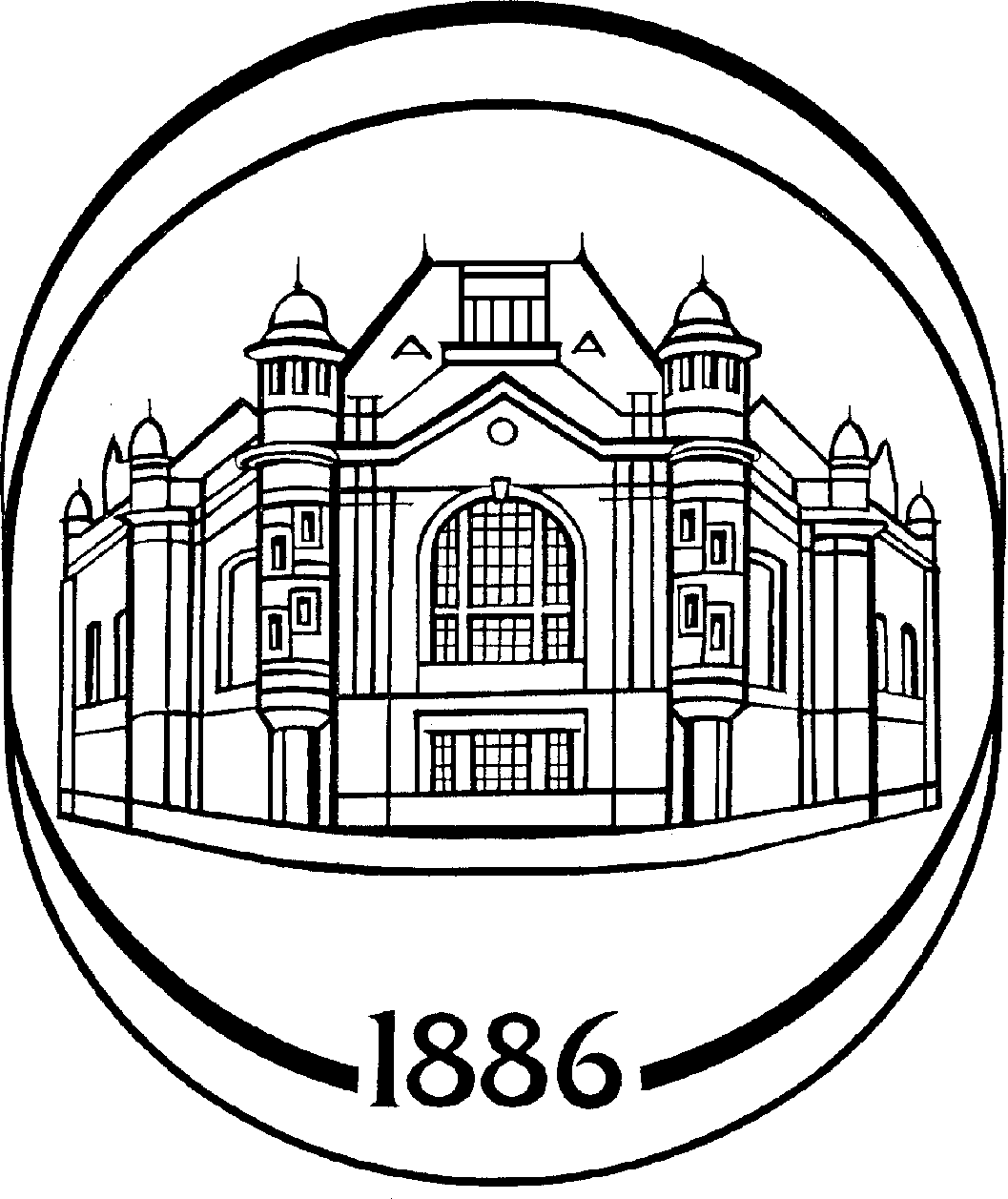
Санкт-Петербургский Государственный Электротехнический Университет

«ЛЭТИ»



Пояснительная записка к курсовой работе

по дисциплине Информационные технологии

Выполнил: Мартыненко Д.А

Факультет: ЭА

Группа: 4401

Компьютер №11

Проверил: Армашев А.А.

Санкт-Петербург 2015

Содержание:

[Цель работы ,задание курсового проекта: 3](#_Toc417330552)

[Пользовательский интерфейс:](#_Toc417330553) 4

[Разработка программного кода:](#_Toc417330554) 5

[Заключение и вывод:](#_Toc417330556) 11

[Список используемой литературы](#_Toc417330556) 12

[Приложение к работе файл 1.сpp](#_Toc417330556) 13

# Цель работы и задание курсового проекта:

**Цель работы:** Выполнение задания, в соответствии с легендой. Отработки п получения программирования и на примере решения задачи «Анализ метеоданных».

**Задание:** Создать программу, для клуба «Метеорологи-любители», с помощью которой участники клуба смогут анализировать метеоданные.

Программа должна обрабатывать метеоданные поступившие с метеозонда (Температура утром, температура днем, температура вечером, температура ночью, влажность утром, влажность днем, влажность вечером, влажность ночью, давление утром, давление днем, давление вечером, давление ночью). Необходимо расшифровать файл, и произвести расчеты среднего значение температуры. Пользователь имеет возможность корректировки файла, и вноса туда дополнительных данных.

Пользовательский интерфейс.

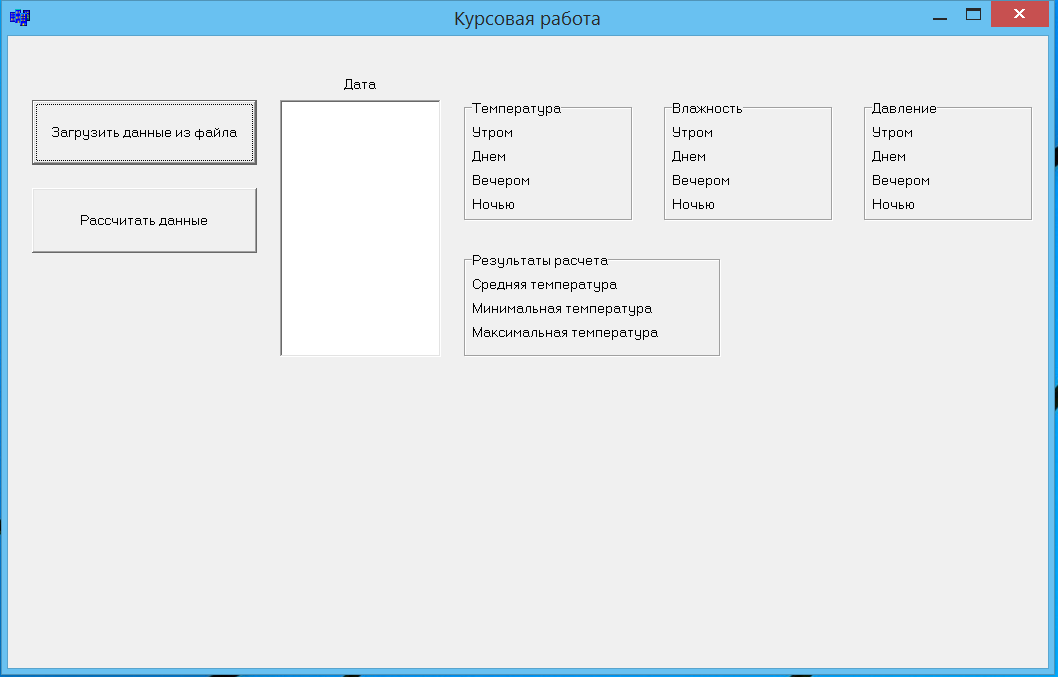


Рисунок 1

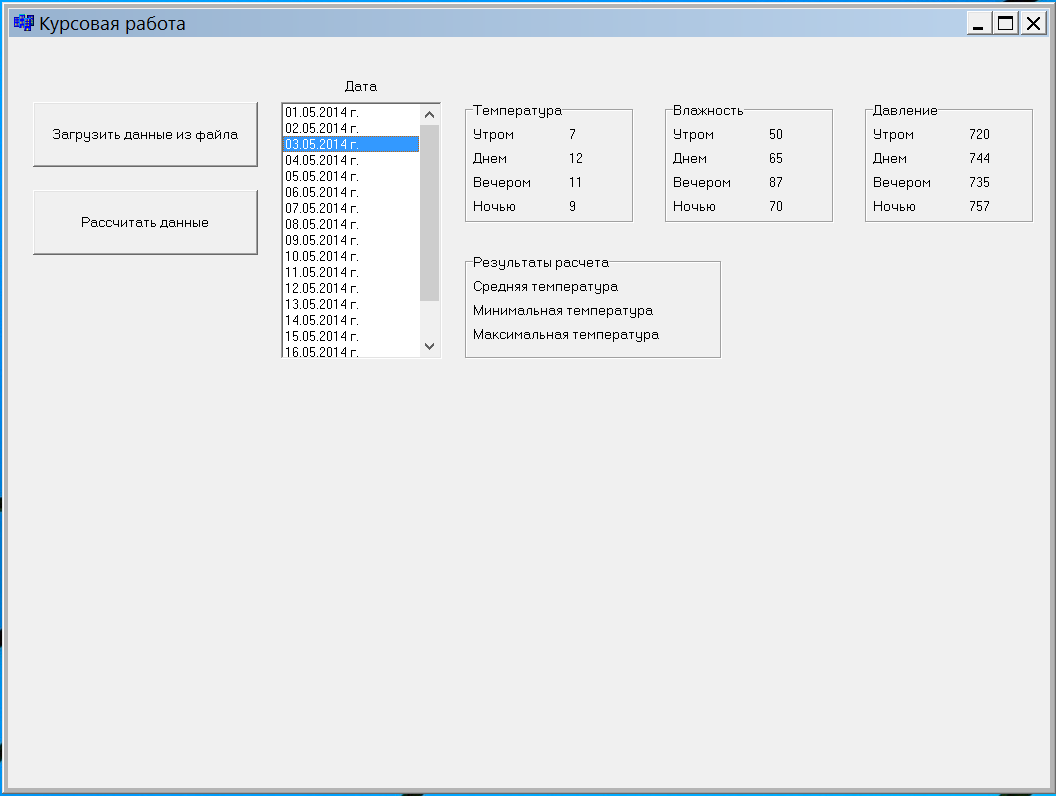


Рисунок 2

Разработка программного кода.

В ходе разработки программного кода, была использована функция чтения данных из файла, и их обработка. Для выбора файла, я использовал компонент ***OpenDialog***. При помощи свойства

***Filer,*** я добавил тип файла .txt, что позволило произвести считывание данного типа файла.

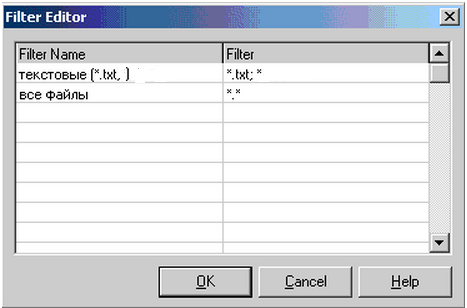


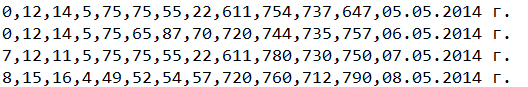
Рисунок 3

Все данные в текстовом файле, введены в строгой последовательности (

*Температура утром, температура днем, температура вечером, температура ночью, влажность утром, влажность днем, влажность вечером, влажность ночью, давление утром, давление днем, давление вечером, давление ночью, время)*

Для дальнейшей обработки.

***Пример***



Значения вводятся через запятую, в порядке, который указан выше. Каждое соответствует элементу структуры.

Для того, что бы считать данные из файла , пришлось написать функцию, которая распознает и переводит эти данные в переменную в структуре типа float. Функция работает по принципу поиску запятой и замены этой заяпятой но 0 ( конец строки ). Это позволяет отделить отдельное значение и перевести его в переменную типа float.

struct Day

{

float TempMorning;

float TempDay;

float TempEvening;

float TempNight;

float VlagMorning;

float VlagDay;

float VlagEvening;

float VlagNight;

float BarMorning;

float BarDay;

float BarEvening;

float BarNight;

char Time [50];

};

Далее, я использовал функцию, открытия файла, и чтения данных из него. Указатель на File является переменной для открытия файла и чтения из него данных. char Str [500] переменная, в которую читается каждая строка (длинной 500 символов) из файла. I- является номером элемента Day в массиве для обрабатываемой строки. Позиции в строках запятой и начало значения (StrPos, StrStart). Функция fopen открывает файл, а c\_str() - фукнция, возвращающая имя файла как указатель на char, для функции fopen. (r – только чтения, как режим работы). if (File == NULL) return 0, возврат ошибки, если файл не был открыт. Так как мы не знаем заранее количество строк, приравниваем переменную DataOfWeatherCount к нулю. while (feof (File) == 0) – функция возвратит нулевое значение, если будет достигнут конец файла.

if (strlen (Str) > 0 && Str [0] != ';') -

Если длина строки больше нуля и начинается не с ; то увеличиваем количество строк с данными.

Память выделяем в размере количества строк\* на размер структуры Day. (DataOfWeather = (struct Day\*) malloc (DataOfWeatherCount \* sizeof (Day));)

int ReadDataFromFile (AnsiString FileName)

{

FILE\* File;

char Str [500];

int i;

char\* StrPos;

char\* StrStart;

File = fopen (FileName.c\_str(), "r");

DataOfWeatherCount = 0;

while (feof (File) == 0)

{

if (fgets (Str, sizeof (Str), File) != NULL)

{

if (strlen (Str) > 0 && Str [0] != ';') DataOfWeatherCount++

}

DataOfWeather = (struct Day\*) malloc (DataOfWeatherCount \* sizeof (Day));

if (DataOfWeather == NULL)

{

fclose (File);

return 0;

}

Далее, заполняем выделенную динамическую память нулями, и возвращаемся в начало файла (rewind (File)), и начинаем отсчет элементов с нуля.

memset (DataOfWeather, 0, DataOfWeatherCount \* sizeof (Day));

rewind (File);

i = 0;

Далее, работаем с чтением и разделением значений из файла (а именно из строк). Если длина строки больше нуля и начинается не с ; то увеличиваем количество строк с данными. Далее ищем запятую в строке (StrPos = strchr (StrStart, ',') и если нашли ее, меняем на 0 (конец строки).

while (feof (File) == 0)

{

if (fgets (Str, sizeof (Str), File) != NULL)

{

if (strlen (Str) > 0 && Str [0] != ';')

{

StrStart = Str;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempMorning = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempDay = StrToFloat (StrStart);

}

………………

Функция применяется дальше для каждого значения в строке.

………………

Функция применяется дальше для каждого значения в строке.

Даллее я работал с элементом ListBox. С помощью данной функции, происходит «занесение даты в ListBox». Удаляем все данные, если они там были, если количество элементов в массиве равно 0, возвращается 0 (ошибка)

int AddDaysInListBox (TListBox\* ListBox)

{

ListBox->Clear();

if (DataOfWeather == NULL) return 0;

if (DataOfWeatherCount == 0) return 0;

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

{

ListBox->AddItem (DataOfWeather [i].Time, NULL);

}

return 1;

}

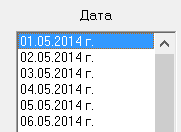


Рисунок 4

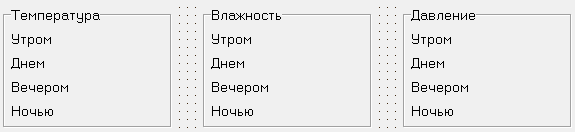


Рисунок 5

Далее, я создал 3 элемента GroupBox, и в каждом по 8 элемента label.

4 элемента видно из скриншота, они служат полем для ввода теста, остальные 4 отображают значения, после двойного клика на дату, из элемента ListBox.

(TListBox\* ListBox = (TListBox\*) Sender ) – Получаем ListBox - объект, с которым связана функция обработки двойного клика.

Переменна типа int - SelectedItem - показывает номер выбранной даты (отсчет начинается с нуля).

void \_\_fastcall TForm1::ListBox1Click(TObject \*Sender)

{

TListBox\* ListBox = (TListBox\*) Sender;

int SelectedItem = ListBox->ItemIndex;

Label6->Caption = FloatToStr (DataOfWeather [SelectedItem].TempMorning);

Label7->Caption = FloatToStr (DataOfWeather [SelectedItem].TempDay);

Label8->Caption = FloatToStr (DataOfWeather [SelectedItem].TempEvening);

Label9->Caption = FloatToStr (DataOfWeather [SelectedItem].TempNight);

return;

}

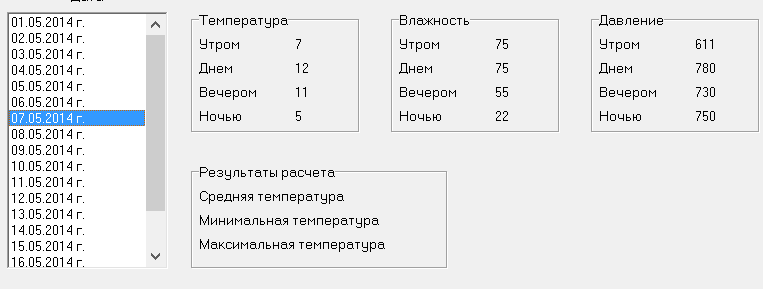


Рисунок 6

Аналогично для расчета средней температуры (GroupBox, Label).

Для расчета максимального значения температуры мы использовали функцию.

Для работы данной функции, максимальное значение температуры заведомо очень маленькое, что бы первое значение, которое мы проверяем, оказалось временно максимальным. Дале, поочередно проверяем все остальные значения.

int SelectedItem = ListBox1->ItemIndex;

float MiddleTemp = (DataOfWeather [SelectedItem].TempMorning + DataOfWeather [SelectedItem].TempDay + DataOfWeather [SelectedItem].TempEvening + DataOfWeather [SelectedItem].TempNight)

float MinTemp = 273;

float MaxTemp = -273;

if (DataOfWeather [SelectedItem].TempMorning > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempMorning;

if (DataOfWeather [SelectedItem].TempDay > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempDay;

if (DataOfWeather [SelectedItem].TempEvening > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempEvening;

if (DataOfWeather [SelectedItem].TempNight > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempNight;

Список используемой литературы

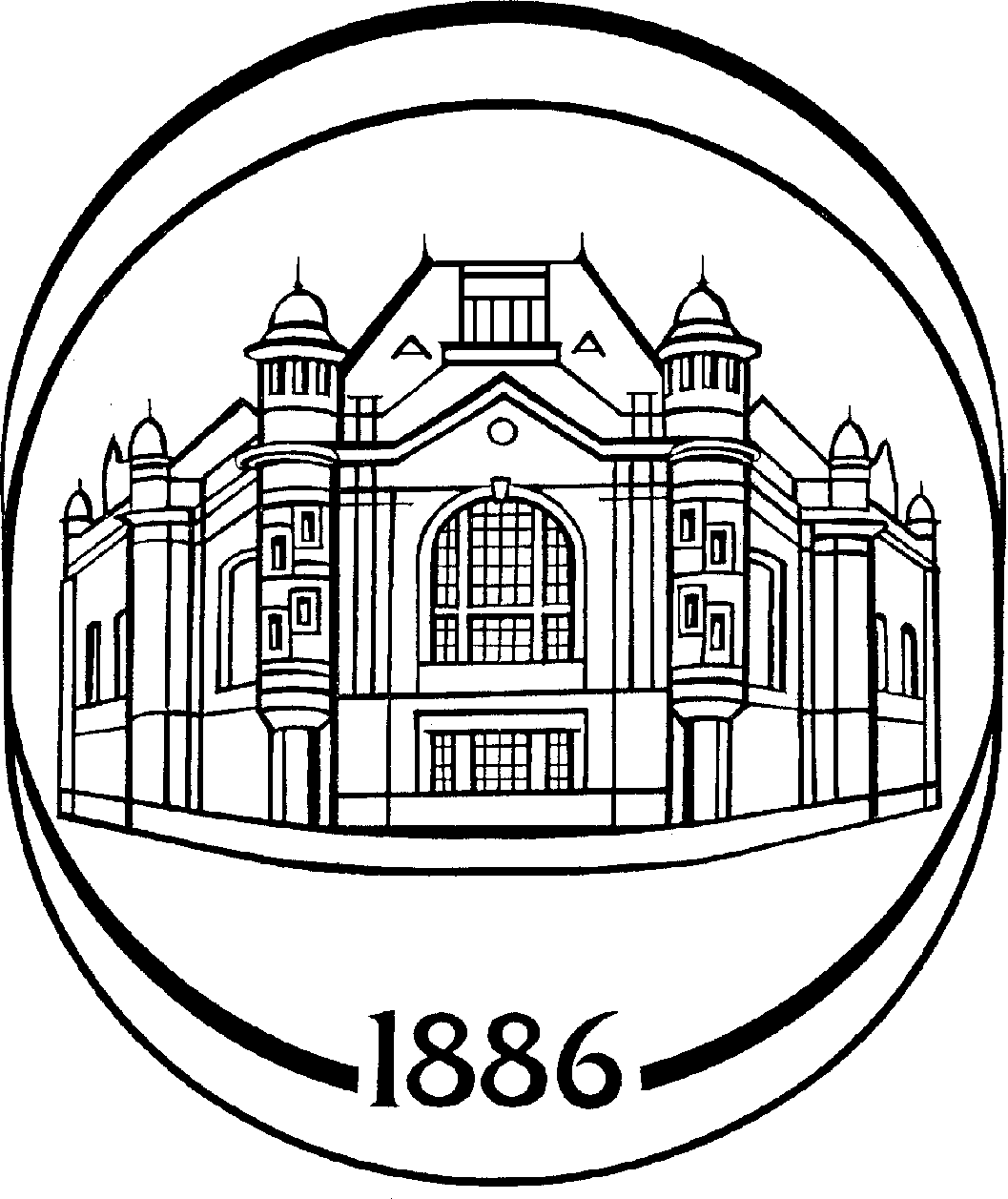
1. Учебное пособие А.В Шевченко «Программирование и основы алгоритмизации»
2. <http://habrahabr.ru/>
3. <http://programmersforum.ru/forumdisplay.php?f=52>
4. <http://www.cyberforum.ru/cpp-builder/>
5. <http://cbuilder.ru/index.php?action=forum>
6. <http://forum.codenet.ru/questions/theme/8>
7. https://www.pokerstarter.org/forum/

**Заключение**: В результате выполнения курсовой работы, я освоил основы работы в C++ Builder 6, а именно: работу с файлами, использование массивов задание и использование переменных различных типов, создание функций и использование указателей, обработчиков и событий. Лабораторные работы, значительно ускорили процесс создания программы. Программа получилась не сложно в использовании и интуитивно понятной.

**Вывод:** В процессе работы гораздо удобнее задавать переменным имена, которые будут легко читаться, и что-то обозначать, например, TempMorning –температура утром. Это позволяет легче обращаться к ней, и не запутаться в обработке большого количества данных. Для считывая информации со строки, и перевода ее в тип данных float, я использовал StrToFl, и для обратного перевода FloatToStr. Данные функции, мне показались удобнее функции cscanf. По легенде, программа предназначаюсь для метеорологов любителей, я думаю, что она бы, удовлетворила потребности заказчика.

Санкт-Петербургский Государственный Электротехнический Университет

«ЛЭТИ»



Приложение к курсовой работе

по дисциплине Информационные технологии

Выполнил: Мартыненко Д.А

Факультет: ЭА

Группа: 4401

Компьютер №11

Проверил: Армашев А.А.

Санкт-Петербург 2015

#include <vcl.h>

#include <stdio.h>

#include <stdlib.h> #include <string.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

struct Day

{

float TempMorning;

float TempDay;

float TempEvening;

float TempNight;

float VlagMorning;

float VlagDay;

float VlagEvening;

float VlagNight;

float BarMorning;

float BarDay;

float BarEvening;

float BarNight;

char Time [50];

};

struct Day\* DataOfWeather = NULL;

int DataOfWeatherCount;

int ReadDataFromFile (AnsiString FileName)

{

FILE\* File; данных из него

char Str [500];

int i;

char\* StrPos;

char\* StrStart;

File = fopen (FileName.c\_str(), "r");

if (File == NULL) return 0;

DataOfWeatherCount = 0;

while (feof (File) == 0)

{

if (fgets (Str, sizeof (Str), File) != NULL)

{

if (strlen (Str) > 0 && Str [0] != ';')DataOfWeatherCount++;

}

}

DataOfWeather = (struct Day\*) malloc (DataOfWeatherCount \* sizeof

Day));

if (DataOfWeather == NULL)

{

fclose (File);

return 0;

}

memset (DataOfWeather, 0, DataOfWeatherCount \* sizeof (Day));

rewind (File);

i = 0;

while (feof (File) == 0)

{

if (fgets (Str, sizeof (Str), File) != NULL)

{

if (strlen (Str) > 0 && Str [0] != ';')

{

StrStart = Str;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempMorning = StrToFl (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempEvening=StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempNight = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].TempNight = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].VlagDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].VlagDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].VlagNight = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].VlagDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].BarDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].VlagDay = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

StrPos = strchr (StrStart, ',');

if (StrPos != NULL)

{

\*StrPos = 0;

DataOfWeather [i].BarNight = StrToFloat (StrStart);

}

StrStart = StrPos + 1;

strncpy (DataOfWeather [i].Time, StrStart, sizeof (DataOfWeather [i].Time) - 1);

i++;

}

}

}

fclose (File);

return 1;

}

int AddDaysInListBox (TListBox\* ListBox)

{

ListBox->Clear();

if (DataOfWeather == NULL) return 0;

if (DataOfWeatherCount == 0) return 0;

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

{

ListBox->AddItem (DataOfWeather [i].Time, NULL);

}

return 1;

}

void \_\_fastcall TForm1::ListBox1Click(TObject \*Sender)

{

TListBox\* ListBox = (TListBox\*) Sender;

int SelectedItem = ListBox->ItemIndex;

Label6->Caption = FloatToStr (DataOfWeather SelectedItem].TempMorning);

Label7->Caption = FloatToStr (DataOfWeather [SelectedItem].TempDay);

Label8->Caption = FloatToStr (DataOfWeather SelectedItem].TempEvening);

Label9->Caption = FloatToStr (DataOfWeather [SelectedItem].TempNight);

Label14->Caption = FloatToStr(DataOfWeather[SelectedItem].VlagMorning);

Label15->Caption = FloatToStr (DataOfWeather [SelectedItem].VlagDay);

Label16->Caption = FloatToStr (DataOfWeather[SelectedItem].VlagEvening);

Label17->Caption = FloatToStr (DataOfWeather [SelectedItem].VlagNight);

Label22->Caption = FloatToStr (DataOfWeather [SelectedItem].BarMorning);

Label23->Caption = FloatToStr (DataOfWeather [SelectedItem].BarDay);

Label24->Caption = FloatToStr (DataOfWeather [SelectedItem].BarEvening);

Label25->Caption = FloatToStr (DataOfWeather [SelectedItem].BarNight);

return;

}

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

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

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

AnsiString FileName;

OpenDialog1->Execute();

FileName = OpenDialog1->FileName;

if (ReadDataFromFile (FileName) == 0)

MessageBox (NULL, "Ошибка чтения из файла!", "Ошибка", MB\_OK | B\_ICONERROR);

return;

}

if (AddDaysInListBox (ListBox1) == 0)

{

MessageBox (NULL, "Ошибка добавления данных в istBox!","Ошибка",

MB\_OK | MB\_ICONERROR);

}

}

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

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

if (ListBox1->Count == 0)

{

MessageBox (NULL, "Данные не загружены!", "Информация", MB\_OK);

return;

}

if (ListBox1->ItemIndex == -1)

{

MessageBox (NULL, "Дата не выбрана!", "Информация", MB\_OK);

return;

}

int SelectedItem = ListBox1->ItemIndex;

float MiddleTemp = (DataOfWeather [SelectedItem].TempMorning + DataOfWeather [SelectedItem].TempDay + DataOfWeather [SelectedItem].TempEvening + DataOfWeather [SelectedItem].TempNight) / 4;

float MinTemp = 273;

float MaxTemp = -273;

if (DataOfWeather [SelectedItem].TempMorning > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempMorning;

if (DataOfWeather [SelectedItem].TempDay > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempDay;

if (DataOfWeather [SelectedItem].TempEvening > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempEvening;

if (DataOfWeather [SelectedItem].TempNight > MaxTemp) MaxTemp = DataOfWeather [SelectedItem].TempNight;

if (DataOfWeather [SelectedItem].TempMorning < MinTemp) MinTemp = DataOfWeather [SelectedItem].TempMorning;

if (DataOfWeather [SelectedItem].TempDay < MinTemp) MinTemp = DataOfWeather [SelectedItem].TempDay;

if (DataOfWeather [SelectedItem].TempEvening < MinTemp) MinTemp = DataOfWeather [SelectedItem].TempEvening;

if (DataOfWeather [SelectedItem].TempNight < MinTemp) MinTemp = DataOfWeather [SelectedItem].TempNight;

Label29->Caption = FloatToStr (MiddleTemp);

Label30->Caption = FloatToStr (MinTemp);

Label31->Caption = FloatToStr (MaxTemp);

return; // Выход из функции

}

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