**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ**

**КАФЕДРА**

КОМПЬЮТЕРНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

***Курсовая работа***

***по дисциплине***

**«Системное программирование»**

**2006**

**Аннотация**

Курсовая работа по дисциплине «Системное программирование» на тему: «Программирование Windows – приложений» содержит \_\_\_ страниц машинописного (рукописного) текста, \_\_ рисунков, \_\_ таблиц, \_\_\_страниц приложения.

В программе рассмотрена программа вычисления значения определённого интеграла.

**Вступление**

Вначале я хотел бы поговорить о ОС, а именно о Windows 2000.

Windows 2000 – операционная система (ОС) фирмы Microsoft, основанная на технологии Windows NT 5.0. NT или New Technology была создана группой разработчиков под руководством Дейва Катлера, ранее работавшего в DEC над проектом VMS. Дейв Катлер пришел в Microsoft в 1988 году специально для работы над проектом NT. NT, в отличии остальных ОС Microsoft.

Windows 2000 – полностью 32-разрядная ОС с приоритетной многозадачностью и улучшенной реализацией работы с памятью. В основе проекта W2k лежат те же принципы, которые когда-то обеспечили успех NT:

* Совместимость. Система имеет привычный интерфейс ОС семейства Windows, поддержку файловых систем NTFS5,NTFS4,FAT16,FAT32. Большинство приложений написанных под MS-DOS, NT4, а также некоторые программы под OS/2 запускаются и функционируют без проблем. При проектировании NT учитывалась возможность работы системы в различных сетевых средах, поэтому в поставку входят средства для работы в UNIX- и NOVELL –сетях.
* Переносимость. Система работает на различных процессорах семейства х86 производства INTEL и AMD. Реализация поддержки процессоров других архитектур возможна, но потребует некоторых усилий.
* Масштабируемость. В W2k реализована поддержка технологий SMP и COW. Количество процессоров при использовании SMP может достигать 32.
* Система безопасности полностью удовлетворяет спецификации С2 по терминологии АНБ США. Правда, для полной поддержки потребуется оборудование, которое также удовлетворяет этой спецификации.
* Распределенная обработка. W2k имеет встроенные в систему сетевые возможности, что обеспечивает возможность связи с различными типами компьютеров-хостов благодаря наличию разнообразных транспортных протоколов и технологий «клиент-сервер».
* Надежность и отказоустойчивость. Архитектура OC защищает приложения от повреждения друг с другом и самой операционной системой. При этом используется отказоустойчивая структурированная обработка особых ситуаций на всех архитектурных уровнях, которая включает восстанавливаемую файловую систему NTFS и обеспечивает защиту с помощью встроенной системы безопасности и усовершенствованных методов управления памятью.
* Локализация. Система предоставляет возможности для работы во многих странах мира на национальных языках, что достигается применении стандарта ISO Unicode.

**1.ТЕОРИТИЧЕСКИЕ АСПЕКТЫ ПРОГРАММЫ**

**1.1 Краткие теоретические сведения**

В задании необходимо вычислить определенный интеграл 1) по формуле трапеций с тремя десятичными знаками и вычислить определенный интеграл 2) по формуле Симпсона при n=50.

Криволинейной трапецией называют часть плоскости, ограниченную снизу Ох, а сверху дугой ab (некоторой линией, уравнением которой является y=f(x)), слева и справа прямыми x=a и x=b.

Пусть нам дана криволинейная трапеция. Требуется вычислить площадь криволинейной трапеции.

Участок b-a разбиваем на n отрезков. Элементарная площадь Si=f(Si)\*(b-a)/n.

Площадью криволинейной трапеции S называют предел:

****

когда n стремится к бесконечности так, что наибольший из участков разбиения стремится к нулю.

Рассмотрим функцию f(x), определенную и непрерывную на заданном отрезке ab.

Построим для этой функции сумму вида:

****

Эту сумму будем называть n-й интегральной суммой, построенной для функции f(x) на отрезке ab.

Если существует предел n-й интегральной суммы указанного вида при n стремящемся к бесконечности, так что наибольший из участков разбиения стремится к нулю, то то этот предел назы**вают определенным интегралом функции f(x), на отрезке ab и обозначают:**

****

а,b - нижний и верхний пределы интегрирования.

Основные свойства определенного интеграла

1) Интеграл от суммы равен сумме интегралов (конечное число слагаемых)



2)Постоянный множитель можно выносить за знак постоянного интеграла:

****

3)Если поменять местами пределы интегрирования, знак интеграла измениться на противоположный:

****

**4)Какие бы ни были числа a,b,c:**

****

**5)Свойство об оценке определенного интеграла:**

**Если м и М соответственно наименьшее и наибольшее значения функции f(x) на ab, то справедлива следующая оценка:**

****

Вычисление методом трапеций

Для того, чтобы вычислить интеграл 1) по методу трапеций, необходимо определить число n – частей, на которые необходимо разбить криволинейную трапецию, чтобы достичь требуемой точности (три знака после запятой).

Предположим, есть некая криволинейная трапеция. Разбиваем криволинейную трапецию сначала на произвольное число частей n, то есть получаем n1- обыкновенных трапеций. Рассчитываем суммарную площадь трапеций (S1). Далее разбиваем криволинейную трапецию на n2>n1 частей и также рассчитываем суммарную площадь трапеций (S2). Следующий шаг – вычисление разности S2-S1. Eсли |

S2-S1|<=0.001, то вычисления можно прервать и взять за искомую площадь (значение интеграла) площадь S

2. Если |S2-S1|>0.001, то S1:=S2, N1:=N2,N2:=N2\*2, до тех пор, пока не будет достигнута требуемая точность.

Расчетная формула метода трапеций:

****

где

Y-функция интеграла;

A-нижний предел интеграла;

B- верхний предел интеграла;

H-точность ;

N-количество итераций;

Расчетная формула метода Симпсона:

S=(h/3)\*(y0+4\*(y1+y3+…+yn-1)+2\*(y2+y4+…+yn-2)+yn)

S-результат вычисления;

h-точность ;

y0-начальное значение полученное от нижнего предела;

yn-значение полученное от верхнего п**редела;**

**1.2 Среда программирования**

При выборе программного средства разработки своей программы я остановился на DELHI. Почему?

Для начала истории.

Delphi - это греческий город, где жил дельфийский оракул. И этим именем был назван новый программный продукт с феноменальными характеристиками. Шесть месяцев назад компания Borland представила на суд программистской общественности новый программный продукт, о котором к моменту его выхода ходило множество слухов. Первая версия продукта явилась результатом разработки, которая велась компанией в обстановке строжайшей секретности в течение двух с половиной лет. Компилятор, встроенный в Delphi, обеспечивает высокую производительность, необходимую для построения приложений в архитектуре "клиент-сервер". Этот компилятор в настоящее время является самым быстрым в мире, его скорость компиляции составляет свыше 120 тысяч строк в минуту на компьютере 486DX33. Он предлагает легкость разработки и быстрое время проверки готового программного блока, характерного для языков четвертого поколения (4GL) и в то же время обеспечивает качество кода, характерного для компилятора 3GL. Кроме того, Delphi обеспечивает быструю разработку без необходимости писать вставки на Си или ручного написания кода (хотя это возможно).

В процессе построения приложения разработчик выбирает из палитры компонент готовые компоненты как художник, делающий крупные мазки кистью. Еще до компиляции он видит результаты своей работы - после подключения к источнику данных их можно видеть отображенными на форме, можно перемещаться по данным, представлять их в том или ином виде. В этом смысле проектирование в Delphi мало чем отличается от проектирования в интерпретирующей среде, однако после выполнения компиляции мы получаем код, который исполняется в 10-20 раз быстрее, чем то же самое, сделанное при помощи интерпретатора. Кроме того, компилятор компилятору рознь, в Delphi компиляция производится непосредственно в родной машинный код, в то время как существуют компиляторы, превращающие программу в так называемый p-код, который затем интерпретируется виртуальной p-машиной. Это не может не сказаться на фактическом быстродействии готового приложения.

## Объектно-ориентированная модель программных компонент

Основной упор этой модели в Delphi делается на максимальном реиспользовании кода. Это позволяет разработчикам строить приложения весьма быстро из заранее подготовленных объектов, а также дает им возможность создавать свои собственные объекты для среды Delphi. Никаких ограничений по типам объектов, которые могут создавать разработчики, не существует. Действительно, все в Delphi написано на нем же, поэтому разработчики имеют доступ к тем же объектам и инструментам, которые использовались для создания среды разработки. В результате нет никакой разницы между объектами, поставляемыми Borland или третьими фирмами, и объектами, которые вы можете создать.

В стандартную поставку Delphi входят основные объекты, которые образуют удачно подобранную иерархию из 270 базовых классов. Для начала - неплохо. Но если возникнет необходимость в решении какой-то специфической проблемы на Delphi, советуем, прежде чем попытаться начинать решать проблему "с нуля", просмотреть список свободно распространяемых или коммерческих компонент, разработанных третьими фирмами, количество этих фирм в настоящее время превышает число 250, хотя, возможно, я не обо всех знаю. Скептики, возможно, не поверят мне, когда я скажу, что на Delphi можно одинаково хорошо писать как приложения к корпоративным базам данных, так и, к примеру, игровые программы. Тем не менее, это так. Во многом это объясняется тем, что традиционно в среде Windows было достаточно сложно реализовывать пользовательский интерфейс. Событийная модель в Windows всегда была сложна для понимания и отладки. Но именно разработка интерфейса в Delphi является самой простой задачей для программиста.

**Классификация версий DELPHI**

Первая версия Delphi обладала набором возможностей, которые немедленно привлекли мое пристальное внимание. Покорившими меня особенностями Delphi были ее идеология форм объектно – ориентированный подход, необычайно быстрый компилятор, прекрасные инструменты для работы с базами данных, тесная интеграция с программированием в среде Windows и технология компонентов. Но самой важной частью был язык OBJECT PASCAL, на фундаменте которого строилось все остальное.

Версия Delphi 2 была еще лучше! Среди ее наиболее важных дополнений были следующие: компонент TBDCtrlGrid для работы с несколькими записями одновременно, улучшенная сетка баз данных,поддержка автоматизации OLE и тип данных вариант, тесная интеграция с Windows 95, тип данных long string и наследование форм. В Delphi 3 к этому добавились технология Code insight (система подсказок для написания кода ), возможность отладки DLL, шаблоны компонентов, TeeChart (библиотека компонентов для работы с графиками), Decision Сube (метакуб –многомерная модель данных),технология Web Broker (компоненты для программирования в интрасетях ), пакеты компонентов, ActiveForms (формы для разработки элементов ActiveX ) и замечательная интеграция с COM основанная на применении интерфейсов.

В Delphi 4 появился редактор AppBrowser, поддержка новых возможностей Windows 98, улучшенная поддержка OLE и COM, расширенные компоненты баз данных и множество добавлений к базовым классам VCL, включая поддержку фиксации ограничений и привязки элементов управления. В Delphi 5 к общей картине добавилось множество других улучшений IDE, расширенная поддержка баз данных, улучшенная версия MIDAS с поддержкой Интернета, инструмент управления версиями TeamSource, возможности лингвистического перевода, концепция фреймов, большое количество новых компонентов.

Delphi 6 добавляет ко всем уже существующим в Delphi возможностям поддержку так называемой кросс-платформенной разработки приложений (или, говоря иначе, возможность разработки приложений для платформы Х ), которая базируется на использовании новой библиотеки компонентов, расширенную библиотеку времени исполнения, новый базовый механизм работы с базами данных dbExpress, поддержку Web –служб и XML, мощную инфраструктуру разработки приложений для Web, новые улучшения IDE, а также огромное количество новых компонентов и классов.

Delphi – прекрасный инструмент разработки, но в то же время и сложная программная среда, состоящая из многих элементов.

**2. ПРАКТИЧЕСКИЕ АСПЕКТЫ ПРОГРАММЫ**

**2.1 Функционально-структурная схема**

Блок программы

Подключение DLL-библиотеки

Вызов вспомогательных форм

Главная форма

Изменение параметров интеграла

Расчет интеграла

Прорисовка графиков

Вызов функций Windows API

Выход

Рис. 1- Функционально-структурная схема

* 1. **Иерархия форм**

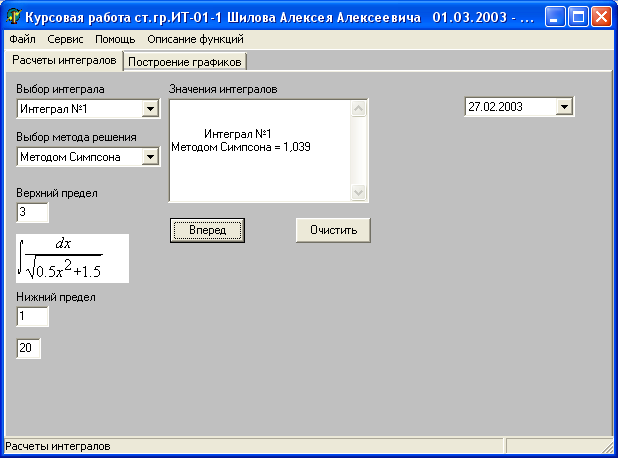


Рис.2 - Главная форма

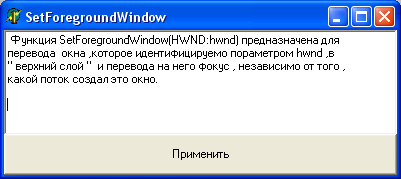


Рис. 3– API –функции

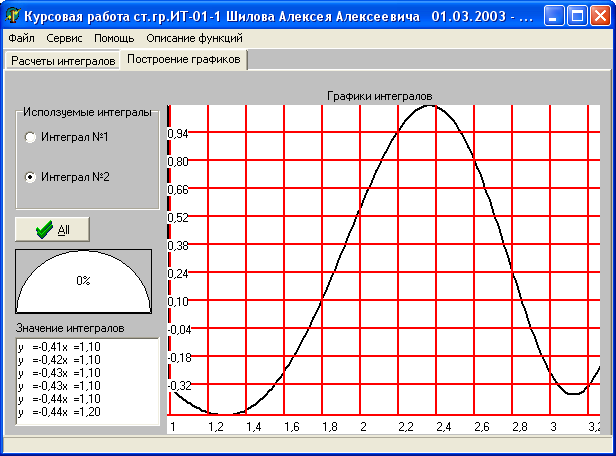


Рис.4 – Главная форма (прорисовка графиков)

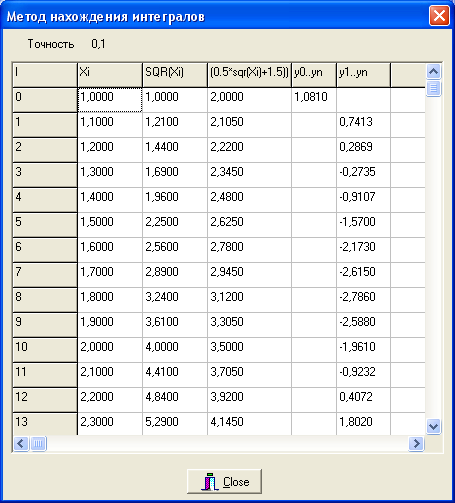


Рис.5 -Наглядный метод решения

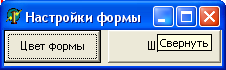


Рис.6 - Настройки

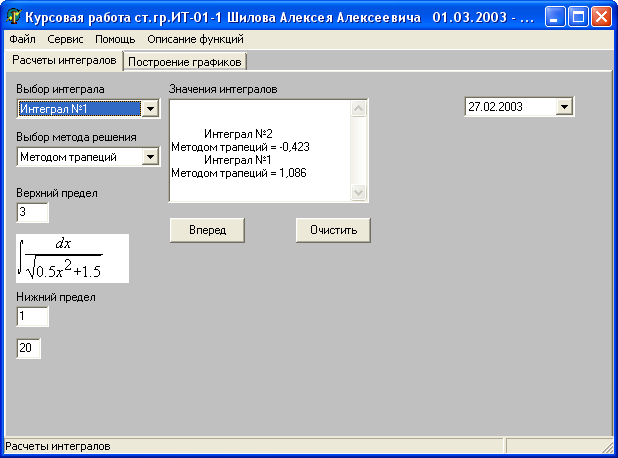


Рис 7 – Главная форма для вычисления интеграла

Работа с программой начинается с этой формы.

На этой форме вы можете рассчитать интеграл:

1. Выбрать интеграл

2. Выбрать метод (трапеций, Симпсона)

3. Ввести верхний предел интеграла

4. Ввести нижний предел интеграла

5. Ввести число разбиений

6. Нажав клавишу «Вперед» вычислить значение интеграла

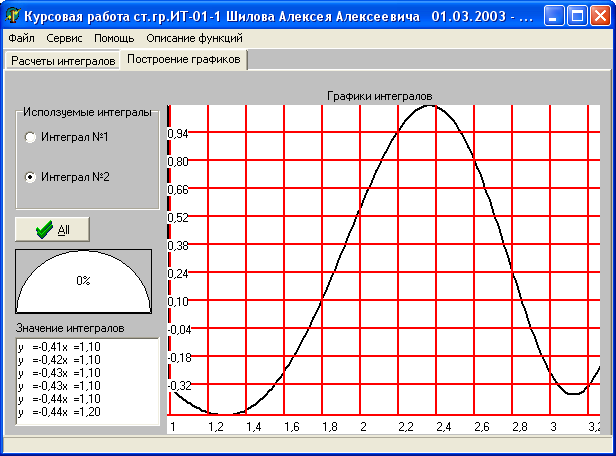


Рис 8 – Форма для прорисовки графиков

На этой форме можно получить значения интегралов в зависимости от значений нижней границы при фиксированных значениях верхней границы определенного интеграла и построить графики

1. Выбрать интеграл

2. Нажать клавишу «График» и получить график выбранного вами интеграла

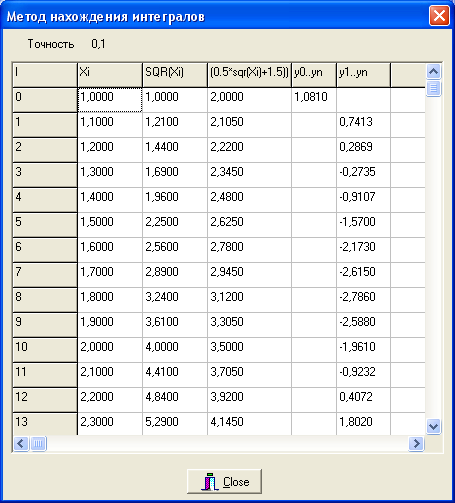


Рис 9 - Таблицы

На этой форме показан более детально метод нахождения интегралов.

**Форма для применения функции Windows API SetForegroundWindow(HWND:hwnd);**

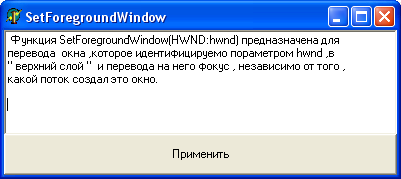


Рис 10 – API –функции

На этой форме можно применить функцию **SetForegroundWindow(HWND:hwnd)** путем нажатия клавиши«Применить».

**Форма для применения функции Windows API SetCurrentDirectory(IpPathName:PChar):bool;**

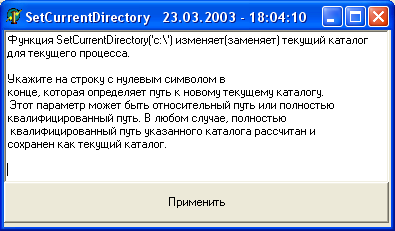


Рис.11 – API-функции

На этой форме можно применить функцию **SetCurrentDirectory(IpPathName:PChar):bool;**

путем нажатия клавиши«Применить».

**Форма настроек главной формы**

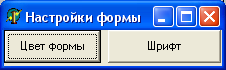


Рис.12 - Настройки

На этой форме расположены настройки с помощью которых настраивается цвет и шрифты главной формы.

**2.3 Элементы главной формы**

|  |  |  |  |
| --- | --- | --- | --- |
| **Элемент формы** | **Назначение** | **Событие** | |
| **Bitgraf** | Кнопка для запуска процедуры прорисовки графиков | BitgrafClick | |
| **Buclear** | Кнопка для очистки поля для вывода значений интегралов | BuclearClick | |
| **Burun** | Кнопка для запуска процедуры вычисления интегралов | BurunClick | |
| **Cbint** | Переключатель для выбора интеграла | CbintChange | |
| **CBmet** | Переключатель для выбора метода решения |  | |
| **ColorDialog** | Палитра для выбора цвета |  | |
| **DateTimePicker** | Календарь (день,месяц,год) |  | |
| **Edzn1** | Поле для ввода нижнего предела интеграла |  | |
| **Edzn2** | Поле для ввода верхнего предела интеграла |  | |
| **Gauge** | Компонент для визуального отображения степени завершенности некоторой длительной орерации или процесса |  | |
| **Imgraf** | Поле для вывода графически результатов вычисления |  | |
| **Imint** | Поле для вывода графического изображения интеграла |  | |
| **Kurs** | Главная форма | MainMenu  FormCreate | |
| **LbCbint** | Метка которая указывает на переключатель **Cbint** |  | |
| **LbCbmet** | Метка которая указывает на переключатель **Cbmet** |  | |
| **LbEdzn1** | Метка которая указывает  поле **Edzn1** |  | |
| **LbEdzn2** | Метка которая указывает  поле **Edzn2** |  | |
| **LbintGraf** | Метка которая указывает  поле **Imgraf** |  | |
| **MMoutput** | Поле для вывода значений припостроении графика |  | |
| **LbznGr** | Метка которая указывает  поле **MMoutput** |  | |
| **N22SetCurrentDirectory** | Пункт меню для применении функции **SetCurrentDirectory** | | N22GetCurrentDirectory1Click |
| N52SetForegroundWindow1Click | Пункт меню для применении функции SetForegroundWindow | | N52SetForegroundWindow1Click |
| RBint1 | Переключатель для построения графика на первый интеграл | | RBint1Click |
| RBint2 | Переключатель для построения графика на второй интеграл | | RBint2Click |
| StatusBar | Строка для вывода подсказок | |  |

**2.4 Функции Windows API**

**SetForeGroundWindow function SetForegroundWindow(hWnd: HWND): BOOL;**

Функция SetForegroundWindow(HWND:hwnd) предназначена для перевода окна, которое идентифицируемо пораметром hwnd,в" верхний слой " и перевода на него фокус, независимо от того, какой поток создал это окно.

SetForegroundWindow( HWND hWnd // Маркер(дескриптор) окна, чтобы перенести к переднему плану)

Параметры

**hWnd-**дискриптор окна который идентифицирует окно, которое должно быть активизировано и перенесено к переднему плану.

Применение:

procedure TForm2.Timer1Timer(Sender: TObject);

begin

SetForegroundWindow(kurs.Handle);

Timer1.Enabled:=false;

end;

Обработчиком события является таймер (Timer1Timer) который и запускает функцию

**SetCurrentDirectory**

**function SetCurrentDirectory(lpPathName: PChar): BOOL;**

Функция **SetCurrentDirectory**(‘**lpPathName: PChar** ‘) изменяет (устанавливает) текущий каталог. Возвращаемое значение было бы True, если текущий каталог был успешно изменен(заменен), или False, если ошибка произошла.

SetCurrentDirectory(

lpPathName // Адрес названия(имени) нового текущего каталога );

**Параметры**

**lpPathName**

Укажите на строку с нулевым символом в

конце, которая определяет путь к новому текущему каталогу.

Применение:

procedure TForm3.Button1Click(Sender: TObject);

begin

SetCurrentDirectory('c:\');

end;

Обработчиком события является кнопка (Button1Click) который и запускает функцию.

**2.5 Основные процедуры**

**Расчет интегралов**

Загрузка в IMAGE рисунка интеграла

procedure TKurs.CbintChange(Sender: TObject);

begin

if CBint.Text='Интеграл №1' then

Imint.Picture.LoadFromFile('int.bmp') else

Imint.Picture.LoadFromFile('int1.bmp');

end;

procedure TKurs.BurunClick(Sender: TObject);

begin

try

try

a:=strtoint(Edzn1.text);

b:=strtoint(Edzn2.text);

Выбор метода и интеграла

n:=strtoint(Edit3.Text);

If (Cbint.Text='Интеграл №1') then

int:=f1;

if (CBmet.Text='Методом Симпсона') then

int1(a,b,n,ss);

if (CBint.Text='Интеграл №1') then

int:=f1;

if (CBmet.Text='Методом трапеций') then

int2(a,b,n,ss);

if (Cbint.Text='Интеграл №2') then

int:=f2;

if (CBmet.Text='Методом Симпсона')then

int1(a,b,n,ss);

if (CBint.Text='Интеграл №2') then

int:=f2;

Обработка исключительных ситуаций

if (CBmet.Text='Методом трапеций')then

int2(a,b,n,ss);

Memo1.Lines.Add(' '+CBint.Text);

Memo1.Lines.Add((CBmet.Text)+' = '+(FloatToStrF(ss,ffNumber,9,3)));

except

on EConvertError do

begin

raise Exception.Create('Внимание!Ошибка в записи числа!');

end;

end;

finally

Динамическая загрузка mydll.dll библиотеки

if (a)>(b)

then

ShowMessage('Вверхняя граница должна быть больше нижней');

end;

end;

procedure TKurs.FormCreate(Sender: TObject);

begin

han:=loadlibrary('mydll.dll');

if han=0 then showmessage('+Билиотека не найдена!')

else

begin

@f1:=getprocaddress(han,'f1');

@f2:=getprocaddress(han,'f2');

end;

end;

procedure TKurs.int1(a,b:real;n:integer;var ss:real);

var i1,i2,i:integer;h,s,s1,x1,s2,x2,yn,x:real;

begin

**INT** – переменная процедурного типа которой присваивается функция первого или второго интеграла

i:=1;

s1:=0;

Метод Симпсона

y0:=INT(a);

h:=(b-a)/n;

repeat

x1:=a+i\*h;

s1:=s1+INT(x1);

i:=i+2;

until i>n-1;

i1:=2;

s2:=0;

yn:=INT(b);

repeat

x2:=a+i1\*h;

s2:=s2+INT(x2);

i1:=i1+2;

until i1>n-2;

yn:=a-n\*h;

s:=(h/3)\*(y0+4\*s1+2\*s2+yn);

ss:=s;

end;

procedure TKurs.int2(a,b:real;n:integer;var ss:real);

var i,j:integer;k:real;

Формирование таблицы со значениями

begin

Form5.StringGrid1.Cells[0,0]:='I';

Form5.StringGrid1.Cells[1,0]:='Xi';

form5.StringGrid1.Cells[2,0]:='SQR(Xi)';

Form5.StringGrid1.Cells[3,0]:='(0.5\*sqr(Xi)+1.5))';

Form5.StringGrid1.Cells[4,0]:='y0..yn';

Form5.StringGrid1.Cells[5,0]:='y1..yn';

s:=0;

Метод трапеций

h:=(b-a)/n;

Form5.Label2.Caption:=floattostr(h);

Form5.StringGrid1.RowCount:=n+3;

Формирование таблицы со значениями

for i:=0 to n do begin

s:=s+iNT(a+i\*h);

j:=0;

Form5.StringGrid1.Cells[j,i+1]:=intToStr(i);

j:=1;

Form5.StringGrid1.Cells[j,i+1]:=FloatToStrF((a+i\*h),ffNumber,4,4);

j:=2;

Form5.StringGrid1.Cells[j,i+1]:=FloatToStrF(sqr(a+i\*h),ffNumber,4,4);

j:=3;

form5.StringGrid1.Cells[j,i+1]:=FloatToStrF(0.5\*sqr(a+i\*h)+1.5,ffNumber,4,4);

j:=5;

Form5.StringGrid1.Cells[j,i+2]:=FloatToStrF(int(a+(i+1)\*h),ffNumber,4,4);

end;

Form5.Show;

Form5.StringGrid1.Cells[0,n+2]:='Сумма';

Form5.StringGrid1.Cells[5,n+2]:=FloatToStrF(s-(int(a+(n)\*h)+int(a+0\*h)),ffNumber,4,4);

Form5.StringGrid1.Cells[4,1]:=FloatToStrF(int(a+0\*h),ffNumber,4,4);

Form5.StringGrid1.Cells[4,n+1]:=FloatToStrF(int(a+(n)\*h),ffNumber,4,4);

Form5.StringGrid1.Cells[4,n+2]:=FloatToStrF(int(a+(n)\*h)+int(a+0\*h),ffNumber,4,4);

s:=((INT(a)+INT(b))/2+s)\*h;

ss:=s;

end;

**2.6 Методика визуализации**

Визуализация полученных значений при решении интеграла в зависимости от значений нижней границы при фиксированных значениях верхней границы определенного интеграла была получена с помощью стандартного компонента визуализации графических данных, а также с помощью класса TCanvas его свойств и методов которые описаны ниже.

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| **Font** | Определяет шрифт, чтобы использовать его при записи текста на изображении |
| **Brush** | Определяет цвет и образец использования холста для заполнения графических форм и фонов |
| **Pen** | Определяет вид пера для использования для рисования строк и структурирования форм |
| **Pixels** | Определяет цвет области пикселов в пределах потока ClipRect. |
| **Метод** | **Описание** |
| **Draw** | Выполняет графический объект указанным параметром Graphic на холсте в местоположении, заданном координатами (X, Y). |
| **FillRect** | Заполняет указанный прямоугольник на холсте, используя текущую кисть. |
| **LineTo** | Тянет(рисует) строку на холсте от PenPos, к координатам указанных X и Y, и устанавливает перьевую позицию в (X, Y). |
| **MoveTo** | Изменяет позицию текущего рисунка к координатам (X, Y). |

procedure TKurs.BitgrafClick(Sender: TObject);

var

x,ymin,ymax,hx1,hx,hy,xn,xk,mx,sx,my,sy,yy,xx:real;i,X1,Y1,Y2,x3,xa,ya,xb,yb:integer;

begin

if Timer1.Enabled=false then

Timer1.Enabled:=true;

Kurs.Imgraf.Canvas.FillRect(rect(0,0,620,370));

xn:=strtofloat(Edzn1.text);

xk:=StrTofloat(Edzn2.Text);

n:=strtoint(Edit3.Text);

x:=xn;ymax:=-1e30;ymin:=1e30;

while x<=xk do

begin

Выбор интеграла путем присвоения переменной процедурного типа INT функции одного из интеграла

if RadioButton1.Checked then

int:=f1;

if RadioButton2.Checked then

int:=f2;

Нахождение минимального и максимального значения интеграла (SS) для последующего масштабирования графика

int1(x,xk,n,ss);

if ss>ymax then ymax:=ss;

if ss<ymin then ymin:=ss;

x:=x+0.01;

end;

x:=xn;

int1(x,xk,n,ss);

x1:=round(mx\*x+sx);

y1:=round(my\*x+sy);

repeat

int1(x,xk,n,ss);

x2:=ROUND(mx\*x+sx);

y2:=round(-my\*ss+sy);

kurs.Imgraf.Canvas.Pen.Color:=ColorDialog2.color;

kurs.Imgraf.Canvas.Brush.Color:=clwhite;

kurs.Imgraf.Canvas.MoveTo(x1,y1);

kurs.Imgraf.Canvas.LineTo(x2,y2);

x1:=x2;y1:=y2;

k:=true;

kurs.Imgraf.Canvas.Pen.Color:=clred;

kurs.Imgraf.Canvas.Pen.Width:=2;

hx1:=1;

hx:=(xk-xn)/10;

Нахождение масштабных коэффициентов

xa:=0;ya:=0;xb:=450;yb:=310;

mx:=(xb-xa)/(xk-xn);

my:=(yb-ya)/(ymax-ymin);

sy:=ya+my\*ymax;sx:=xa-mx\*xn;

kurs.Imgraf.Canvas.MoveTo(xb,yb);

kurs.Imgraf.Canvas.LineTo(xa,yb);

x:=x+0.01;

until x>xk;

i:=3;

xx:=xn;

//вертикальные состовляющие

Прорисовка сетки (вертикальные составляющие)

repeat

kurs.Imgraf.Canvas.MoveTo(i,ya);

kurs.Imgraf.Canvas.LineTo(i,yb);

kurs.Imgraf.Canvas.TextOut(i,yb+5,FloatToStr(xx));

inc(i,Round((Imgraf.Width)/10));

Прорисовка сетки (горизонтальные составляющие)

xx:=xx+hx;

until i>Imgraf.Width;

//горизонтальные состовляющие

i:=Imgraf.Height-50;

repeat

yy:=(sy-i)/my;

kurs.Imgraf.Canvas.MoveTo(0,i);

kurs.Imgraf.Canvas.LineTo(Imgraf.Width,i);

kurs.Imgraf.Canvas.TextOut(0,i-5,FloatToStrF(yy,ffNumber,4,2));

i:=i-round(hy);

dec(i,Round((Imgraf.Height)/8));

until i<0 ;

**2.6 Подключение DLL-библиотеки**

Существует два типа исполняемых файлов Windows: программы и библиотеки динамической компоновки (Dynamic Link Libraries,Dll). Когда мы разрабатываем приложение Delphi, мы, как правило создаем исполняемый файл программы, который обладает расширением EXE. Однако приложения Delphi в процессе своей работы часто обращаются к функциям, содержащимся в библиотеках DLL. Например, каждый раз, когда мы напрямую обращаемся к Windows API, на самом деле мы обращаемся к DLL.

Что такое динамическая компоновка ?

При использовании *динамической* компоновки, которая выполняется в случае, если разрабатываемый нами код обращается к функции, содержащейся в библиотеки DLL, компоновщик получает необходимую информацию о функции из ее обьявления *external.*Эта информация используется для заполнения некоторых таблиц внутри исполняемого файла. Когда операционная система Windows загружает исполняемый файл в память, в первую очередь она осуществляет загрузку всех необходимых для функционирования программы библиотек, и только после этого программы начинает работу. В процессе загрузки библиотек DLL операционная система заполняет таблицы внутренние таблицы программы адресами функций из загруженных в память библиотек DLL.

Подключение Dll –подключение путем динамической Dll-загрузки.

procedure TKurs.FormCreate(Sender: TObject);

begin

han:=loadlibrary('mydll.dll');

if han=0 then showmessage('+Библиотека не найдена!')

else

begin

@f1:=getprocaddress(han,'f1');

@f2:=getprocaddress(han,'f2');

end;

**Листинг Dll-библиотеки**

library mydll;

{ Important note about DLL memory management: ShareMem must be the

first unit in your library's USES clause AND your project's (select

Project-View Source) USES clause if your DLL exports any procedures or

functions that pass strings as parameters or function results. This

applies to all strings passed to and from your DLL--even those that

are nested in records and classes. ShareMem is the interface unit to

the BORLNDMM.DLL shared memory manager, which must be deployed along

with your DLL. To avoid using BORLNDMM.DLL, pass string information

using PChar or ShortString parameters. }

uses

SysUtils,

Classes;

{$R \*.res}

function f1(x:real):real;

Функция первого интеграла

begin

f1:=1/sqrt(0.5\*sqr(x)+1.5);

end;

Функция второго интеграла

function f2(x:real):real;

begin

f2:=(x+1)\*cos(sqr(x));

end;

exports

f1 index 1 name 'f1',f2 index 2 name 'f2';

begin

end.

**2.7 Компоненты индивидуального задания**

DateTimePicker1: TDateTimePicker;

TDateTimePicker - визуальный компонент, разработанный (предназначенный) определенно для ввода дат или времен. В DateTimePicker датируют режим, это походит на список или поле со списком, за исключением того, что раскрывающийся список заменен календарной иллюстрацией; пользователи могут выбирать дату от календаря. Даты или времена могут также быть выбраны, листая со Стрелками "вверх" и "вниз".

Gauge используется для визуального отображения пользователю степень завершенности некоторой длительной орерации или процесса

|  |  |
| --- | --- |
| Свойство | Назначение |
| Backcolor | Изменение цвета фона заднего плана |
| Forecolor | Изменение цвета индикатора движения |
| Kind  gkVerticalBar  gkHorizontalBar  gkNeedle  gkPie  gkText | Изменение основной формы индикатора движения  *gkVerticalBar –*вертикальный вид индикатора движения  *gkHorizontalBar-* горизонтальный вид индикатора движения  *gkNeedle-* индикатора движения в виде полукруга  *gkPie-* индикатора движения в виде сектора  *gkText-* текстовое отображение индикатора движения |
| MaxValue | Максимальное значение диапазона (конечное) изменения индикатора движения |
| MinValue | Минимальное значение диапазона (начальное) изменения индикатора движения |
| Progress | Задание начального положения индикатора движения |
| ShowText | Значение True означает отображение индикатора движения с текстовой информацией (числовой) |

**3.** **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**3.1 Системные требования**

Pentium 133 Mhz, RAM 32 Mb, HDD 10Mb, Windows 98

* 1. **Помощь**

Для справки нажмите F1, и появится окно

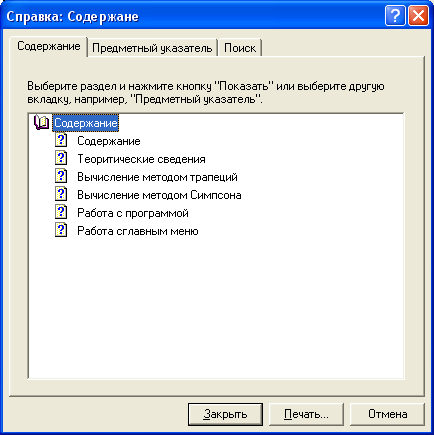


Рис.8- Справка

**3.3 Установка программы**

Для установки программы необходимо скопировать папку EXE

Содержимое папки:

Project1.exe

Mydll.dll

Helpkurs

**Выводы**

Таким образом,данная работа позволила мне:

закрепить на опыте знания, полученные при изучении дисциплины “Системное программирование”.

научился разрабатывать программные продукты под операционную среду Windows

закрепил следующие знания:

Программирование в Delphi

Программирование программ использующие код, написанный на других языках программирования (использование DLL )

Динамическая компоновка с DLL

Численные методы для вычисления определенных интегралов

Написание Help в стиле Windows.

**Список используемой литературы**

1. Delphi 6. Для профессионалов. Разработка COM: Пер.с англ.:-464c. 2005
2. Разработка приложений в среде Delphi: пер.с англ.:Уч.пос-М.:Издательский дом «Вильямс»,2006.-464с.

**Листинг программы**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls, Menus, ExtDlgs, ExtCtrls, ToolWin, Gauges,

AppEvnts, TeEngine, Series, TeeProcs, Chart, TeeFunci, Buttons, ShellAPI,

ScktComp, ImgList;

type

TKurs = class(TForm)

PageControl1: TPageControl;

TabSheet1: TTabSheet;

TabSheet2: TTabSheet;

Edzn1: TEdit;

Edzn2: TEdit;

Edit3: TEdit;

MainMenu1: TMainMenu;

file1: TMenuItem;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

SavePictureDialog1: TSavePictureDialog;

N4: TMenuItem;

N5: TMenuItem;

N6: TMenuItem;

StatusBar1: TStatusBar;

Imint: TImage;

Cbint: TComboBox;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

CBmet: TComboBox;

Memo1: TMemo;

Label5: TLabel;

Burun: TButton;

N7: TMenuItem;

N22GetCurrentDirectory1: TMenuItem;

N52SetForegroundWindow1: TMenuItem;

DateTimePicker1: TDateTimePicker;

Imgraf: TImage;

Label6: TLabel;

Buclear: TButton;

Bitgraf: TBitBtn;

GroupBox1: TGroupBox;

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

ApplicationEvents1: TApplicationEvents;

N8: TMenuItem;

N9: TMenuItem;

ImageList1: TImageList;

Timer1: TTimer;

Gauge1: TGauge;

N10: TMenuItem;

N11: TMenuItem;

ColorDialog1: TColorDialog;

ColorDialog2: TColorDialog;

N12: TMenuItem;

procedure N1Click(Sender: TObject);

procedure N6Click(Sender: TObject);

procedure CbintChange(Sender: TObject);

procedure BurunClick(Sender: TObject);

procedure int1(a,b:real;n:integer;var ss:real);

procedure int2(a,b:real;n:integer;var ss:real);

procedure FormCreate(Sender: TObject);

procedure N22GetCurrentDirectory1Click(Sender: TObject);

procedure N52SetForegroundWindow1Click(Sender: TObject);

procedure ApplicationEvents1Hint(Sender: TObject);

procedure BuclearClick(Sender: TObject);

procedure BitgrafClick(Sender: TObject);

procedure RadioButton1Click(Sender: TObject);

procedure RadioButton2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N9Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure N10Click(Sender: TObject);

procedure N11Click(Sender: TObject);

procedure N12Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

tfun=function(x:real):real;

var

Kurs: TKurs;

INT,F1,f2:tfun;

han:thandle;

k:boolean;

a,b,s,h,ss,y0,yn,s2,x2,xx1,xx2,xk:real;

n,n1,i1:integer;

mplementation

uses Unit2, Unit3, Unit4, Unit5 ;

{$R \*.dfm}

procedure TKurs.N1Click(Sender: TObject);

begin

Imgraf.Picture.SaveToFile(SavePictureDialog1.FileName+'.bmp');

end;

procedure TKurs.N4Click(Sender: TObject);

begin

if k=true then begin

N1.Enabled:=true;

if SavePictureDialog1.Execute then

Imgraf.Picture.SaveToFile(SavePictureDialog1.FileName+'.bmp');

end

else

ShowMessage('Ãðàôèê íå îòðèñîâàí');

end;

procedure TKurs.N6Click(Sender: TObject);

begin

Application.HelpCommand(HELP\_FINDER,0);

end;

procedure TKurs.CbintChange(Sender: TObject);

begin

if CBint.Text='Èíòåãðàë ¹1' then

Imint.Picture.LoadFromFile('int.bmp') else

Imint.Picture.LoadFromFile('int1.bmp');

end;

procedure TKurs.BurunClick(Sender: TObject);

begin

a:=strtofloat(Edzn1.text);

b:=strtofloat(Edzn2.text);

n:=strtoint(Edit3.Text);

if (edzn1.Text<>'') and (Edzn2.Text<>'') and

(Cbint.ItemIndex<>-1)and (CBmet.ItemIndex<>-1) then

begin

try

If (Cbint.Text='Èíòåãðàë ¹1') then

int:=f1;

if (CBmet.Text='Ìåòîäîì Ñèìïñîíà') then

int1(a,b,n,ss);

if (CBint.Text='Èíòåãðàë ¹1') then

int:=f1;

if (CBmet.Text='Ìåòîäîì òðàïåöèé') then

int2(a,b,n,ss);

if (Cbint.Text='Èíòåãðàë ¹2') then

int:=f2;

if (CBmet.Text='Ìåòîäîì Ñèìïñîíà')then

int1(a,b,n,ss);

if (CBint.Text='Èíòåãðàë ¹2') then

int:=f2;

if (CBmet.Text='Ìåòîäîì òðàïåöèé')then

int2(a,b,n,ss);

Memo1.Lines.Add(' '+CBint.Text);

Memo1.Lines.Add((CBmet.Text)+' = '+(FloatToStrF(ss,ffNumber,9,3)));

finally

if (a)>(b)

then

ShowMessage('Ââåðõíÿÿ ãðàíèöà äîëæíà áûòü áîëüøå íèæíåé');

end;

end

else

if (Edzn1.Text=' ') then ShowMessage('Íå ââåäåíà ââåðõíÿÿ ãðàíèöà!');

if (Edzn2.Text=' ') then ShowMessage('Íå ââåäåíà íèæíÿÿ ãðàíèöà!');

if (Cbint.ItemIndex=-1) then ShowMessage('Âíèìàíèå!Íå âûáðàí èíòåãðàë!');

if (CBmet.ItemIndex=-1) then ShowMessage('Íå âûáðàí ìåòîä ðåøåíèÿ!');

end;

procedure TKurs.FormCreate(Sender: TObject);

begin

han:=loadlibrary('mydll.dll');

if han=0 then showmessage('+Áèëèîòåêà íå íàéäåíà!')

else

begin

@f1:=getprocaddress(han,'f1');

@f2:=getprocaddress(han,'f2');

end;

end;

procedure TKurs.int1(a,b:real;n:integer;var ss:real);

ar i1,i2,i:integer;h,s,s1,x1,s2,x2,yn,x:real;

begin

i:=1;

s1:=0;

y0:=INT(a);

h:=(b-a)/n;

repeat

x1:=a+i\*h;

s1:=s1+INT(x1);

i:=i+2;

until i>n-1;

i1:=2;

s2:=0;

yn:=INT(b);

repeat

x2:=a+i1\*h;

s2:=s2+INT(x2);

i1:=i1+2;

until i1>n-2;

yn:=a-n\*h;

s:=(h/3)\*(y0+4\*s1+2\*s2+yn);

ss:=s;

end;

procedure TKurs.int2(a,b:real;n:integer;var ss:real);

var i,j:integer;k:real;

begin

Form5.StringGrid1.Cells[0,0]:='I';

Form5.StringGrid1.Cells[1,0]:='Xi';

form5.StringGrid1.Cells[2,0]:='SQR(Xi)';

Form5.StringGrid1.Cells[3,0]:='(0.5\*sqr(Xi)+1.5))';

Form5.StringGrid1.Cells[4,0]:='y0..yn';

Form5.StringGrid1.Cells[5,0]:='y1..yn';

s:=0;

h:=(b-a)/n;

Form5.Label2.Caption:=floattostr(h);

Form5.StringGrid1.RowCount:=n+3;

for i:=0 to n do begin

s:=s+iNT(a+i\*h);

j:=0;

Form5.StringGrid1.Cells[j,i+1]:=intToStr(i);

j:=1;

Form5.StringGrid1.Cells[j,i+1]:=FloatToStrF((a+i\*h),ffNumber,4,4);

j:=2;

orm5.StringGrid1.Cells[j,i+1]:=FloatToStrF(sqr(a+i\*h),ffNumber,4,4);

j:=3;

form5.StringGrid1.Cells[j,i+1]:=FloatToStrF(0.5\*sqr(a+i\*h)+1.5,ffNumber,4,4);

j:=5;

Form5.StringGrid1.Cells[j,i+2]:=FloatToStrF(int(a+(i+1)\*h),ffNumber,4,4);

end;

Form5.Show;

Form5.StringGrid1.Cells[0,n+2]:='Ñóììà';

Form5.StringGrid1.Cells[5,n+2]:=FloatToStrF(s-(int(a+(n)\*h)+int(a+0\*h)),ffNumber,4,4);

Form5.StringGrid1.Cells[4,1]:=FloatToStrF(int(a+0\*h),ffNumber,4,4);

Form5.StringGrid1.Cells[4,n+1]:=FloatToStrF(int(a+(n)\*h),ffNumber,4,4);

Form5.StringGrid1.Cells[4,n+2]:=FloatToStrF(int(a+(n)\*h)+int(a+0\*h),ffNumber,4,4);

s:=((INT(a)+INT(b))/2+s)\*h;

ss:=s;

end;

procedure TKurs.N22GetCurrentDirectory1Click(Sender: TObject);

begin

form2.Show;

end;

procedure TKurs.N52SetForegroundWindow1Click(Sender: TObject);

begin

form3.Show;

end;

procedure TKurs.ApplicationEvents1Hint(Sender: TObject);

begin

StatusBar1.Panels[0].Text:=Application.Hint;

end;

procedure TKurs.BuclearClick(Sender: TObject);

begin

Memo1.Lines.Clear;

end;

procedure TKurs.BitgrafClick(Sender: TObject);

var

x,ymin,ymax,hx1,hx,hy,xn,xk,mx,sx,my,sy,yy,xx:real;i,X1,Y1,Y2,x3,xa,ya,xb,yb:integer;

begin

if Timer1.Enabled=false then

Timer1.Enabled:=true;

Kurs.Imgraf.Canvas.FillRect(rect(0,0,620,370));

xn:=strtofloat(Edzn1.text);

xk:=StrTofloat(Edzn2.Text);

n:=strtoint(Edit3.Text);

x:=xn;ymax:=-1e30;ymin:=1e30;

while x<=xk do

begin

if RadioButton1.Checked then

int:=f1;

if RadioButton2.Checked then

int:=f2;

int1(x,xk,n,ss);

if ss>ymax then ymax:=ss;

if ss<ymin then ymin:=ss;

x:=x+0.01;

end;

x:=xn;

int1(x,xk,n,ss);

x1:=round(mx\*x+sx);

y1:=round(my\*x+sy);

repeat

int1(x,xk,n,ss);

x3:=ROUND(mx\*x+sx);y2:=round(-my\*ss+sy);

kurs.Imgraf.Canvas.Pen.Color:=ColorDialog2.color;

kurs.Imgraf.Canvas.Brush.Color:=clwhite;

kurs.Imgraf.Canvas.MoveTo(x1,y1);

kurs.Imgraf.Canvas.LineTo(x3,y2);

x1:=x3;y1:=y2;

k:=true;

kurs.Imgraf.Canvas.Pen.Color:=clred;

kurs.Imgraf.Canvas.Pen.Width:=2;

hx1:=1;

hx:=(xk-xn)/10;

xa:=0;ya:=0;xb:=450;yb:=310;

mx:=(xb-xa)/(xk-xn);my:=(yb-ya)/(ymax-ymin);

sy:=ya+my\*ymax;sx:=xa-mx\*xn;

kurs.Imgraf.Canvas.MoveTo(xb,yb);

kurs.Imgraf.Canvas.LineTo(xa,yb);

x:=x+0.01;

until x>xk;

i:=3;

xx:=xn;

//âåðòèêàëüíûå ñîñòîâëÿþùèå

repeat

kurs.Imgraf.Canvas.MoveTo(i,ya);

kurs.Imgraf.Canvas.LineTo(i,yb);

kurs.Imgraf.Canvas.TextOut(i,yb+5,FloatToStr(xx));

inc(i,Round((Imgraf.Width-50)/10));

xx:=xx+hx;

until i>Imgraf.Width;

//ãîðèçîíòàëüíûå ñîñòàâëÿþùèå

i:=Imgraf.Height-50;

repeat

yy:=(sy-i)/my;

kurs.Imgraf.Canvas.MoveTo(0,i);

kurs.Imgraf.Canvas.LineTo(Imgraf.Width,i);

kurs.Imgraf.Canvas.TextOut(0,i-5,FloatToStrF(yy,ffNumber,4,2));

i:=i-round(hy);

dec(i,Round((Imgraf.Height-50)/10));

until i<0 ;

end;

rocedure TKurs.RadioButton1Click(Sender: TObject);

begin

Bitgraf.Enabled:=true;

end;

procedure TKurs.RadioButton2Click(Sender: TObject);

begin

Bitgraf.Enabled:=true;

end;

procedure TKurs.Button3Click(Sender: TObject);

begin

SetForegroundWindow(HWND\_TOP );

end;

procedure TKurs.N9Click(Sender: TObject);

begin

Form4.Show;

end;

procedure TKurs.Timer1Timer(Sender: TObject);

begin

Gauge1.Progress:=Gauge1.Progress+10;

if Gauge1.Progress=100 then begin

Timer1.Enabled:=false;

Gauge1.Progress:=0;

end;

end;

procedure TKurs.N10Click(Sender: TObject);

begin

if ColorDialog2.Execute then

kurs.Imgraf.Canvas.Pen.Color:=ColorDialog1.Color;

end;

procedure TKurs.N11Click(Sender: TObject);

begin

if ColorDialog1.Execute then

kurs.Imgraf.Canvas.Pen.Color:=ColorDialog1.Color;

end;

procedure TKurs.N12Click(Sender: TObject);

begin

Kurs.Close;

end;

end.