КОНТРОЛЬНАЯ РАБОТА №1

**3, 4 Вариант.**

**Задание 1**

###### Вычисление значения аналитического выражения

**(линейный алгоритм)**

**Цель работы**: составить программу для вычисления значения функции y(x) при заданном значении аргумента x; вывести на экран значения аргумента и функции.

**Элементы языка Turbo Pascal**

**Алфавит**

Алфавит Turbo Pascal включает:

1. заглавные и строчные латинские буквы от A до Z и от a до z, символ подчеркивания \_;
2. арабские цифры от 0 до 9;
3. специальные символы: + - \* / = , ‘ . : ; < > [ ] ( ) { } ^ @ $ # и пары символов <> <= >= := ;
4. ключевые (зарезервированные) слова, например:

and

array

begin

const

var

end

for

function

label;

ключевые слова не могут входить в состав идентификаторов.

**Идентификаторы (имена)**

Идентификаторы – имена констант, переменных, меток, типов, процедур и функций. Идентификатор всегда начинается буквой или знаком подчеркивания, за которым могут следовать буквы, цифры и знак подчеркивания. Пробелы и специальные символы алфавита не могут входить в идентификатор.

Примеры правильных идентификаторов:

a

ALPHA

date\_27\_08\_98

\_beta .

**Константы и переменные**

Константы – данные, значения которых не изменяются в процессе выполнения программы. Значения переменных во время выполнения программы могут изменяться.

**Константы**

В Turbo Pascal используются три вида констант:

- числовые (целые и вещественные);

- логические;

- символьные и строковые.

Целые константы – целые числа (без точки). Знак + можно опускать. Вещественные константы могут быть представлены в двух видах – с фиксированной и плавающей точкой.

Пример: Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| Значение  константы | Пример записи | | |
| целая константа | конст.с фикс.точкой | конст. с плав. точкой |
| -257 | -257 | -257.0 | -2.57e2 |
| 16.4 | - | 16.4 | 1.64e1 |

Запись –2.57e2 означает “ –2.57 умножить на 10 в степени 2” Если в записи вещественного числа присутствует десятичная точка, перед точкой и за ней должно быть хотя бы по одной цифре.

Логическая константа – константа, принимающая либо значение FALSE (ложь) либо значение TRUE (истина).

Символьная константа – один любой символ, заключенный в апострофы:

‘a’ , ‘5’ , ‘!’.

Строковая константа – любая последовательность символов, заключенная в апострофы: ‘ABCD’ , ‘100OK’ .

**Переменные**

Переменная целого типа (типа integer) может принимать значения в диапазоне -32768...32767.

Переменная вещественного типа (типа real) принимает значения в диапазоне 10-38..1038.

Переменная логического типа (типа boolean) может принимать значения FALSE либо TRUE.

Переменная символьного типа (типа char) может принимать значения только одного символа.

**Операции** Таблица 2

Арифметические операции

|  |  |  |
| --- | --- | --- |
| Операция | Действие | Пример записи |
| + | сохранение знака | +A |
| - | изменение знака | -A |
| + | сложение | A+B |
| - | вычитание | A-B |
| \* | умножение | A\*B |
| / | деление | A/B |
| div | целочисленное деление | A div B, 3 div 2 |
| mod | остаток от целочисл. деления | A mod B, 5 mod 3 |

В любом выражении, если один или более операндов имеют вещественный тип, то результат будет вещественным. В операциях **div** и **mod** оба операнда (A и B) должны быть целого типа.

Таблица 3

Логические операции

|  |  |  |
| --- | --- | --- |
| Операция | Действие | Пример записи |
| not | отрицание | not A |
| and | логическое “И” | A and B |
| or | логическое “ИЛИ” | A or B |
| xor | исключающее “ИЛИ” | A xor B |

Т

Таблица 4

Операции отношения

|  |  |  |
| --- | --- | --- |
| Операция | Действие | Пример записи |
| = | равно | A=B |
| <> | не равно | A<>B |
| < | меньше | A<B |
| > | больше | A>B |
| <= | меньше либо равно | A<=B |
| >= | больше либо равно | A>=B |

При выполнении операций отношения оба операнда (A,B) должны быть одного и того же типа. Допускается лишь одно исключение: A – целого типа, B – вещественного (и наоборот).

**Выражения**

Выражения представляют собой правила получения новых значений. Арифметические выражения строятся из числовых констант, переменных, стандартных функций и арифметических операций над ними. В арифметическом выражении принят следующий приоритет операций (в порядке убывания приоритета):

1. вычисление значений стандартных функций;
2. умножение и деление;
3. сложение и вычитание.

Порядок выполнения операций может регулироваться с помощью скобок.

Примеры арифметических выражений:

a+b

(a+b)\*c

sin(t) .

Логические выражения строятся из логических констант и переменных, логических операций и операций отношения. В операциях отношения могут участвовать арифметические и логические выражения, а также символьные данные. Результатом логического выражения является значение TRUE (истинно) или FALSE (ложно). При вычислении логических выражений принят такой приоритет операций (в порядке убывания приоритета):

1. not;
2. \* , / , div, mod, and;
3. + , - , or, xor;
4. операции отношения.

Примеры логических выражений: not M

(B or C) and (D or E) .

В языке Turbo Pascal имеются стандартные (встроенные) функции, наиболее часто используемые из них приведены в табл.5:

Таблица 5

|  |  |  |
| --- | --- | --- |
| Функция,  запись на языке Turbo Pascal | Назначение | Математическая форма  записи |
| Abs(x) | вычисление абсолютного значения x | |x| |
| Sqr(x) | вычисление квадрата x | x2 |
| Sin(x) | вычисление синуса x | sin x |
| Cos(x) | вычисление косинуса x | cos x |
| Arctan(x) | вычисление арктангенса x | arctg x |
| Exp(x) | вычисление экспоненты x | ex |
| Ln(x) | вычисление натурального логарифма x | ln x |
| Sqrt(x) | вычисление квадратного корня из x |  |
| Trunc(x) | вычисление целой части x |  |
| Round(x) | округление x в сторону ближайшего целого |  |
| Odd(x) | TRUE, если x – нечетное,  FALSE, если x - четное |  |

В функциях синуса и косинуса аргумент x должен быть задан в радианах. Если x задан в градусах, то для перевода его в радианы используется формула: x = x⋅π/180.

Логарифм с основанием a вычисляется по формуле:

loga(x) = ln(x)/ln(a) .

Для возведения x в степень a используется соотношение:

xa = ea⋅lnx .

**Операторы**

Операторы указывают, какие алгоритмические действия необходимо выполнить над исходными данными для получения результата.

Для ввода исходных данных используются **операторы ввода**:

**read** **(**список\_переменных**);** – каждое вводимое (с клавиатуры) значение последовательно присваивается переменным из списка;

**readln (**список\_переменных**);** – то же, что и оператор **read**, только после ввода данных происходит переход на новую строку (т.е. следующий оператор ввода будет вводить данные с новой строки);

**readln;** – происходит переход на новую строку без ввода данных.

Значения вводимых переменных должны соответствовать типам переменных из списка\_переменных. Допускается вводить значения: целых (integer) , вещественных (real), символьных (char) данных.

Пример. Введем значения переменных A=0.5; B=6.25; C=-0.71 и с новой строки введем I=1, K=5 : **readln(**A,B,C**);** **read(**I,K**);** .

Для присваивания переменной нового значения служит **оператор присваивания**. Его общий вид: имя **:=** выражение**;**

где имя – имя переменной, текущее значение которой заменяется новым значением, определяемым данным выражением. Пример: y:=Sqrt(x)+1; - переменной y присваивается значение, которое будет получено в результате вычисления выражения +1.

В операторе присваивания переменная и выражение должны иметь один и тот же тип. Разрешается присваивать переменной типа real выражение типа integer.

Для вывода результатов служат **операторы вывода**:

**write(**список\_переменных**);** - выводит последовательно значения из списка переменных;

**writeln(**список\_переменных**);** - то же, что и **write**, но после вывода переменных осуществляется переход на новую строку;

**writeln** - осуществляет переход на новую строку без ввода данных.

Допустим вывод значений следующих данных:

1. целых (integer), вещественных (real), символьных (char), логических (boolean) переменных;
2. символьных констант;
3. арифметических и логических выражений.

Пример. Выведем на экран значение переменной y :

write(y);

или write(Sqrt(x)+1));

или write(‘y=’,y);

В Turbo Pascal предусмотрен форматный вывод данных: write(y:m:n);

где m – общее число позиций для выводимой величины y;

n – число позиций дробной части.

**Структура программы**

В Turbo Pascal программа имеет следующую структуру:

**PROGRAM** имя;

**CONST**  {раздел констант};

**……**  раздел описаний

**VAR**  {раздел переменных};

#### BEGIN

оператор\_1;

оператор\_2; раздел операторов

……………;

оператор\_n

**END**.

Любая программа начинается со слова PROGRAM и заканчивается точкой. Имя – имя программы, образуемое так же, как и имена переменных. В разделе описаний должны быть описаны все константы и переменные, используемые в программе. Раздел описаний предшествует разделу операторов. Раздел операторов начинается с ключевого слова BEGIN и заканчивается словом END с точкой и содержит исполняемые операторы, отделяющиеся друг от друга знаком “**;**” (точкой с запятой).

Указания для выполнения основных действий, необходимых для создания программ и проведения расчетов в среде Pascal, даны в приложении (см стр. 40–47).

Пример. Вычислим значение выражения



при c=1.5, d=0.6, x=0.64.

Составим программу:

**PROGRAM** L1; *{L1-имя программы}*

**CONST**  c=1.5;d=0.6; *{величины c и d описываем константами}*

**VAR**  x,y:real; *{описываем x и y переменными вещественного типа}***BEGIN** read(x); *{вводим с клавиатуры значение аргумента x}*

y:=Sqrt(c\*x)-2.7\*(Abs(c)+Abs(x))\*Exp(d\*x)/ *{вычисляем значение выражения-}*

(Sqr(c\*x)+1)+Cos(Sqr(x)); *{-и присваиваем его переменной y}*

write(' x=',x:5:2,' y(x)=',y:6:2) *{выводим на экран значения x и y}*

**END**.

Результат работы программы получим в виде

**x= 0.64 y(x)= -2.52.**

**Варианты заданий**

1. 
2. 
3. 
4. 
5. 
6. 
7. 

8) 

9) 

10) 

11) 

1. 

13) 

1. 
2. 

16) 

17) 

1. 
2. 

20) 

21)



22) 

23) 

24) 

25) 

**Задание 2**

###### Вычисление значения функции, заданной различными аналитическими выражениями, при заданных значениях аргумента

###### (разветвляющийся алгоритм)

**Цель работы**: оставить программу для вычисления значений функции y(x); вывести на экран значения x и y(x).

## Оператор условного перехода

Для программирования разветвляющегося алгоритма используется **оператор условного** **перехода**. Его общий вид:

**if** логическое выражение **then** оператор\_ 1 [**else**  оператор\_2];

Квадратные скобки означают, что конструкция **else** оператор\_2 может отсутствовать. Если логическое\_выражение истинно, то выполняется оператор\_1, иначе – оператор\_2 (или следующий за **if**  оператор, если **else** отсутствует).

Пример. Вычислим значение функции:

sinx, если x≤a,

y(x)= cosx, если a<x<b,

tgx , если x≥b .

Указанное выражение может быть запрограммировано в виде:

**if** x<=a **then** y:=sin(x);**if** (x>a)and(x<b) **then** y:=cos(x);**if** x>=b **then** y:=sin(x)/cos(x);

или

**if** x<=a **then** y:=sin(x) **else** **if** x<b **then** y:=cos(x)

**else** y:=sin(x)/cos(x); .

## Варианты заданий

Значения x (по одному значению для каждого промежутка) задать самостоятельно.

1) ab-x3, если a<x<b, 2) 7(b/x-a/b)2, если x>a,

y(x)= 4(x2+a), если x≥b, y(x)= 2(a-b)3(ax+bc), если x<a,

x2(a+b), если x≤a; 3abx, если x=a;

a=-0.0000124; b=2/5; a=2; b=3.5; c=4.1;

|  |  |  |
| --- | --- | --- |
| 3) cos(ax2+b)2, если -19≤x<-16,  y(x)= 5(bx+|a+d|), если x≥-16,  2xd+a2, если x<-19;  a=-9/2; b=5.61; d=24; | | 4) a2b/9-|x|, если x≤-2,  y(x)= 4(a3x2-c2), если -2<x<4,  tg[(a-b)2+x2], если x≥4;  a=-0.0025; b=-2.7;c=3; |
| 5) , если -1≤x<a,  y(x) = 10(|ax|+bx), если x>a,  ax2, если x<-1;  a=21.345; b=-2/5;c=1; | | 6) x-a, если x<-1,  y(x)= , если -1≤x≤1,  9x2-a, если x>1;  a=-1.5; |
| 7) , если -2<x≤3,  y(x)= ab/x-x2, если x>3,  4(x+a/b), если x≤-2;  a=-0.024; b=0.752479; | 8) 5|b-a|/(|b|-|c|)x2, если 3≤x≤5,  y(x)= 2.5(|x|-c)(ax+b), если x<3,  102dbc/x, если x>5;  a=7.4; b=2.5; c=3.1; d=0.5; | |
| 9) 5c2d2b, если x=3.5,  y(x)= 4(cd2+x2), если 1.5≤x<3.5,  (ax+b)2, если x>3.5;  a=4.7; b=2.1; c=1.7; d=0.5; | 10) 4(x/a2-a), если x<-1,  y(x)= , если -1≤x≤1,  x3, если x>1;  a=12.1; | |
| 11) x2+cos(a+bx), если x<0,  y(x) = 4(x-1), если x>1,  (x+2)+0.0024, если 0≤x≤1;    a=-0.24; b=2; | 12) (a2-0.008b2)/x, если a<x≤b,  y(x)= bx-a, если x≤a,  (a+b)cosx, если x>b;  a=-0.2; b=2; | |
| 13) 0.003a/x, если x≤c,  y(x) = 2x/a, если x≥a+b,  c(ax+b)2, если c<x<a+b;    a=0.8; b=-0.26; c=-0.24; | 14) |a|b-x2, если x≤0,  y(x)= a2x+b2, если 0<x≤0.1,  b(x2-a2), если x>0.1;  a=2/7; b=-3.01; | |
| 15) a3+dx2, если 0.1≤x≤a,  y(x) = (b-x)2, если x>a,  (b2+x2)a-x, если x<0.1;  a=2.51; b=-3.06; d=5/3; | 16) x2+a, если x<a,  y(x)= 8.4(x+a), если a≤x≤d,  dx/15-ax, если x>d;  a=0.2; d=1.9; | |
| 17) eax, если x<-1,  y(x)= cosx+0.0085, если -1≤x<0,  4x2-a, если x≥0;  a=29.4; | 18) (ax2+2)/(x2+1), если 1<|x|<3,  y(x)= a, если |x|≥3,  ax/(x+2), если |x|≤1;  a=9/4; | |
| 19) (|a|-|b|)/(ab+x)/c, если 3<x≤5,  y(x)= (a-b)2x3, если 5<x<9.8,  (ax2+b)/4x, если x≥9.8;  a=-19.2457; b=24.7; c=1.5; | 20) 4[ax-6(a+b)3], если a+b≤x<a,  y(x) = a2+(a-b)+x2, если x≥a,  x2(a+b), если x<a+b;  a=2.005; b=-0.0009; | |
| 21) a2(a-b)x, если |a|<x≤|b|,  y(x) = 5(ax2+b), если x>|b|,  (ax-|a|)2, если x≤|a|;  b=-1.5; a=0.6; | 22) (ab-cx)/abcx2, если x<0,  y(x)=, если 0≤x≤0.8,  x2+ab, если x>0.8;  a=-5/3; b=2.3; c=-3.335; | |
| 23) cos(x3-ab-a1/2), если a<x≤b,  y(x)= (x-a)3+b2, если x≤a,  (c+d)2+cx, если x>b;  a=0.24; b=2; c=8/3; d=0.5; | 24) 9.25-|ac+dx|, если a≤x≤d,  y(x)= 4(d2/a+c2x/b2), если x>d,  ad2cx, если x<a;  a=-2.3; b=5.61; c=4; d=2.5; | |
| 25) cos(ax+z), если a≤x<z,  y(x)= x+z+4.28, если x<a,  9(x+za), если x≥z;  a=-0.025; z=0.25; |  | |

**Задание 3: Вычислить таблицы значений функции, заданной аналитически, при ИЗВЕСТНЫХ начальном И конечном значениях АРГГУМЕНТА и шаге его изменения**

###### (циклический алгоритм)

**Цель работы**: составить программу для табулирования функции y(x);

вывести на экран значения x и y(x).

**Операторы организации циклов**

Для табулирования функции используются **операторы организации циклов**:

**1) while** – оператор цикла с предварительным условием;

**2) repeat** – оператор цикла с последующим условием.

Общий вид оператора **while**: **while** логическое выражение **do** оператор; Оператор действует следующим образом. Предварительно проверяется значение логического выражения. Пока логическое выражение истинно, выполняется оператор циклической части (после **do**). Как только оно становится ложным, происходит выход из цикла.

Общий вид оператора **repeat**: **repeat**

операторы

циклической

части

**until** логическое выражение;

Оператор действует так. Выполняются операторы циклической части, проверяется значение логического выражения: если логическое выражение ложно, то вновь выполняются операторы циклической части; если же оно истинно, то цикл заканчивается.

Пример. Вычислим и выведем на экран таблицу значений функции y=lnx при x, изменяющемся от x0 до xk с шагом dx.

Организация цикла с оператором **while**:

x:=x0;

**while** x<xk+dx/2 **do**

begin

y:=ln(x); *{операторы*

writeln(‘ x=’,x,’ y(x)=’,y); *циклической*

x:=x+dx; *части}*

end;

Организация цикла с оператором **repeat**:

x:=x0;**repeat**

y:=ln(x); *{операторы*

writeln(' x=',x,' y(x)=',y); *циклической*

x:=x+dx *части}*

**until** x>xk;

## Варианты заданий

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | | | |
|  | | | | | | | |  |
|  | | | | | |  | | |
|  | | | | | | | |  |
|  | | | | |  | | | |
|  | | | | | | | |  |
|  | | |  | | | | | |
|  | | | | | | |  | |
|  | | | | | |  | | |
|  | | | | | | |  | |
|  | | | | | |  | | |
|  | | | |  | | | | |
|  | | | |  | | | | |
|  | | | |  | | | | |
|  | | |  | | | | | |
|  | | | |  | | | | |
|  | | | |  | | | | |
|  | | | |  | | | | |
|  | | | | | |  | | |
|  | | | |  | | | | |
|  | |  | | | | | | |
|  |  | | | | | | | |
|  |  | | | | | | | |
|  |  | | | | | | | |
|  |  | | | | | | | |
|  |  | | | | | | | |

КОНТРОЛЬНАЯ РАБОТА №2

**Задание 1**

**ИНТЕРПОЛИРОВАНИЕ ФУНКЦИИ**

**Цель работы**: составить программу для интерполирования функции y(x), заданной таблицей значений, используя интерполяционный полином Лагранжа.

## Постановка задачи и ее решение

**Известна** функция y(x), заданная таблицей значений:

|  |  |
| --- | --- |
| x | y(x) |
| x0  x1  x2  …  xn | y0  y1  y2  …  yn |

**Требуется**, используя значения функции yi  , i=0, 1, 2, …, n в узлах интерполяции xi, вычислить значение y(x) для любого x из промежутка [x0 ; xn].

Для решения задачи строится интерполяционный полином Лагранжа:



имеющий в заданных узлах x0, x1, x2, …, xn те же значения, что и функция y(x),

т.е. такой, что

 i=0, 1, 2, …, n;

x – значение аргумента, для которого требуется вычислить y(x);

значение полинома Ln(x) будет считаться искомым значением y(x).

Таким образом, решение задачи интерполирования сводится к программированию алгоритма вычисления значения полинома Ln(x).

Пример. Пусть функция y(x) задана таблицей значений

|  |  |
| --- | --- |
| x | y(x) |
| 0,1 | 0,9950 |
| 0,3 | 0,9553 |
| 0,5 | 0,8776 |
| 0,7 | 0,7648 |
| 0,9 | 0,6216 |
| 1,1 | 0,4536 |
| 1,3 | 0,2675 |
| 1,5 | 0,0707 |
| 1,7 | -0,1288 |
| 1,9 | -0,3233 |

Требуется вычислить значение функции y(x) для x=0,2 и x=1,8.

Составим программу:

**PROGRAM** Lagrange;**USES** crt;

**CONST** n=10;**VAR** x,y:array[1..n] of real ;

x0,p,s:real;

i,j:integer;

**BEGIN**

clrscr;

x[1]:=0.1; y[1]:=0.995;

x[2]:=0.3; y[2]:=0.9553;

x[3]:=0.5; y[3]:=0.8776;

x[4]:=0.7; y[4]:=0.7648;

x[5]:=0.9; y[5]:=0.6216;

x[6]:=1.1; y[6]:=0.4536;

x[7]:=1.3; y[7]:=0.2675;

x[8]:=1.5; y[8]:=0.0707;

x[9]:=1.7; y[9]:=-0.1288;

x[10]:=1.9; y[10]:=-0.3233;

writeln('введите значение x');

read(x0);

s:=0;

for i:=1 to n do begin p:=1; for j:=1 to n do if j<>i then

p:=p\*(x0-x[j])/(x[i]-x[j]);

s:=s+y[i]\*p;

end;

writeln('искомое значение функции: y(x) =',s:7:4)

**END**.

В данной программе: x0 – переменная вещественного типа, в которую записывается значение аргумента x; значение переменной s –значение полинома Ln(x) – искомое значение функции y(x).

Результат работы программы получим в виде:

**искомое значение функции: y(x) = 0.9799** (для x=0,2)

**искомое значение функции: y(x) =-0.2270** (для x=1,8)

Листинг программы с результатами расчетов представлен в приложении (см рис.П.3, рис.П.10, рис.П.14).

## Варианты заданий

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1) x | y(x) | 2) x | y(x) | | 3) x | | y(x) |
| 2,0  2,2  2,4  2,6  2,8  3,0  3,2  3,4  3,6  3,8 | 0,9093  0,8085  0,6755  0,5155  0,3350  0,1411  -0,0584  -0,2555  -0,4425  -0,6119 | 6,3  6,5  6,7  6,9  7,1  7,3  7,5  7,7  7,9  8,1 | 0,0168  0,2151  0,4048  0,5784  0,7290  0,8504  0,9380  0,9882  0,9989  0,9699 | | 0,1  0,3  0,5  0,7  0,9  1,1  1,3  1,5  1,7  1,9 | | 0,9093  0,8085  0,6755  0,5155  0,3350  0,1411  -0,0584  -0,2555  -0,4425  -0,6119 |
| x=2,1 x=3,7 | | x=6,4 x=7,6 | | | x=0,17 x=1,89 | | |
|  | |  | | |  | | |
| 4) x | y(x) | 5) x | y(x) | | 6) x | y(x) | |
| 2,0  2,2  2,4  2,6  2,8  3,0  3,2  3,4  3,6  3,8 | -0,4161  -0,5885  -0,7374  -0,8596  -0,9422  -0,9900  -0,9668  -0,8968  -0,7910  -0,6709 | 0,72  0,92  1,12  1,32  1,52  1,72  1,92  2.12  2,32  2,52 | 0,4868  0,3985  0,3269  0,2671  0,2187  0,1791  0,1446  0,1200  0,0983  0,0805 | | 0,45  0,50  0,55  0,60  0,65  0,70  0,75  0,80  0,85  0,90 | 0,4831  0,5463  0,6131  0,6841  0,7602  0,8423  0,9316  1,0296  1,1383  1,2602 | |
| x=2,1 x=3,7 | | x=0,75 x=2,41 | | | x=0,48 x=0,87 | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
| 7) x | y(x) | 8) x | y(x) | | 9) x | y(x) | |
| 0,49  0,54  0,59  0,64  0,69  0,74  0,79  0,84  0,89  0,94 | 0,5334  0,5994  0,6696  0,7445  0,8253  0,9131  1,0692  1,1156  1,2346  1,3692 | 0,47  0.52  0.57  0,62  0,67  0,72  0,77  0,82  0,87  0,92 | 0,5080  0,5726  0,6410  0,7139  0,7922  0,8770  0,9696  1,0717  1,1853  1,3133 | | 0,50  0,55  0,60  0,65  0,70  0,75  0,80  0,85  0,90  0,95 | 0,5463  0,6131  0,6841  0,7602  0,8423  0,9316  1,0296  1,1383  1,2602  1,3984 | |
| x=0,5 x=0,93 | | x=0,48 x=0,9 | | | x=0,49 x=0,92 | | |
|  | |  | | |  | | |
| 10) x | y(x) | 11) x | y(x) | | 12) x | | y(x) |
| 0,48  0,53  0,58  0,63  0,68  0,73  0,78  0,83  0,88  0,93 | 0,5206  0,5859  0,6552  0,7291  0,8087  0,8949  0,9893  1,0934  1,2097  1,3409 | 0,53  0,58  0,63  0,68  0,73  0,78  0,83  0,88  0,93  0,98 | 0,5859  0,6552  0,7291  0,8087  0,8949  0,9893  1,0934  1,2097  1,3409  1,4909 | | 0,71  0,81  0,91  1,01  1,11  1,21  1,31  1,41  1,51  1,61 | | 2,0340  2,2479  2,4843  2,7456  3,0344  3,3535  3,7062  4,0960  4,5267  5,0028 |
| x=0,5 x=0,9 | | x=0,56 x=0,95 | | | x=0,75 x=1,6 | | |
|  | |  | | |  | | |
| 13) x | y(x) | 14) x | y(x) | | 15) x | | y(x) |
| 0,75  0,85  0,95  1,05  1,15  1,25  1,35  1,45  1,55  1,65 | 2,1170  2,3396  2,5857  2,8577  3,1582  3,4903  3,8574  4,2631  4,7115  5,2070 | 0,73  0,83  0,93  1,03  1,13  1,23  1,33  1,43  1,53  1,63 | 2,0751  2,2933  2,5345  2,8011  3,0957  3,4212  3,7810  4,1787  4,6182  5,1039 | | 0,7  0,9  1,1  1,3  1,5  1,7  1,9  2,1  2,3  2,5 | | 0,4966  0,4066  0,3329  0,2725  0,2231  0,1827  0,1496  0,1224  0,1002  0,0821 |
| x=0,8 x=1,62 | | x=0,77 x=1,59 | | | x=0,8 x=2,45 | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  |  |  |  | |  |  | |
| 16) x | y(x) | 17) x | y(x) | | 18) x | y(x) | |
| 0,74  0,84  0,94  1,04  1,14  1,24  1,34  1,44  1,54  1,64 | 2,0959  2,3164  2,5600  2,8292  3,1268  3,4556  3,8190  4,2207  4,6646  5,1552 | 0,55  0,65  0,75  0,85  0,95  1,05  1,15  1,25  1,35  1,45 | 0,5782  0,6968  0,8223  0,9561  1,0995  1,2539  1,4208  1,6019  1,7991  2,0143 | | 1,7  1,8  1,9  2,0  2,1  2,2  2,3  2,4  2,5  2,6 | 2,8283  3,1075  3,4177  3,7622  4,1443  4,5679  5,0372  5,5569  6,1323  6,7690 | |
| x=0,79 x=1,55 | | x=0,57 x=1,31 | | | x=1,81 x=2,56 | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
|  | |  | | |  | | |
| 19) x | y(x) | 20) x | | y(x) | 21) x | y(x) | |
| 1,73  1,83  1,93  2,03  2,13  2,23  2,33  2,43  2,53  2,63 | 2,9090  3,1979  3,5173  3,8727  4,2669  4,7037  5,1876  5,7235  6,3166  6,9729 | 1,71  1,81  1,91  2,01  2,11  2,21  2,31  2,41  2,51  2,61 | | 2,8549  3,1371  3,4506  3,7987  4,1847  4,6127  5,0868  5,6119  6,1931  6,8363 | 1,74  1,84  1,94  2,04  2,14  2,24  2,34  2,44  2,54  2,64 | 2,9364  3,2277  3,5512  3,9103  4,3085  4,7499  5,2388  5,7801  6,3793  7,0423 | |
| x=1,76 x=2,5 | | x=1,74 x=2,6 | | | x=1,77 x=2,55 | | |
|  |  |  | |  |  | |  |
|  |  |  | |  |  | |  |
|  |  |  | |  |  | |  |
| 22) x | y(x) | 23) x | | y(x) | 24) x | | y(x) |
| 1,72  1,82  1,92  2,02  2,12  2,22  2,32  2,42  2,52  2,62 | 2,8818  3,1669  3,4838  3,8355  4,2256  4,6580  5,1370  5,6674  6,2545  6,9043 | 1,75  1,85  1,95  2,05  2,15  2,25  2,35  2,45  2,55  2,65 | | 2,9642  3,2585  3,5855  3,9483  4,3507  4,7966  5,2905  5,8373  6,4426  7,1123 | 1,35  1,45  1,55  1,65  1,75  1,85  1,95  2,05  2,15  2,25 | | 2,4604  3,0486  3,7239  4,4921  5,3594  6,3316  7,4149  8,6151  9,9384  11,3906 |
| x=1,73 x=2,6 | | x=1,77 x=2,53 | | | x=1,4 x=2,23 | | |
| 25) x | y(x) |
| 0,1  0,3  0,5  0,7  0,9  1,1  1,3  1,5  1,7  1,9 | 0,0998  0,2955  0,4794  0,6442  0,7843  0,8912  0,9636  0,9975  0,9917  0,9463 |
| x=0,2 x=1,55 | |

**Задание 2**

**ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ**

**Цель работы**: составить программу для вычисления определенного интеграла методом Симпсона.

## Постановка задачи и ее решение

Для решения задачи воспользуемся геометрическим истолкованием определенного интеграла: интеграл



численно равен площади криволинейной трапеции, ограниченной кривой y(x), прямыми x=a, x=b и отрезком [a; b] оси Ox (рис.1):



Рис. 1

*Формула Симпсона*



с геометрической точки зрения означает, что график функции y(x) заменен другой кривой ϕ(x), состоящей из дуг парабол: каждая сдвоенная дуга кривой y(x) заменяется параболой (рис.2):



Рис. 2

На рис.2: отрезок [a; b] разделен на четное число 2n, n=1,2, … равных отрезков точками x1, x2, …, x2n-1;

y0, y1, y2, …, y2n значения функции y(x) в точках x0, x1, x2, …, x2n;

точки x1, x3, …, x2n-1 - середины сдвоенных отрезков [x0, x2], [x2, x4], …,

[x2n-2, x2n];

ϕ(x) - кривая, составленная из дуг парабол.

За приближенное значение интеграла I принимается площадь криволинейной трапеции, ограниченной кривой ϕ(x), прямыми x=x0, x=x2n  и отрезком [x0, x2n].

Таким образом, решение задачи о приближенном вычислении определенного интеграла сводится к программированию алгоритма вычисления площади криволинейной трапеции по *формуле Симпсона.*

Пример. Вычислить по формуле Симпсона интеграл .

Составим программу:

**program** Simpson;

**USES** crt;

**VAR** x,a,b,h,s:real;

n:integer;

**FUNCTION** Y(p:real):real;

begin

Y:=1/(1+p\*p);

end;

**BEGIN**

clrscr;

write(' Отрезок интегрирования [a,b] ? ');

read(a,b);

write(' На сколько частей разбиваем отрезок интегрирования? n=');

read(n);

h:=(b-a)/n;

s:=0; x:=a+h;

while x<b do

begin

s:=s+4\*Y(x);

x:=x+h;

s:=s+2\*Y(x);

x:=x+h;

end;

s:=h/3\*(s+Y(a)-Y(b));

writeln;

writeln(' Интеграл равен I=',s);

**END**.

В программе: подынтегральная функция y(x) оформлена как функция Turbo Pascal (в разделе **FUNCTION )**; n - число частичных отрезков (**четное**); h - длина каждого частичного отрезка; в переменную s записывается приближенное значение интеграла, вычисленное по *формуле Симпсона*.

Листинг программы с результатами расчетов представлен в приложении (см рис.П.15, рис.П.16).

## Варианты заданий

|  |
| --- |
| 1)  2)  3)  4)  5)  6)  7)  8)  9)  10)  11)  12)  13)  14)  15)  16)  17)  18)  19) 20)  21) 22)  23)  24)  25)  26)  27)  28)  29)  30) |