## СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

**1. ТЕОРЕТИЧЕСИЙ РАЗДЕЛ** 4

**1.1** **Основные характеристики языка** 4

**1.1.1** **Общая характеристика Delphi.** 4

**1.1.2** **Место Delphi в современном программировании.** 4

**1.2** **Численное интегрирование** 5

**1.2.1** **Метод прямоугольников** 6

**1.2.2** **Метод трапеций** 6

**1.2.3** **Метод парабол (метод Симпсона)** 7

**1.2.4** **Метод Боде** 7

**1.2.5** **Метод Уэддля** 7

**2.** **ПРОЕКТНАЯ ЧАСТЬ** 8

**2.1.** **Постановка задачи** 8

**2.3.** **Блок-схемы подпрограмм** 9

**2.3.1.** **Метод прямоугольников** 9

**2.3.2.** **Метод трапеций** 10

**2.3.3.** **Метод парабол (Симпсона)** 11

2.3.4. **Метод Бодэ** 12

**2.3.5.** **Метод Уэдлла** 12

**2.4.** **Разработка интерфейса** 13

**3.** **ЭКСПЕРИМЕНТАЛЬНАЯ ЧАСТЬ** 15

**3.1.** **Тестирование программы** 15

ЗАКЛЮЧЕНИЕ 18

СПИСОК ЛИТЕРАТУРЫ 19

Приложение 1. Листинг программы. 20

## ВВЕДЕНИЕ

Появление и непрерывное совершенствование быстродействующих электронных вычислительных машин (ЭВМ) привело к подлинно революционному преобразованию наукиa вообще и математики в особенности. Изменилась технология научных исследований, колоссально увеличились возможности теоретического изучения, прогноза сложных процессов, проектирования инженерных конструкций. Решение крупных научно-технических проблем, примерами которых могут служить проблемы овладения ядерной энергией и освоения космоса, стало возможным лишь благодаря применению математического моделирования и новых численных методов, предназначенных для ЭВМ.

В настоящее время можно говорить, что появился новый способ теоретического исследования сложных процессов, допускающих математическое описание, - вычислительный эксперимент, т.е. исследование естественнонаучных проблем средствами вычислительной математики. Разработка и исследование вычислительных алгоритмов, и их применение к решению конкретных задач составляет содержание огромного раздела современной математики - вычислительной математики.

Численные методы дают приближенное решение задачи. Это значит, что вместо точного решения и (функции или функционала) некоторой задачи мы находим решение у другой задачи, близкое в некотором смысле (например, по норме) к искомому. Основная идея всех методов - дискретизация или аппроксимация (замена, приближение) исходной задачи другой задачей, более удобной для решения на ЭВМ, причем решение аппроксимирующей задачи зависит от некоторых параметров, управляя которыми, можно определить решение с требуемой точностью. Например, в задаче численного интегрирования такими параметрами являются узлы и веса квадратурной формулы. Далее, решение дискретной задачи является элементом конечномерного пространства.

Численное интегрирование (историческое название: квадратура) - вычисление значения определённого интеграла (как правило, приближённое), основанное на том, что величина интеграла численно равна площади криволинейной трапеции, ограниченной осью абсцисс, графиком интегрируемой функции и отрезками прямых, которые являются пределами интегрирования.

Необходимость применения численного интегрирования чаще всего может быть вызвана отсутствием у первообразной функции представления в элементарных функциях и, следовательно, невозможностью аналитического вычисления значения определённого интеграла по формуле Ньютона-Лейбница. Также возможна ситуация, когда вид первообразной настолько сложен, что быстрее вычислить значение интеграла численным методом.

1. **ТЕОРЕТИЧЕСИЙ РАЗДЕЛ**
   1. Основные характеристики языка
      1. **Общая характеристика Delphi.**

В настоящее время программирование бурно развивается, как с точки зрения расширения круга решаемых им задач, так и с точки зрения существенного усложнения используемых в программировании технологий Причем особо необходимо отметить немалые размеры разрабатываемы программных продуктов. Все это требует максимального упрощения и ускорения процесса разработки приложений и использования ранее реализованных программных фрагментов.  
 Такие требования к современному программированию привели к созданию многочисленных RAD-систем (от англ. RAD — Rapid Application Development — быстрая разработка приложений), представляющих собой интегрированные среды разработчика, включающие в себя:  
 - средства быстрого и удобного построения программ, в том числе визуального;  
 - встроенные компиляторы и отладчики;  
 - системы коллективной разработки проектов и т.д.  
 Одной из таких RAD-систем является Delphi. Итак, Delphi — это объектно-ориентированная среда для визуального проектирования Windows приложений с развитыми механизмами повторного использования программного кода. Основным конкурентом Delphi является среда разработк Microsoft Visual C++, имеющая свои преимущества и недостатки, однако являющаяся более популярной, в основном, в силу того, что разработана именно фирмой Microsoft Существенной чертой Delphi является компонентная модель разработки программных продуктов. Суть модели заключается в поддержке системой постоянно расширяемого набора объектных компонентов, из которых и строится программа. Компоненты в Delphi просты для использования и развития, как результат сокрытия значительной части той структуры программы, которая близка к взаимодействию с операционной системой. Таким образом, для создания в Delphi несложных программных продуктов совершенно не обязательно понимать внутреннюю структуру Windows-приложения, получаемого после разработки в Delphi. Достаточно просто уметь работать с некоторыми компонентами, поставляемыми вместе со средой разработчика. При этом начать работу со средой можно практически без предварительного ознакомления, а написание первого приложения не потребует углубления в особенности системы. Этому отчасти способствует удобный интерфейс среды разработчика, не перегруженный излишними вопросами к разработчику.  
 Однако такой подход совершенно неприемлем для серьезного программирования, и, рано или поздно, придется освоить и основы программирования под ОС Windows, и серьезно изучить саму среду разработки Delphi, а также возможности, которые она предоставляет. Кроме того, конечно же, для создания качественных программных продуктов необходимо глубокое понимание компонентной модели. 

* + 1. **Место Delphi в современном программировании.**

Наиболее существенный отрыв Delphi от ближайших аналогов состоит в действительно быстрой разработке приложений, обладающих сложным пользовательским интерфейсом, особенно имеющим сильные взаимосвязи между элементами управления, расположенными в окнах программы. Также Delphi предлагает довольно мощный набор компонентов для работы с базами данных. Причем иерархия компонентов для работы с БД организована таким образом, что практически неважно, какой именно базой данных пользуется приложение — это может быть и локальная БД и промышленный сервер, типа Oracle или MS SQL Server. Существенным преимуществом Delphi в этой области является возможность управления базами данных на логическом уровне, соответствующем понятиям самих баз данных, без использования низкоуровневых запросов к драйверам. Такие возможности Delphi обусловили ее широкую применяемость при разработке АСУП — автоматизированных систем управления предприя-тиями. Однако это не единственная область применения, так как возможности Delphi не ограничиваются вышеперечисленными. Delphi является языком программирования широкого назначения и позволяет разработать программный продукт любой сложности для любой области. Даже если какие-либо возможности и не поддерживаются напрямую, то этот недостаток может быть исправлен добавлением соответствующих компонентов в систему. Такой подход касается, например, технологии DirectX, не поддерживаемой Delphi в ее исходной комплектации, но существуют компоненты для использования DirectX, которые легко интегрируются в среду разработки.  
В любом случае, подпрограммы, реализованные в других Windows языках программирования, могут быть использованы в Delphi через механизм динамически компонуемых библиотек (от англ. Dynamic Link Library — DLL — Динамически компонуемая библиотека). Заметим, что многие системные библиотеки Windows изначально подключены к Delphi, а вызов функций из них ничем не отличается от использования обычных библиотек Pascal.  
 С появлением среды разработки Kylix под операционную систему Linux, полностью соответствующую Delphi за исключением некоторых аспектов, связанных с различиями в технологиях, используемых в этих операционных системах, часть приложений, написанных в Delphi, стала переносимой под Linux, что открывает еще более широкие возможности этой среды разработки.

* 1. **Численное интегрирование**

Задача численного интегрирования состоит в замене исходной подинтегральной функции f(x), для которой трудно или невозможно записать первообразную в аналитике, некоторой аппроксимирующей функцией φ(x). Такой функцией обычно является полином (кусочный полином)

http://solidbase.karelia.ru/edu/meth_calc/files/04.files/image002.gif

То есть:

http://solidbase.karelia.ru/edu/meth_calc/files/04.files/image004.gif, где

http://solidbase.karelia.ru/edu/meth_calc/files/04.files/image006.gif – априорная погрешность

метода на интервале интегрирования,  а r(x) – априорная погрешность метода на отдельном шаге интегрирования.

Кратко рассмотрим основные методы численного интегрирования.

* + 1. **Метод прямоугольников**

Метод прямоугольников получается при замене подынтегральной функции на константу. В качестве константы можно взять значение функции в любой точке отрезка left[a, b\right]\,\!. Наиболее часто используются значения функции в середине отрезка и на его концах. Соответствующие модификации носят названия методов средних прямоугольников, левых прямоугольников и правых прямоугольников. Формула для приближенного вычисления значения определённого интеграла методом прямоугольников имеет вид

 \approx f(x) (b-a),

где =\frac{a+b}{2}, \  или \ , соответственно.

* + 1. **Метод трапеций**

Если функцию на каждом из частичных отрезков аппроксимировать прямой, проходящей через конечные значения, то получим метод трапеций.

Площадь трапеции на каждом отрезке:

I_i \approx \frac{f(x_{i-1})+f(x_{i})}{2} (x_{i}-x_{i-1}).

Погрешность аппроксимации на каждом отрезке:

\left| R_{i} \right| \leqslant \frac{\left( b-a \right)^3}{12n^2} M_{2,i},

где

_{2,i}=\max_{x\mathcal{2}[x_{i-1},x_i]} \left| f''(x) \right| .

Полная формула трапеций в случае деления всего промежутка интегрирования на отрезки одинаковой длины h:

I \approx h\left( \frac{f(x_{0})+f(x_{n})}{2} + \sum_{i=1}^{n-1}f(x_{i})\right), где =\frac{b-a}{n}

Погрешность формулы трапеций:

\left| R \right| \leqslant \frac{\left( b-a \right)^3}{12n^2} M_{2}, где _{2}=\max_{x\mathcal{2}[a,b]} \left| f''(x) \right| 

* + 1. **Метод парабол (метод Симпсона)**

Использовав три точки отрезка интегрирования можно заменить подынтегральную функцию параболой. Обычно в качестве таких точек используют концы отрезка и его среднюю точку. В этом случае формула имеет очень простой вид

 \approx \frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right).

Если разбить интервал интегрирования на 2N равных частей, то имеем

 \approx \frac{b-a}{6N}\left(f_0 + 4 \left(f_1 + f_3 + ... +f_{2N-1}\right) + 2 \left(f_2 + f_4 + ... +f_{2N-2}\right) + f_{2N}\right),

где _i = f\left(a+\frac{(b-a)i}{2N}\right).

* + 1. **Метод Боде**

Пусть значения функции  (x)  d в точках _i  равномерно распределенных по  =x_(i+1)-x_i , так что  _1=f(x_1), _2=f(x_2), ..., _5=f(x_5)

Составная формула Боде имеет вид:



* + 1. **Метод Уэддля**

Метод Уэддля позволяет значениям функции быть в таблице на точках равномерно распределенных(отделенных)  =x_(i+1)-x_i , так что  _1=f(x_1), _2=f(x_2), ....

Затем метод задан как формула Ньютона-Котеса

int_(x_1)^(x_7)f(x)dx=3/(10)h(f_1+5f_2+f_3+6f_4+f_5+5f_6+f_7). 

1. **ПРОЕКТНАЯ ЧАСТЬ**
   1. **Постановка задачи**

Содержательная задача:

Разработать приложение для Windows в среде Delphi для вычисления определенных интегралов с заданной точностью следующими методами:

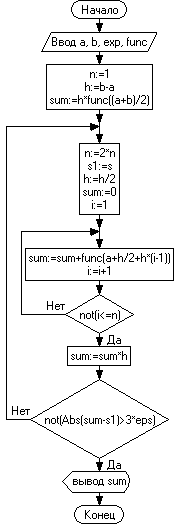
* Прямоугольников;
* Трапеций;
* Парабол(Симпсона);
* Бодэ;
* Уэдлла;

Специальные требования:

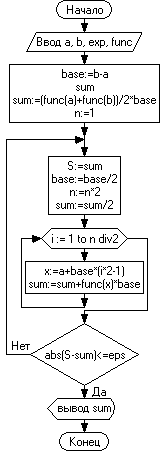
1. Исходные данные для тестирования приложения подготовить в текстовых файлах;
2. Результаты тестирования представить в элементах диалоговых форм;
3. Каждый метод вычисления с заданной точностью оформить в виде подпрограммы. Включить все подпрограммы в модуль, не связанный с формой.
   1. **Описание входных данных**

Входные данные предствлены в виде текстовых файлов, содержащих описание подынтегральной функции

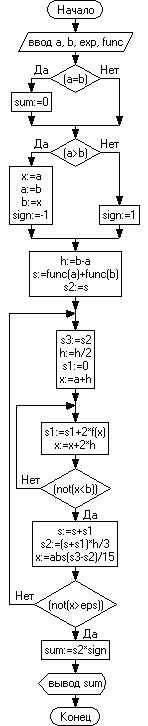
* 1. **Блок-схемы подпрограмм**
     1. **Блок-схемы Метод прямоугольников**

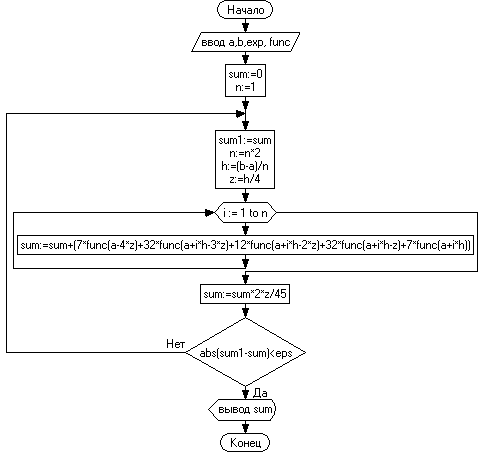


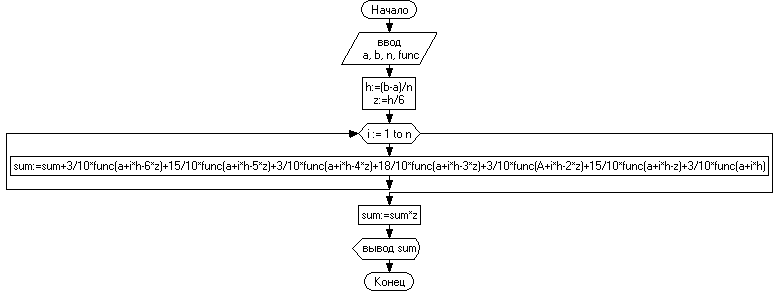
* + 1. **Метод трапеций**



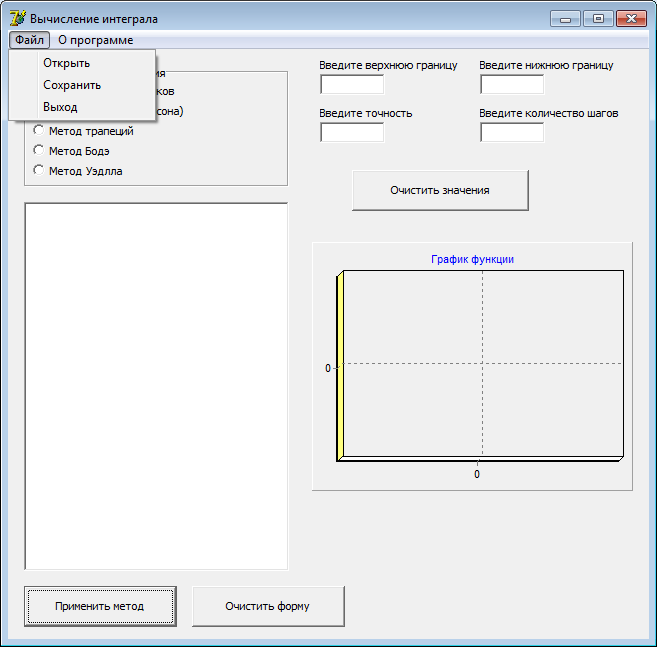
* + 1. **Метод парабол (Симпсона)**



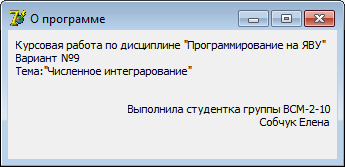
* + 1. **Метод Бодэ**
    2. **Метод Уэдлла**

****

* 1. **Разработка интерфейса**



*Рисунок 1 – Главная форма программы*



*Рисунок 2 – Форма «о программе»*

|  |  |  |  |
| --- | --- | --- | --- |
| *№* | *Объект* | *Наименование свойств объекта* | *Значение свойства объекта* |
| Главная форма | | | |
| 1 | Form1 | Caption | Вычисление интеграла |
| 2 | btn1 | Caption | Применить метод |
| 3 | btn2 | Caption | Очистить значение |
| 4 | btn3 | Caption | Очистить фоpму |
| 5 | cht1 | Button2 | Выход |
| 6 | digOpen1 | FilterName | Текстовые файлы |
| 7 | digOpen1 | Filter | \*txt |
| 8 | digSave1 | FilterName | Текстовые файлы |
| 9 | digSave1 | Filter | \*txt |
| 10 | Lbl1 | Caption | Введите верхнюю границу |
| 11 | Lbl2 | Caption | Введите нижнюю границу |
| 11 | Lbl3 | Caption | Введите точность |
| 12 | Lbl4 | Caption | Введите количество шагов |
| 13 | Mm1 | Items | Файл, Открыть, Сохранить, Выход, о программе |
| 14 | Rg1 | Caption | Выберите способ решения |
| 15 | Rg1 | Items | Метод прямоугольников  Метод парабол(Симпсона)  Метод трапеций  Метод Бодэ  Метод Уэдлла |
| Форма «о программе» | | | |
| 1 | Form2 | Caption | О программе |
| 2 | Lbl1 | Caption | Курсовая работа по дисциплине "Программирование на ЯВУ"  Вариант №9  Тема:"Численное интеграрование" |
| 3 | Lbl2 | Caption | Выполнила студентка группы ВСМ-2-10  Собчук Елена |

*Таблица 1 – Свойства объектов*

1. **ЭКСПЕРИМЕНТАЛЬНАЯ ЧАСТЬ**
   1. **Тестирование программы**

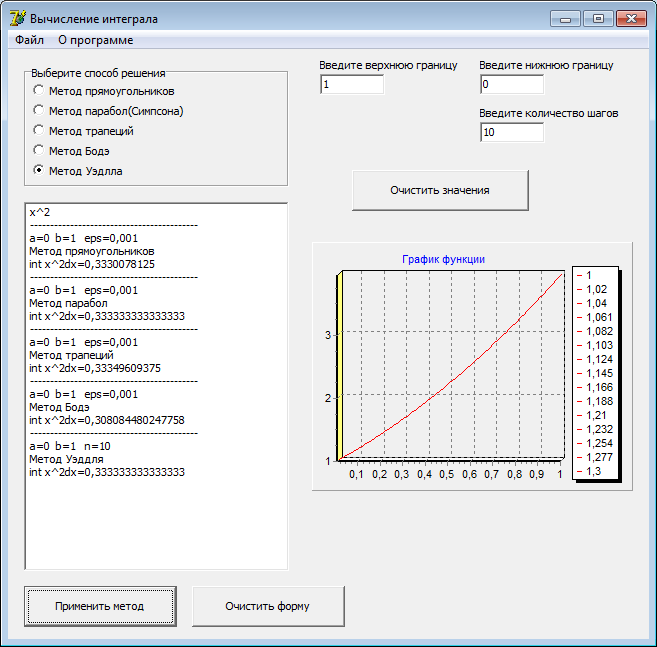
Тестирование оценивает корректность выполнения приложения БД. Программа тестируется с целью обнаружения ошибок, а также с целью проверки соответствия всем требованиям поставленной задачи.

**Тест №1**

Входные данные *2*

Эталон результата =0,33333

Результат программы представлен на рисунке 3.



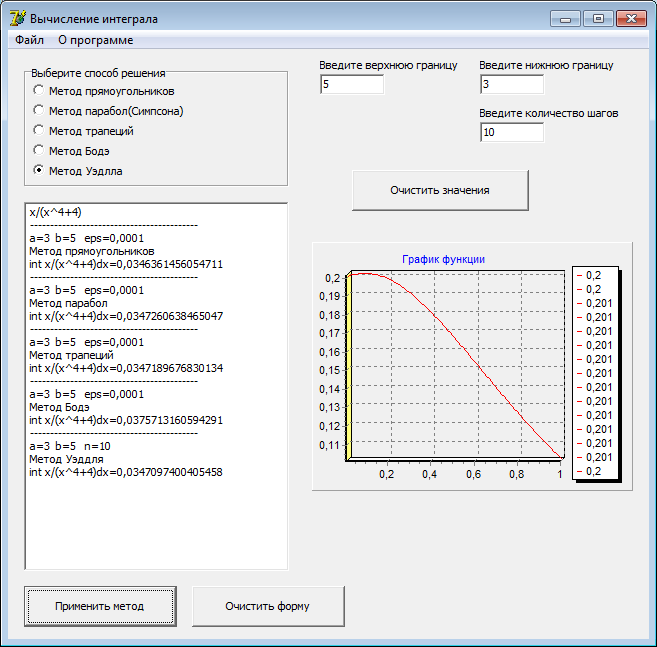
*Рисунок 3– вычисление 2dx*

**Тест №2**

Входные данные *dx*

Эталон результата =0,0347097

Результат программы представлен на рисунке 4.



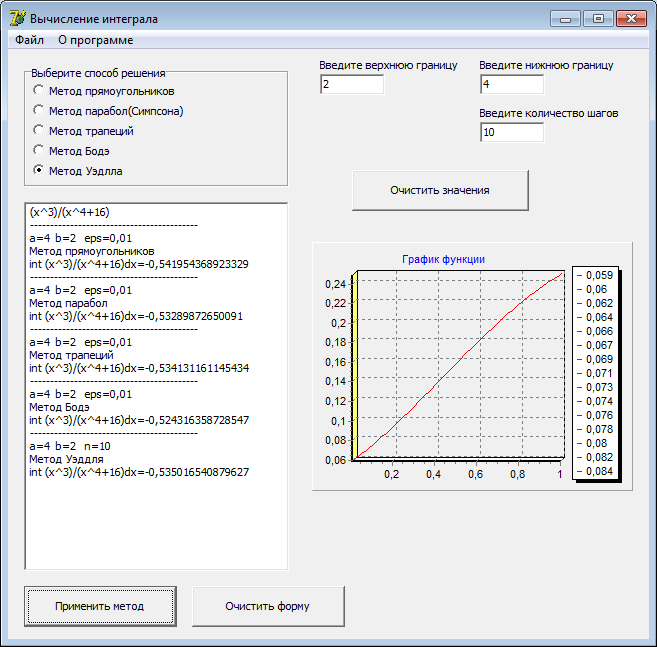
*Рисунок 4– вычисление dx*

**Тест №3**

Входные данные *dx*

Эталон результата =0,535017

Результат программы представлен на рисунке 5.



*Рисунок 5– вычисление dx*

## ЗАКЛЮЧЕНИЕ

Проблема повышения качества вычислений, как несоответствие между желаемым и действительным, существует и будет существовать в дальнейшем. Ее решению будет содействовать развитие информационных технологий, которое заключается как в совершенствовании методов организации информационных процессов, так и их реализации с помощью конкретных инструментов - сред и языков программирования.

Итогом работы можно считать созданную функциональную модель реализации численного интегрирования функции двух переменных. Созданная функциональная модель и ее программная реализация могут служить органической частью решения более сложных задач.

## СПИСОК ЛИТЕРАТУРЫ

1. Численное интегрирование [Электронный ресурс] - Режим доступа: <http://mathworld.wolfram.com/>
2. Дьяконов В.П. Справочник по расчетам на микрокалькуляторах М.: Наука, Гл. ред. физ-мат литературы, 1985. -224 с.
3. Л. Залогова Разработка Паскаль-компилятора М.:Бином. Лаборатория знаний , 2007. -184 с.
4. Васильев Ф.П. Численные методы решения экстремальных задач. [Текст] / Ф.П. Васильев - М.: Наука, 2002. - 415с.

## Приложение 1. Листинг программы.

program ProjectKurs;

uses

Forms,

Unit1 in 'Unit1.pas' {Form1},

evalcomp in 'evalcomp.pas',

Integration in 'Integration.pas',

Unit2 in 'Unit2.pas' {Form2};

{$R \*.res}

begin

Application.Initialize;

Application.CreateForm(TForm1, Form1);

Application.CreateForm(TForm2, Form2);

Application.Run;

end.

unit Unit1;

interface

uses

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

Dialogs, StdCtrls, ExtCtrls, Menus, evalcomp, Integration, TeEngine,

Series, TeeProcs, Chart;

type

TForm1 = class(TForm)

dlgOpen1: TOpenDialog;

rg1: TRadioGroup;

btn1: TButton;

mmo1: TMemo;

mm1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

dlgSave1: TSaveDialog;

N5: TMenuItem;

edt1: TEdit;

edt2: TEdit;

lbl1: TLabel;

lbl2: TLabel;

cht1: TChart;

fstlnsrsSeries1: TFastLineSeries;

lbl3: TLabel;

edt3: TEdit;

lbl4: TLabel;

edt4: TEdit;

btn2: TButton;

btn3: TButton;

procedure N2Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure btn1Click(Sender: TObject);

procedure edt1KeyPress(Sender: TObject; var Key: Char);

procedure edt2KeyPress(Sender: TObject; var Key: Char);

procedure edt3KeyPress(Sender: TObject; var Key: Char);

procedure edt4KeyPress(Sender: TObject; var Key: Char);

procedure changeMetod(Sender: TObject);

procedure btn2Click(Sender: TObject);

procedure btn3Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

uses Unit2;

var

formula:string;

{$R \*.dfm}

procedure TForm1.changeMetod(Sender: TObject);

begin

case (rg1.ItemIndex)of

4:

begin

edt3.Visible:=False;

lbl3.Visible:=False;

edt4.Visible:=True;

lbl4.Visible:=True;

end

else

begin

edt4.Visible:=False;

lbl4.Visible:=False;

edt3.Visible:=True;

lbl3.Visible:=True;

end;

end;

end;

procedure TForm1.N2Click(Sender: TObject); //загрузка из текстового файла

begin

if dlgOpen1.Execute then

mmo1.Lines.LoadFromFile(dlgOpen1.FileName);

formula:=mmo1.Lines.Text;

end;

procedure TForm1.N3Click(Sender: TObject); //сохранение результатов в текстовой файл

var

ext: String;

begin

ext:='.txt';

if dlgSave1.Execute then

mmo1.Lines.SaveToFile(dlgSave1.FileName+ext);

end;

procedure TForm1.N4Click(Sender: TObject); //закрытие программы

begin

Form1.Close

end;

procedure TForm1.N5Click(Sender: TObject); //вызов формы “о программе”

begin

if(not Assigned(Form2))then

Form2:=TForm2.Create(Self);

Form2.Show;

end;

Function StringIsEmpty (S :String) :Boolean; //проверка на данные

Var

N :Integer;

Begin

Result:=FALSE;

For N:=1 To Length(S) Do

If S[N]<>' 'Then

Begin

Result:=TRUE;

Break;

End

End;

procedure TForm1.btn1Click(Sender: TObject);

var x, y, g:real;

begin

if(StringIsEmpty(edt2.Text) and StringIsEmpty(edt1.Text) and (StringIsEmpty(edt3.Text) or StringIsEmpty(edt4.Text)) and StringIsEmpty(formula) and StringIsEmpty(mmo1.Text) ) then

begin

mmo1.Lines.Add('------------------------------------------');

case rg1.ItemIndex of

0:

begin

mmo1.Lines.Add('a='+edt2.Text+' '+'b='+edt1.Text+' '+'eps='+edt3.Text);

mmo1.Lines.Add('Метод прямоугольников');

mmo1.Lines.Add('int'+formula+('dx')+'='+ FloatToStr(SquareInt(StringToCursor(edt2.Text), StringToCursor(edt1.Text), StrToFloat(edt3.Text), formula)));

end;

1:

begin

mmo1.Lines.Add('a='+edt2.Text+' '+'b='+edt1.Text+' '+'eps='+edt3.Text);

mmo1.Lines.Add('Метод парабол');

mmo1.Lines.Add('int '+formula+('dx') +'='+ FloatToStr(SimpsonInt(StringToCursor(edt2.Text),StringToCursor(edt1.Text), StrToFloat(edt3.Text), formula)));

end;

2:

begin

mmo1.Lines.Add('a='+edt2.Text+' '+'b='+edt1.Text+' '+'eps='+edt3.Text);

mmo1.Lines.Add('Метод трапеций');

mmo1.Lines.Add('int '+formula+('dx') +'='+ FloatToStr(TrapezeInt(StringToCursor(edt2.Text),StringToCursor(edt1.Text), StrToFloat(edt3.Text), formula)));

end;

3:

begin

mmo1.Lines.Add('a='+edt2.Text+' '+'b='+edt1.Text+' '+'eps='+edt3.Text);

mmo1.Lines.Add('Метод Бодэ');

mmo1.Lines.Add('int '+formula+('dx') +'='+ FloatToStr(BodeInt(StringToCursor(edt2.Text),StringToCursor(edt1.Text), StrToFloat(edt3.Text), formula)));

end;

4:

begin

mmo1.Lines.Add('a='+edt2.Text+' '+'b='+edt1.Text+' '+'n='+edt4.Text);

mmo1.Lines.Add('Метод Уэддля');

mmo1.Lines.Add('int '+formula+('dx') +'='+ FloatToStr(UeddlsInt(StringToCursor(edt2.Text),StringToCursor(edt1.Text), StrToInt(edt4.Text), formula)));

end;

else

Application.Messagebox('Выберите метод','Ошибка', mb\_iconerror or mb\_iconerror );

end;

cht1.SeriesList[0].clear;

g:=0.01;

x:=0;

while (x<=1) do // Начинаем построение графика функции у=ln(х+1).

begin

y:=returnY(x, formula);

x:=x+g;

cht1.SeriesList[0].AddXY(x,y,'',clRed);

end;

end

else

Application.Messagebox('Введите параметры','Ошибка', mb\_iconerror or mb\_iconerror );

end;

procedure TForm1.edt1KeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9': ;

#8 : ;

#13 : edt2.SetFocus ;

else

Key :=Chr(0);

end;

end;

procedure TForm1.edt2KeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9': ;

#8 : ;

#13 : edt3.SetFocus ;

else

Key :=Chr(0);

end;

end;

procedure TForm1.edt3KeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9': ;

',': ;

#8 : ;

#13 : btn1.SetFocus ;

else

Key :=Chr(0);

end;

end;

procedure TForm1.edt4KeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9': ;

',': ;

#8 : ;

#13 : btn1.SetFocus ;

else

Key :=Chr(0);

end;

end;

procedure TForm1.btn2Click(Sender: TObject); //очиска полей

begin

edt1.Clear;

edt2.Clear;

edt3.Clear;

edt4.Clear;

end;

procedure TForm1.btn3Click(Sender: TObject);//очистка формы

begin

mmo1.Clear;

end;

end.

unit evalcomp; //модуль парсинга строки в мат формулу

interface

type fun= function(x,y:real):real;

evalvec= ^evalobj;

evalobj= object

f1,f2:evalvec;

f1x,f2y:real;

f3:fun;

function eval:real;

function eval1d(x:real):real;

function eval2d(x,y:real):real;

function eval3d(x,y,z:real):real;

constructor init(st:string);

destructor done;

end;

var evalx,evaly,evalz:real;

implementation

var analysetmp:fun;

function search (text,code:string; var pos:integer):boolean;

var

i,l,count:integer;

flag:boolean;

newtext:string;

begin

if length(text)<length(code) then begin search:=false; exit; end;

flag:=false;

pos:=length(text)-length(code)+1;

repeat

if code=copy(text,pos,length(code))

then flag:=true

else dec(pos);

if flag then

begin

count:=0;

for i:= pos+1 to length(text) do

begin

If copy(text,i,1) = '(' then inc(count);

if copy(text,i,1) = ')' then dec(count);

end;

if count<>0 then

begin

dec(pos);

flag:=false;

end;

end;

until (flag=true) or (pos=0);

search:=flag;

end;

//далее функции преобразования найденных строк-формул в формулы

function myid(x,y:real):real;

begin

myid:=x;

end;

function myunequal(x,y:real):real;

begin

if x<>y then

myunequal:=1

else

myunequal:=0;

end;

function mylessequal(x,y:real):real;

begin

if x<=y then

mylessequal:=1

else

mylessequal:=0;

end;

function mygreaterequal(x,y:real):real;

begin

if x>=y then

mygreaterequal:=1

else

mygreaterequal:=0;

end;

function mygreater(x,y:real):real;

begin

if x>y then

mygreater:=1

else

mygreater:=0;

end;

function myless(x,y:real):real;

begin

if x<y then

myless:=1

else

myless:=0;

end;

function myequal(x,y:real):real;

begin

if x=y then

myequal:=1

else

myequal:=0;

end;

function myadd(x,y:real):real;

begin

myadd:=x+y;

end;

function mysub(x,y:real):real;

begin

mysub:=x-y;

end;

function myeor(x,y:real):real;

begin

myeor:=trunc(x) xor trunc(y);

end;

function myor(x,y:real):real;

begin

myor:=trunc(x) or trunc(y);

end;

function mymult(x,y:real):real;

begin

mymult:=x\*y;

end;

function mydivid(x,y:real):real;

begin

mydivid:=x/y;

end;

function myand(x,y:real):real;

begin

myand:=trunc(x) and trunc(y);

end;

function mymod(x,y:real):real;

begin

mymod:=trunc(x) mod trunc(y);

end;

function mydiv(x,y:real):real;

begin

mydiv:=trunc(x) div trunc(y);

end;

function mypower(x,y:real):real;

begin

if x=0 then

mypower:=0

else

if x>0 then

mypower:=exp(y\*ln(x))

else

if trunc(y)<>y then

begin

break;

end

else

if odd(trunc(y))=true then

mypower:=-exp(y\*ln(-x))

else

mypower:=exp(y\*ln(-x))

end;

function myshl(x,y:real):real;

begin

myshl:=trunc(x) shl trunc(y);

end;

function myshr(x,y:real):real;

begin

myshr:=trunc(x) shr trunc(y);

end;

function mynot(x,y:real):real;

begin

mynot:=not trunc(x);

end;

function mysinc(x,y:real):real;

begin

if x=0 then

mysinc:=1

else

mysinc:=sin(x)/x

end;

function mysinh(x,y:real):real;

begin

mysinh:=0.5\*(exp(x)-exp(-x))

end;

function mycosh(x,y:real):real;

begin

mycosh:=0.5\*(exp(x)+exp(-x))

end;

function mytanh(x,y:real):real;

begin

mytanh:=mysinh(x,0)/mycosh(x,0)

end;

function mycoth(x,y:real):real;

begin

mycoth:=mycosh(x,0)/mysinh(x,0)

end;

function mysin(x,y:real):real;

begin

mysin:=sin(x)

end;

function mycos(x,y:real):real;

begin

mycos:=cos(x)

end;

function mytan(x,y:real):real;

begin

mytan:=sin(x)/cos(x)

end;

function mycot(x,y:real):real;

begin

mycot:=cos(x)/sin(x)

end;

function mysqrt(x,y:real):real;

begin

mysqrt:=sqrt(x)

end;

function mysqr(x,y:real):real;

begin

mysqr:=sqr(x)

end;

function myarcsinh(x,y:real):real;

begin

myarcsinh:=ln(x+sqrt(sqr(x)+1))

end;

function mysgn(x,y:real):real;

begin

if x=0 then

mysgn:=0

else

mysgn:=x/abs(x)

end;

function myarccosh(x,y:real):real;

begin

myarccosh:=ln(x+mysgn(x,0)\*sqrt(sqr(x)-1))

end;

function myarctanh(x,y:real):real;

begin

myarctanh:=ln((1+x)/(1-x))/2

end;

function myarccoth(x,y:real):real;

begin

myarccoth:=ln((1-x)/(1+x))/2

end;

function myarcsin(x,y:real):real;

begin

if x=1 then

myarcsin:=pi/2

else

myarcsin:=arctan(x/sqrt(1-sqr(x)))

end;

function myarccos(x,y:real):real;

begin

myarccos:=pi/2-myarcsin(x,0)

end;

function myarctan(x,y:real):real;

begin

myarctan:=arctan(x);

end;

function myarccot(x,y:real):real;

begin

myarccot:=pi/2-arctan(x)

end;

function myheavy(x,y:real):real;

begin

myheavy:=mygreater(x,0)

end;

function myfrac(x,y:real):real;

begin

myfrac:=frac(x)

end;

function myexp(x,y:real):real;

begin

myexp:=exp(x)

end;

function myabs(x,y:real):real;

begin

myabs:=abs(x)

end;

function mytrunc(x,y:real):real;

begin

mytrunc:=trunc(x)

end;

function myln(x,y:real):real;

begin

myln:=ln(x)

end;

function myodd(x,y:real):real;

begin

if odd(trunc(x)) then

myodd:=1

else

myodd:=0;

end;

function mypred(x,y:real):real;

begin

mypred:=pred(trunc(x));

end;

function mysucc(x,y:real):real;

begin

mysucc:=succ(trunc(x));

end;

function myround(x,y:real):real;

begin

myround:=round(x);

end;

function myint(x,y:real):real;

begin

myint:=int(x);

end;

function myfac(x,y:real):real;

var

n : integer;

r : real;

begin

if x<0 then begin

halt;

end;

If x = 0 then myfac := 1

else

begin

r := 1;

for n := 1 to trunc ( x ) do

r := r \* n;

myfac:= r;

end;

end;

function myrnd(x,y:real):real;

begin

myrnd:=random;

end;

function myrandom(x,y:real):real;

begin

myrandom:=random(trunc(x));

end;

function myevalx(x,y:real):real;

begin

myevalx:=evalx;

end;

function myevaly(x,y:real):real;

begin

myevaly:=evaly;

end;

function myevalz(x,y:real):real;

begin

myevalz:=evalz;

end;

//анализ строк

procedure analyse (st:string; var st2,st3:string);

label start;

var pos:integer;

value:real;

newterm,term:string;

begin

term:=st;

start:

if term='' then begin analysetmp:=myid; st2:='0'; st3:=''; exit; end;

newterm:='';

for pos:= 1 to length(term) do

if copy(term,pos,1)<>' ' then newterm:=newterm+copy(term,pos,1);

term:=newterm;

if term='' then begin analysetmp:=myid; st2:='0'; st3:=''; exit; end;

val(term,value,pos);

if pos=0 then begin

analysetmp:=myid;

st2:=term;

st3:='';

exit;

end;

if search(term,'<>',pos) then begin

analysetmp:=myunequal;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+2,length(term)-pos-1);

exit;

end;

if search(term,'<=',pos) then begin

analysetmp:=mylessequal;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+2,length(term)-pos-1);

exit;

end;

if search(term,'>=',pos) then begin

analysetmp:=mygreaterequal;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+2,length(term)-pos-1);

exit;

end;

if search(term,'>',pos) then begin

analysetmp:=mygreater;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'<',pos) then begin

analysetmp:=myless;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'=',pos) then begin

analysetmp:=myequal;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'+',pos) then begin

analysetmp:=myadd;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'-',pos) then begin

analysetmp:=mysub;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'eor',pos) then begin

analysetmp:=myeor;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if search(term,'or',pos) then begin

analysetmp:=myor;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+2,length(term)-pos-1);

exit;

end;

if search(term,'\*',pos) then begin

analysetmp:=mymult;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'/',pos) then begin

analysetmp:=mydivid;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'and',pos) then begin

analysetmp:=myand;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if search(term,'mod',pos) then begin

analysetmp:=mymod;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if search(term,'div',pos) then begin

analysetmp:=mydiv;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if search(term,'^',pos) then begin

analysetmp:=mypower;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+1,length(term)-pos);

exit;

end;

if search(term,'shl',pos) then begin

analysetmp:=myshl;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if search(term,'shr',pos) then begin

analysetmp:=myshr;

st2:=copy(term,1,pos-1);

st3:=copy(term,pos+3,length(term)-pos-2);

exit;

end;

if copy(term,1,1)='(' then begin

term:=copy(term,2,length(term)-2);

goto start;

end;

if copy(term,1,3)='not' then begin

analysetmp:=mynot;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,4)='sinc' then begin

analysetmp:=mysinc;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,4)='sinh' then begin

analysetmp:=mysinh;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,4)='cosh' then begin

analysetmp:=mycosh;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,4)='tanh' then begin

analysetmp:=mytanh;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,4)='coth' then begin

analysetmp:=mycoth;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,3)='sin' then begin

analysetmp:=mysin;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,3)='cos' then begin

analysetmp:=mycos;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,3)='tan' then begin

analysetmp:=mytan;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,3)='cot' then begin

analysetmp:=mycot;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,4)='sqrt' then begin

analysetmp:=mysqrt;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,3)='sqr' then begin

analysetmp:=mysqr;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,7)='arcsinh' then begin

analysetmp:=myarcsinh;

st2:=copy(term,8,length(term)-7);

st3:='';

exit;

end;

if copy(term,1,7)='arccosh' then begin

analysetmp:=myarccosh;

st2:=copy(term,8,length(term)-7);

st3:='';

exit;

end;

if copy(term,1,7)='arctanh' then begin

analysetmp:=myarctanh;

st2:=copy(term,8,length(term)-7);

st3:='';

exit;

end;

if copy(term,1,7)='arccoth' then begin

analysetmp:=myarccoth;

st2:=copy(term,8,length(term)-7);

st3:='';

exit;

end;

if copy(term,1,6)='arcsin' then begin

analysetmp:=myarcsin;

st2:=copy(term,7,length(term)-6);

st3:='';

exit;

end;

if copy(term,1,6)='arccos' then begin

analysetmp:=myarccos;

st2:=copy(term,7,length(term)-6);

st3:='';

exit;

end;

if copy(term,1,6)='arctan' then begin

analysetmp:=myarctan;

st2:=copy(term,7,length(term)-6);

st3:='';

exit;

end;

if copy(term,1,6)='arccot' then begin

analysetmp:=myarccot;

st2:=copy(term,7,length(term)-6);

st3:='';

exit;

end;

if copy(term,1,5)='heavy' then begin

analysetmp:=myheavy;

st2:=copy(term,6,length(term)-5);

st3:='';

exit;

end;

if copy(term,1,3)='sgn' then begin

analysetmp:=mysgn;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,4)='frac' then begin

analysetmp:=myfrac;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,3)='exp' then begin

analysetmp:=myexp;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,3)='abs' then begin

analysetmp:=myabs;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,5)='trunc' then begin

analysetmp:=mytrunc;

st2:=copy(term,6,length(term)-5);

st3:='';

exit;

end;

if copy(term,1,2)='ln' then begin

analysetmp:=myln;

st2:=copy(term,3,length(term)-2);

st3:='';

exit;

end;

if copy(term,1,3)='odd' then begin

analysetmp:=myodd;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,4)='pred' then begin

analysetmp:=mypred;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,4)='succ' then begin

analysetmp:=mysucc;

st2:=copy(term,5,length(term)-4);

st3:='';

exit;

end;

if copy(term,1,5)='round' then begin

analysetmp:=myround;

st2:=copy(term,6,length(term)-5);

st3:='';

exit;

end;

if copy(term,1,3)='int' then begin

analysetmp:=myint;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if copy(term,1,3)='fac' then begin

analysetmp:=myfac;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if term='rnd' then begin

analysetmp:=myrnd;

st2:='';

st3:='';

exit;

end;

if copy(term,1,3)='rnd' then begin

analysetmp:=myrandom;

st2:=copy(term,4,length(term)-3);

st3:='';

exit;

end;

if term='x' then begin

analysetmp:=myevalx;

st2:='';

st3:='';

exit;

end;

if term='y' then begin

analysetmp:=myevaly;

st2:='';

st3:='';

exit;

end;

if term='z' then begin

analysetmp:=myevalz;

st2:='';

st3:='';

exit;

end;

if (term='pi') then begin

analysetmp:=myid;

str(pi,st2);

st3:='';

exit;

end;

if term='e' then begin

analysetmp:=myid;

str(exp(1),st2);

st3:='';

exit;

end;

analysetmp:=myid;

st2:='';

st3:='';

end;

function evalobj.eval:real;

var

tmpx,tmpy:real;

begin

if f1=nil then

tmpx:=f1x

else

tmpx:=f1^.eval;

if f2=nil then

tmpy:=f2y

else

tmpy:=f2^.eval;

eval:=f3(tmpx,tmpy);

end;

function evalobj.eval1d(x:real):real;

begin

evalx:=x;

evaly:=0;

evalz:=0;

eval1d:=eval;

end;

function evalobj.eval2d(x,y:real):real;

begin

evalx:=x;

evaly:=y;

evalz:=0;

eval2d:=eval;

end;

function evalobj.eval3d(x,y,z:real):real;

begin

evalx:=x;

evaly:=y;

evalz:=z;

eval3d:=eval;

end;

constructor evalobj.init(st:string);

var

st2,st3:string;

error:integer;

begin

f1:=nil;

f2:=nil;

analyse(st,st2,st3);

f3:=analysetmp;

val(st2,f1x,error);

if st2='' then

begin

f1x:=0;

error:=0;

end;

if error<>0 then

new (f1,init(st2));

val(st3,f2y,error);

if st3='' then

begin

f2y:=0;

error:=0;

end;

if error<>0 then

new (f2,init(st3));

end;

destructor evalobj.done;

begin

if f1<>nil then

dispose(f1,done);

if f2<>nil then

dispose(f2,done);

end;

end.

unit Integration;

interface

uses evalcomp;

function f(x:real):real;

function TrapezeInt(a,b:Double; eps: Double; StrF: string): Double;

function SquareInt(a,b:Double; eps: Double; StrF: string): Double;

function SimpsonInt(a,b:Double; eps: Double; StrF: string): Double;

function BodeInt(a,b:Double; eps: Double;StrF:string): Double;

function UeddlsInt(a,b:Double; n: Integer;StrF:string): Double;

function returnY(a:Double;StrF:string): Double;

implementation

var

calc:EVALVEC;

FORMULA: string;

function f(x:real):real;

begin

f:=calc^.eval1d(x);

end;

function returnY(a:Double;StrF:string): Double;

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

Result:=f(a+1);

dispose(calc,done);

end;

// Вычисление интеграла методом Уэддля

function UeddlsInt(a,b:Double; n: integer;StrF:string): Double;

var

h,s,z: real; i: integer;

const k: array [1..7] of real = (3/10,15/10,3/10,18/10,3/10,15/10,3/10);

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

h:=(b-a)/n;

s:=0;

z:=h/6;

for i:=1 to n do s:=s+k[1]\*f(a+i\*h-6\*z)+k[2]\*f(a+i\*h-5\*z)+k[3]\*f(a+i\*h-4\*z)+k[4]\*f(a+i\*h-3\*z)+k[5]\*f(A+i\*h-2\*z)+k[6]\*F(a+i\*h-z)+k[7]\*F(a+i\*h);

s:=s\*z;

Result:=s;

dispose(calc,done);

end;

// метод Боде

function BodeInt(a,b:Double; eps: Double;StrF:string): Double;

var

h,s,sum1,z: real;

i,n: integer;

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

s:=0; n:=1;

repeat

sum1:=s;

n:=n\*2;

h:=(b-a)/n;

z:=h/4;

for i:=1 to n do

s:=s+(7\*f(a-4\*z)+32\*f(a+i\*h-3\*z)+12\*f(a+i\*h-2\*z)+32\*f(a+i\*h-z)+7\*f(a+i\*h));

s:=s\*2\*z/45;

until abs(sum1-s)<eps;

Result:=s;

dispose(calc,done);

end;

//метод прямоугольников

function SquareInt(a,b:Double; eps: Double; StrF:string): Double;

var

i, n: Integer;

h, s1, s2 : Double;

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

n := 1;

h := b-a;

s2 := h\*f((a+b)/2);

repeat

n := 2\*n;

s1 := s2;

h := h/2;

s2 := 0;

i := 1;

repeat

s2 := s2+f(a+h/2+h\*(i-1));

i := i+1;

until not (i<=n);

s2 := s2\*h;

until not (Abs(s2-s1)>3\*eps);

Result := s2;

dispose(calc,done);

end;

//Метод трапеций

function TrapezeInt(a,b:Double; eps: Double; StrF: string): Double;

var

S, x, base: Double;

i, n: Integer;

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

base := b - a;

Result := (f(a) + f(b)) / 2 \* base;

n := 1;

repeat

S := Result;

base := base / 2;

n := n \* 2;

Result := Result / 2;

for i := 1 to n div 2 do

begin

x := a + base \* (i \* 2 - 1);

Result := Result + f(x) \* base;

end;

until abs(S - Result) <= eps;

dispose(calc,done);

end;

//Метод парабол

function SimpsonInt(a,b:Double; eps: Double; StrF: string): Double;

var

h, x, s, s1, s2, s3, sign: double;

begin

FORMULA:=StrF;

new (calc,init(FORMULA));

if (a = b) then

begin

Result := 0; exit

end;

if (a > b) then

begin

x := a; a := b; b := x; sign := -1

end

else sign:=1;

h := b - a; s := f(a) + f(b); s2 := s;

repeat

s3 := s2; h := h/2; s1 := 0; x := a + h;

repeat

s1 := s1 + 2\*f(x); x := x + 2\*h;

until (not(x < b));

s := s + s1; s2 := (s + s1)\*h/3; x := abs(s3 - s2)/15

until (not(x > eps));

Result := s2\*sign;

dispose(calc,done);

end;

end.