateEl();

void SaveElement();

void DeleteElement();

void EnterElement(int i);

void QuickSort(Equip\* node[],int L, int R);

void EditEnter();

};

//---------------------------------------------------------------------------

extern PACKAGE TForm2 \*Form2;

//---------------------------------------------------------------------------

#endif