1 курс

ПЛАН – КОНСПЕКТ
проведения лекционного занятия, практических занятий № 27-28
по дисциплине «Информатика»

Раздел 4. «Основы алгоритмизации и программирования.»

Тема 4.3.: «Графический режим.»

часть 2

Подготовил: преподаватель
В.Н. Борисов

Рязань 2023

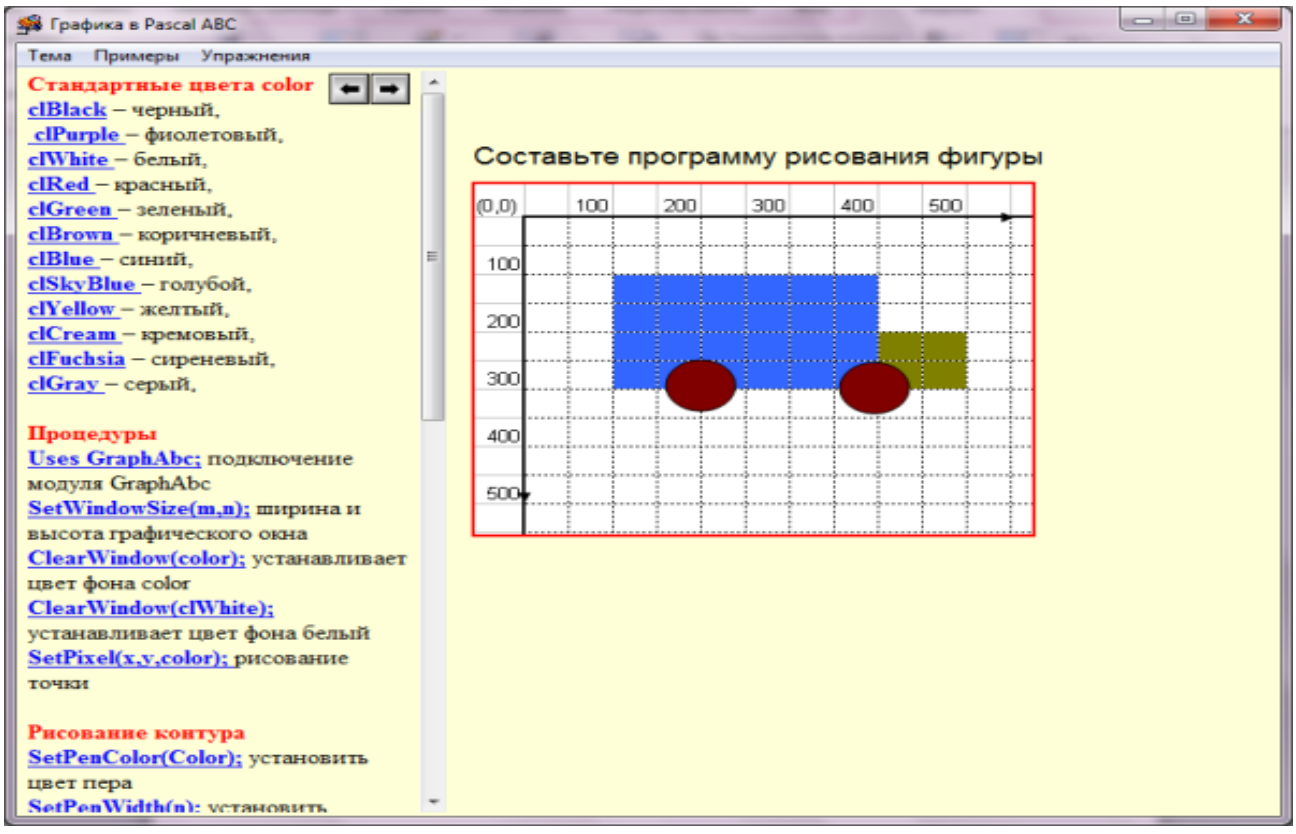
Вопросы занятий:

1. Операторы цикла в графическом режиме.
2. Результат исполнения циклической программы в графическом режиме.
3. Генератор случайных чисел в графическом режиме.
4. Построение графика функции.
5. Компьютерное моделирование геометрических и физических задач в графическом режиме.
6. Построение графика функции (практическое занятие № 27, теоретическая часть, выполнение практического задания).
7. Составление программ для графической интерпретации и исследования физических моделей (практическое занятие № 28, теоретическая часть, выполнение практического задания).

Время проведения лекционного, практических занятий – 6 часов.

Графика в Pascal ABC
Модуль GraphABC

SetPixel(x,y; c); - цветная точка (c – цвет)
Line(x1,y1,x2,y2); - линия
MoveTo(x,y); - назначить нач. точку
LineTo(x,y); - линия от пред. точки
DrawCircle(x,y,r); - окружность (r – радиус)
FillCircle(x,y,r);
FillEllipse(x1,y1,x2,y2); - овал
DrawRectangle(x1,y1,x2,y2); - прямоугольник
FillRect(x1,y1,x2,y2); - заполн. прямоугольник
TextOut(x,y; s); - вывод текста (s – строка)



Модуль Graph

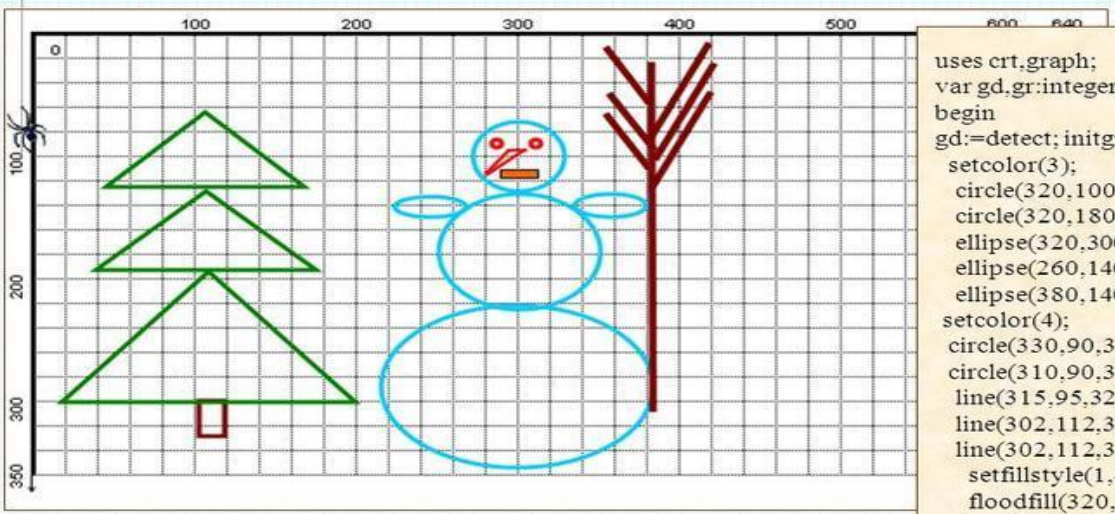
Модуль *Graph* Турбо Паскаля содержит около пятидесяти различных процедур и функций, предназначенных для работы с графическим экраном. В этом же модуле описаны некоторые встроенные константы и переменные, которые могут быть использованы в графических программах:

- Основную часть модуля составляют процедуры вывода базовых графических элементов, таких как точки, отрезки прямых линий, дуги и целые окружности и т.д. Такие элементы называются *графическими примитивами*.
- Другая группа процедур предназначена для управления графическим режимом.
- закрашивание областей заданным цветом и шаблоном;
- вывод текста различным шрифтом, заданного размера и направления;
- определение окон и отсечение по их границе;

18

Пишем программу № 23.

Вывести на экран рисунок




```
uses crt,graph;
var gd,gr:integer;
begin
gd:=detect; initgraph(gd,gr,'');
  setcolor(3);
   circle(320,100,30);
   circle(320,180,50);
   ellipse(320,300,0,360,100,70);
   ellipse(260,140,0,360,30,10);
   ellipse(380,140,0,360,30,10);
 setcolor(4);
  circle(330,90,3);
  circle(310,90,3);
   line(315,95,325,95);
   line(302,112,315,95);
   line(302,112,325,95);
     setfillstyle(1,4);
     floodfill(320,97,4);
    bar(330,110,310,113);
readkey;
end.
```

Задание:

A. Составить программу вывода на экран снеговика (программа прилагается)

B. Составить программу вывода на экран всего предлагаемого рисунка

C. Доработать программу вывода на экран рисунка, добавив дополнительные детали (заливку ёлочки, снеговика, вставить изображение шляпы у снеговика, дерева и т.п.)

Выход

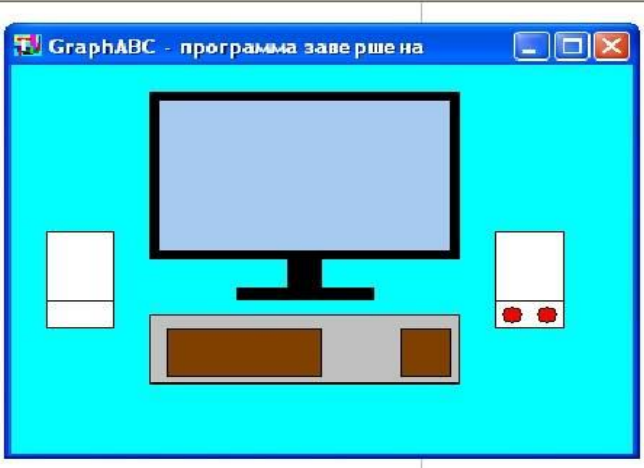
Pascal ABC

Файл Правка Вид Программа Сервис Помощь

Komp.pas

```
uses GraphABC;
begin
clearwindow(claqua);
setwindowsize(360,280);
Rectangle(80,20,260,140); //monitor
floodfill(82,22,clblack);
//setbrushcolor(claqua);
Rectangle(85,25,255,135);  //monitor
floodfill(90,30,clskyblue);
Rectangle(160,140,180,160);  //nojka monitora
floodfill(165,145,clblack);
Rectangle(130,160,210,170);    //lapka monitora
floodfill(132,162,clblack);
Rectangle(80,180,260,230);   //klava
floodfill(82,182,clsilver);
Rectangle(225,190,255,225);  //num klavy
floodfill(230,195,clbrown);
Rectangle(90,190,180,225);   //bukvy klavy
floodfill(95,195,clbrown);
Rectangle(20,120,60,190);   //   kolonka levaya
Rectangle(280,120,320,190); // kolonka pravaya
line(20,170,60,170);   // kolonka
line(280,170,320,170); //kolonka
circle(290,180,6); //kolonka
floodfill(290,180,clred);
circle(310,180,6); //kolonka
floodfill(310,180,clred);

end.
```

GraphABC - программа завершена



Строка: 14 Столбец: 49

Pascal ABC

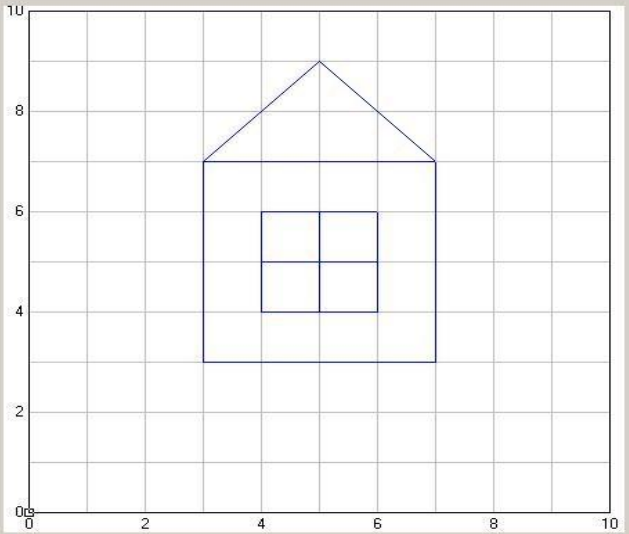
Файл Правка Вид Программа Сервис Помощь

Домашнее задание Домик.pas

```
program Domik;
 uses Drawman;
begin
 Field (10,10);
 //квадрат в центре поля
 ToPoint(3,3); PenDown;
 ToPoint(3,7);
 ToPoint(3,7);
 ToPoint(7,7);
 ToPoint(7,3);
 ToPoint(3,3); PenUp;
 //крыша
 ToPoint(3,7); PenDown;
 ToPoint(5,9);
 ToPoint(7,7); PenUp;
  //окно
 ToPoint(4,4); PenDown;
 ToPoint(4,6);
 ToPoint(6,6);
 ToPoint(6,4);
 ToPoint(4,4); PenUp;
 //рама
 ToPoint(4,5); PenDown;
 ToPoint(6,5); PenUp;
 ToPoint(5,4); PenDown;
 ToPoint(5,6); PenUp;
 // домой
 ToPoint(0,0);
end.
```

Исполнитель Чертежник

Поле размера 10 x 10



Пуск (Enter) Шаг (Space) Заново (BS) Выход (Esc) Справка (F1)

Скорость:

Чертежник: Работа окончена, задание не выполнено

Состояние: Шаг: 29

Строка: 1 Столбец: 1

Пуск PABC IT2: Анализ и сам... Реферат Цыбико... Referat_Guseva_... Pascal ABC RU 0:04

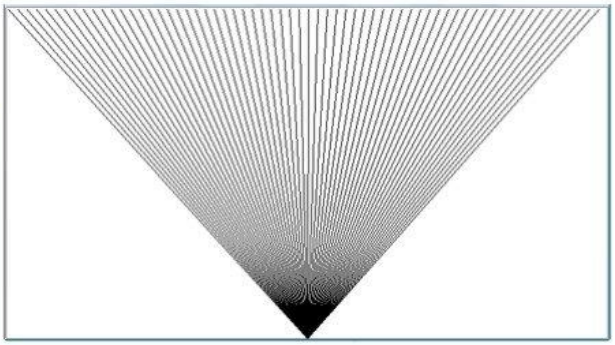
Первый вопрос: Операторы цикла в графическом режиме.

Второй вопрос: Результат исполнения циклической программы в графическом режиме.

Графические задачи на циклы. 9

Задача 1.

Составить программу выводящую на экран следующее изображение



```
Program n1;
Uses Crt, GraphABC;
var x,i:Integer;
begin
  x:=1;
  For i:=1 to 64
              lin
              x:=
              end.
```

Более короткое решение:

```
Program n1;
Uses Crt, GraphABC;
var x:Integer;
begin
  For x:=1 to 64 do line(320,400,x*10,1);
end.
```

Третий вопрос: Генератор случайных чисел в графическом режиме.

Датчик случайных чисел

`random` - генерирует случайное вещественное число в диапазоне [0,1).

`random(x)` - генерирует случайное целое число в диапазоне [0,x).

`x + random *(y-x)` - генерирует случайное вещественное число в диапазоне [x,y).

`x + random(y-x)` - генерирует случайное целое число в диапазоне [x,y).

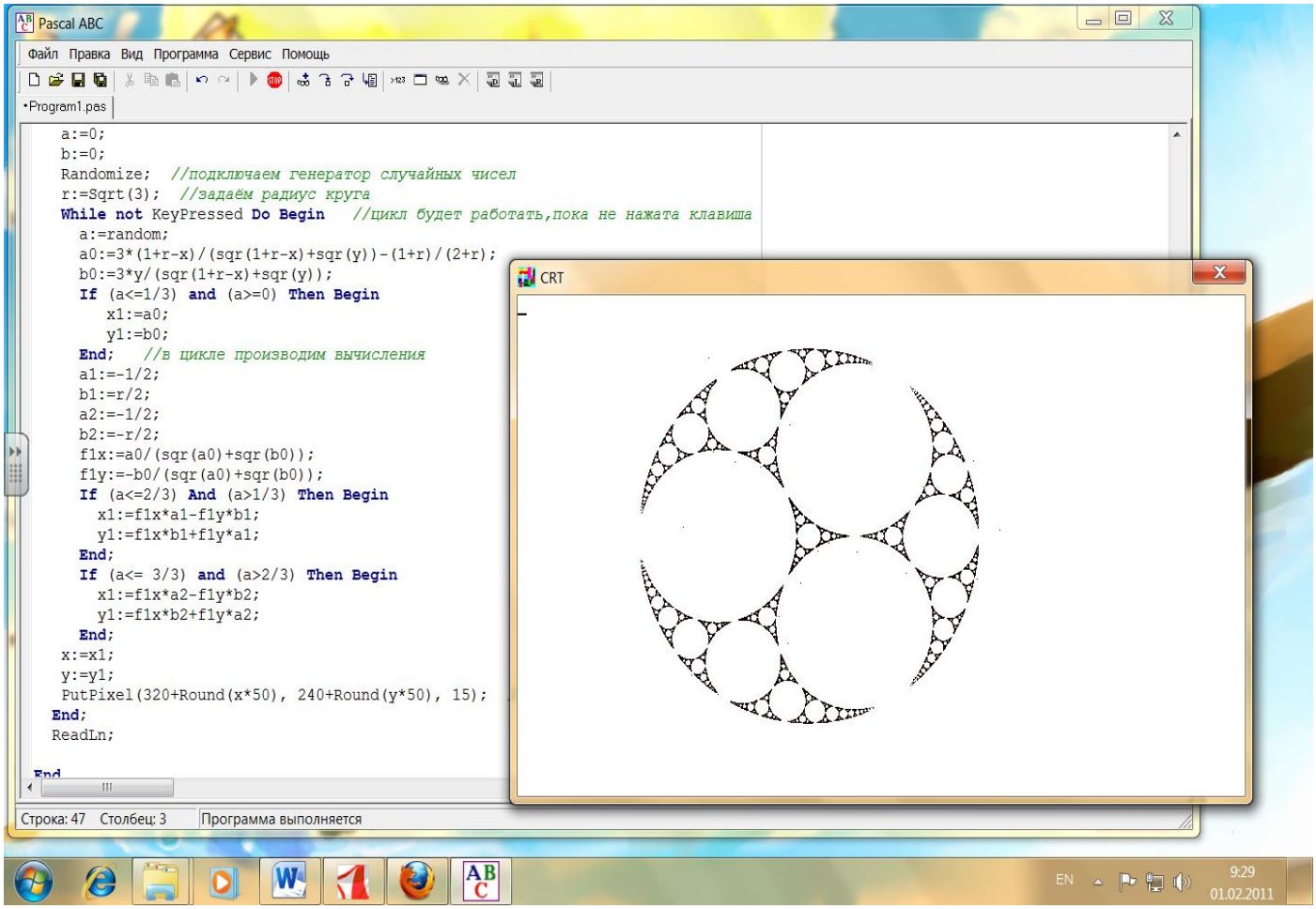
Для повышения «степени случайности» существует процедура `randomize`, которая меняет базу генерации, ее используют до функции `random`.

```
randomize;
random(n);
```

Выдаст число в диапазоне 0...n-1

```
random(5);
```

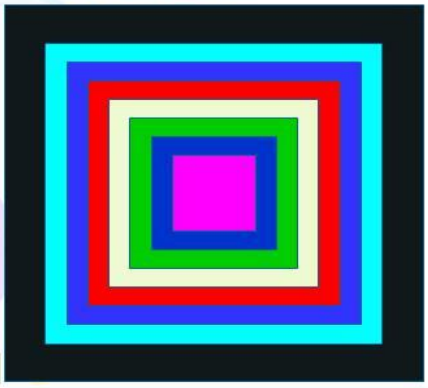
Выдаст число в диапазоне 0...4

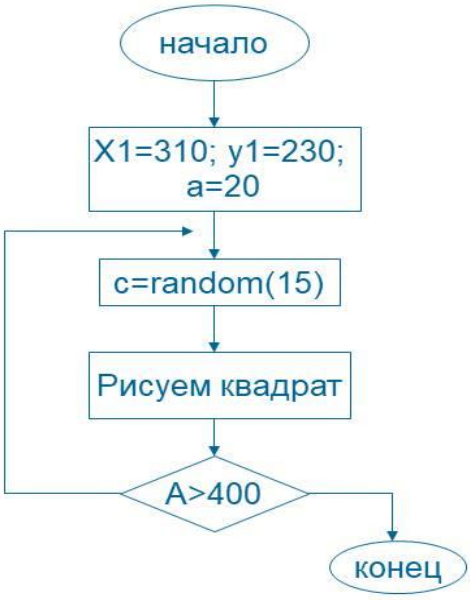


Примеры решения задач:

Забавные квадраты

- Нарисовать квадраты, стороны которых увеличиваются от 20 до 400 ppi, с шагом 20 ppi. Центр экрана совпадает с центром квадрата. Цвет квадратов задаётся случайным числом. Для решения задачи используется цикл с постусловием.





ГРАФИКА + ЦВЕТ 66

SetColor(Color: word);
Устанавливает цвет пера

GetColor: word;
Возвращает цвет пера

SetBkColor(color: word);
Устанавливает цвет фона.

GetBkColor: word;
Возвращает цвет фона.

Цвета	
Black	– чёрный
Blue	– синий
Green	– зелёный
Cyan	– циановый
Red	– красный
Magenta	– сиреневый
Brown	– коричневый
LightGray	– светло-серый
DarkGray	– тёмно-серый
LightBlue	– голубой
LightGreen	– светло-зелёный
LightCyan	– светло-циановый
LightRed	– розовый
LightMagenta	– светло-сиреневый
Yellow	– жёлтый
White	– белый

```
Program My_Program;
 Uses CRT,Graph;
 Var g,b:Integer;
Begin
 ClrScr;
 Randomize;
 g:=0; InitGraph(g,b,'d:\BP\BGI');
 SetBkColor (Red);
 For g:=0 to 10 do Begin
    Line (320, 240, Random(640), Random(480));
    SetColor (Random(15));
 End;
 ReadKey;
 CloseGraph;
End.
```

Четвертый вопрос: Построение графика функции.

Шестой вопрос: Построение графика функции (практическое занятие № 27, теоретическая часть, выполнение практического задания).

УРОК 6. Построение графиков функций на языке Паскаль

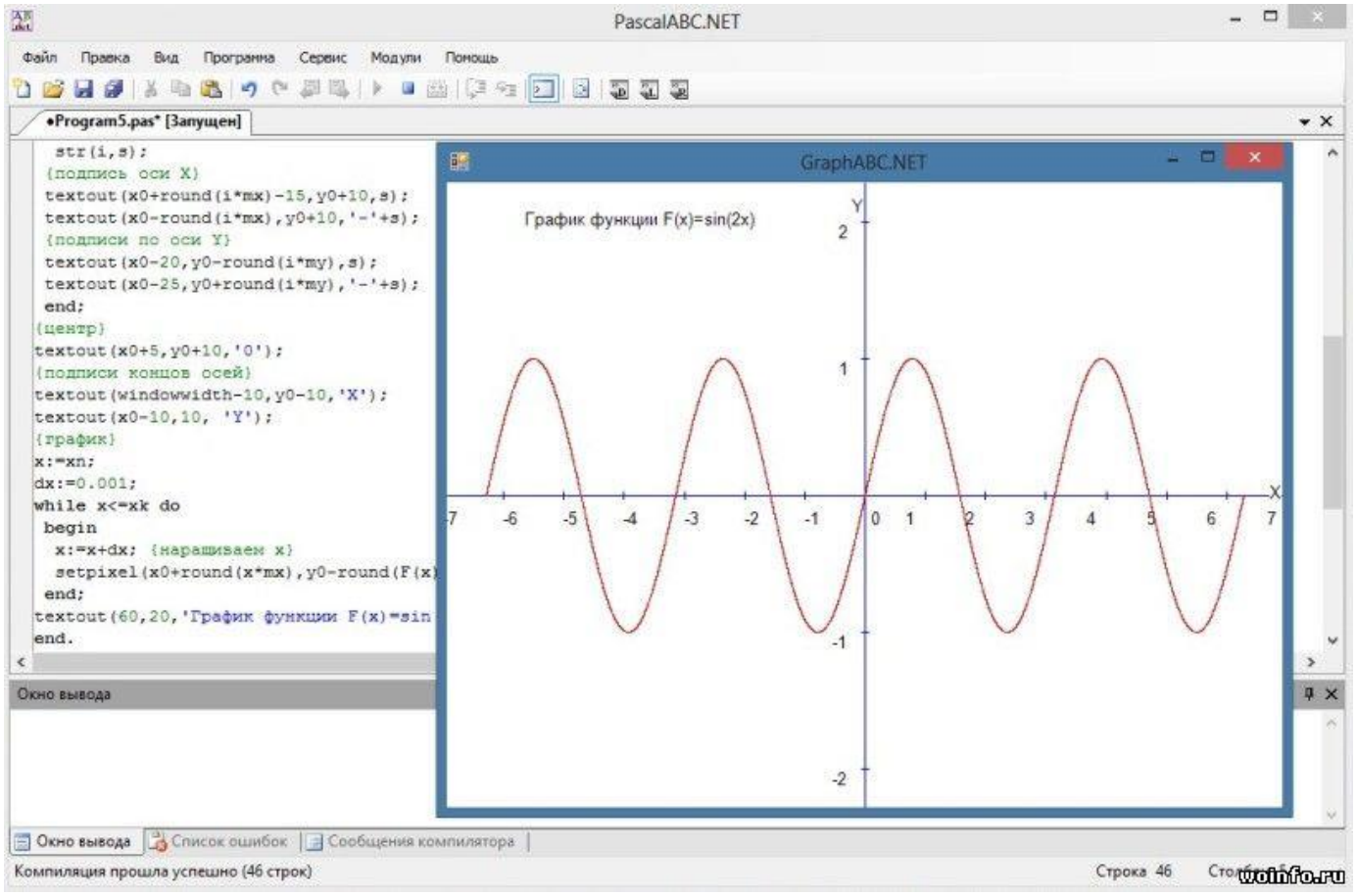
Вывод символьной информации в графическое окно.

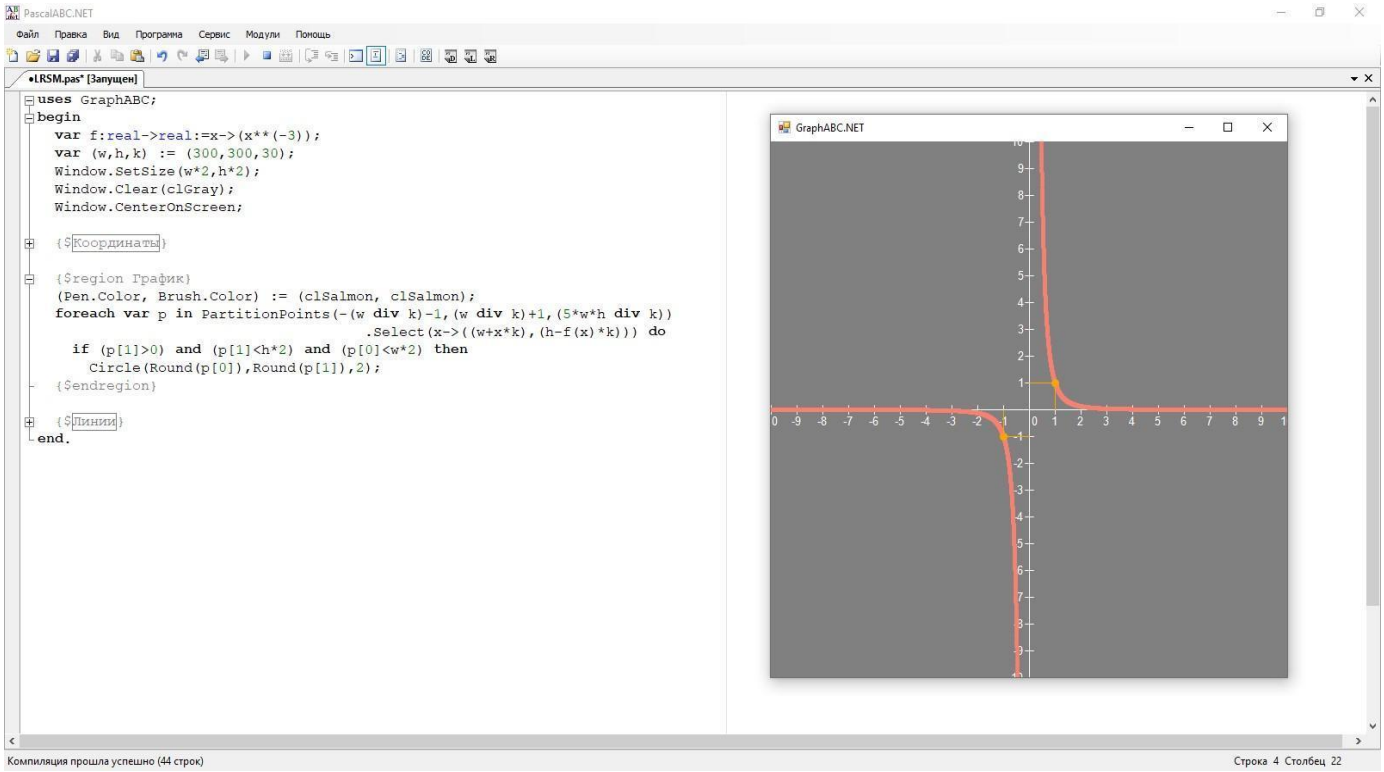
```
Program Grafik;
uses GraphABC;
var x,y,i: integer;
begin
line (500,0,500,500);
line (0,250,1000,250);
for x:=-100 to 200 do
begin
i:=i+1;
y:=x+2;
SetPixel(x+500,250-y,RGB (0,0,0));
if 250-y<0 then break;
end;
TextOut(5,45,'функция построена на отрезке от  '+IntToStr(x-i)+' до
'+IntToStr(x));
end.
```

Счетчик цикла, считает количество итераций (повторов)

Строка, выводящая текст на экран с позиции x=5,y=45

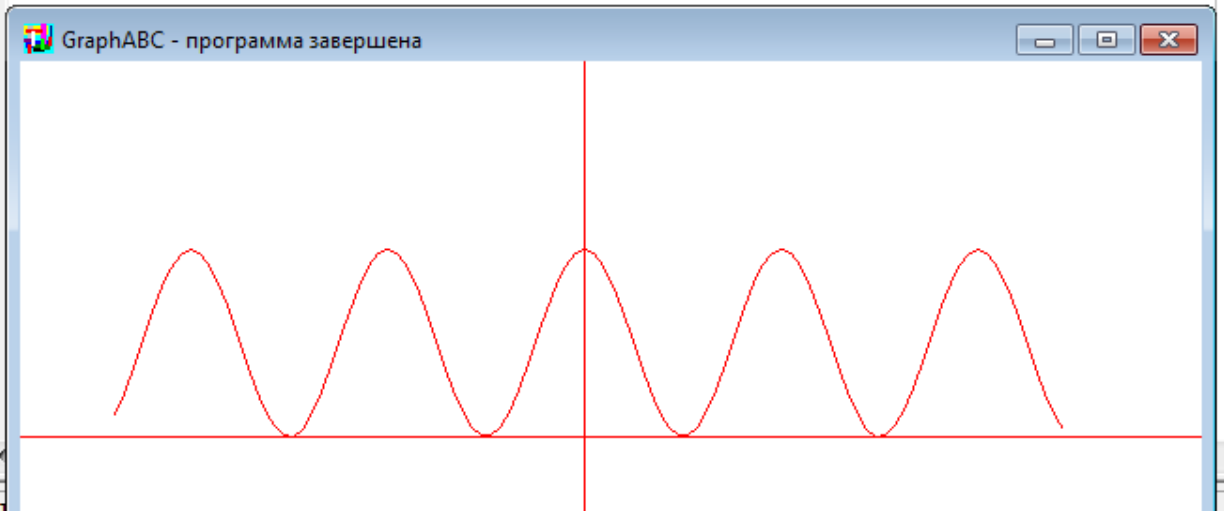
Оператор , превращающий численный тип данных в строковый


PascalABC.NET
Файл Правка Вид Программа Сервис Модули Помощь
•Program5.pas* [Запущен]
str(i,s);
{подпись оси X}
textout(x0+round(i*mx)-15,y0+10,s);
textout(x0-round(i*mx),y0+10,'-'+s);
{подписи по оси Y}
textout(x0-20,y0-round(i*my),s);
textout(x0-25,y0+round(i*my),'-'+s);
end;
{центр}
textout(x0+5,y0+10,'0');
{подписи концов осей}
textout(windowwidth-10,y0-10,'X');
textout(x0-10,10, 'Y');
{график}
x:=xn;
dx:=0.001;
while x<=xk do
begin
x:=x+dx; {наращиваем x}
setpixel(x0+round(x*mx),y0-round(F(x)
end;
textout(60,20,'График функции F(x)=sin
end.
GraphABC.NET
График функции F(x)=sin(2x)
Y
X
2
1
-1
-2
-7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7
Окно вывода
Окно вывода
Список ошибок
Сообщения компилятора
Компиляция прошла успешно (46 строк)
Строка 46
woinfo.ru


PascalABC.NET
Файл Правка Вид Программа Сервис Модули Помощь
•LRSM.pas* [Запущен]
uses GraphABC;
begin
var f:real->real:=x->(x**(-3));
var (w,h,k) := (300,300,30);
Window.SetSize(w*2,h*2);
Window.Clear(clGray);
Window.CenterOnScreen;
{$Координаты}
{$region График}
(Pen.Color, Brush.Color) := (clSalmon, clSalmon);
foreach var p in PartitionPoints(-(w div k)-1,(w div k)+1,(5*w*h div k))
.Select(x->((w+x*k),(h-f(x)*k))) do
if (p[1]>0) and (p[1]<h*2) and (p[0]<w*2) then
Circle(Round(p[0]),Round(p[1]),2);
{$endregion}
{$Линии}
end.
GraphABC.NET
Компиляция прошла успешно (44 строк)
Строка 4 Столбец 22

•Рисование графика функции.pas

```
uses GraphABC;
var x,x1,xn,dx,k: real;
function y(x:real):real;
begin
y:=cos(3*x)+1
end;
begin
SetPenColor(clred); //Цвет линий
x1:=-5;
xn:=5;
dx:=0.1;
k:=50;
Line(0,200,630,200);
Line(300,0,300,400);
MoveTo(round(k*x1+300),round(200-k*y(x1)));
x:=x1;
while x<=xn do begin
x:=x+dx;
LineTo(round(k*x+300),round(200-k*y(x)));
end;
end.
```

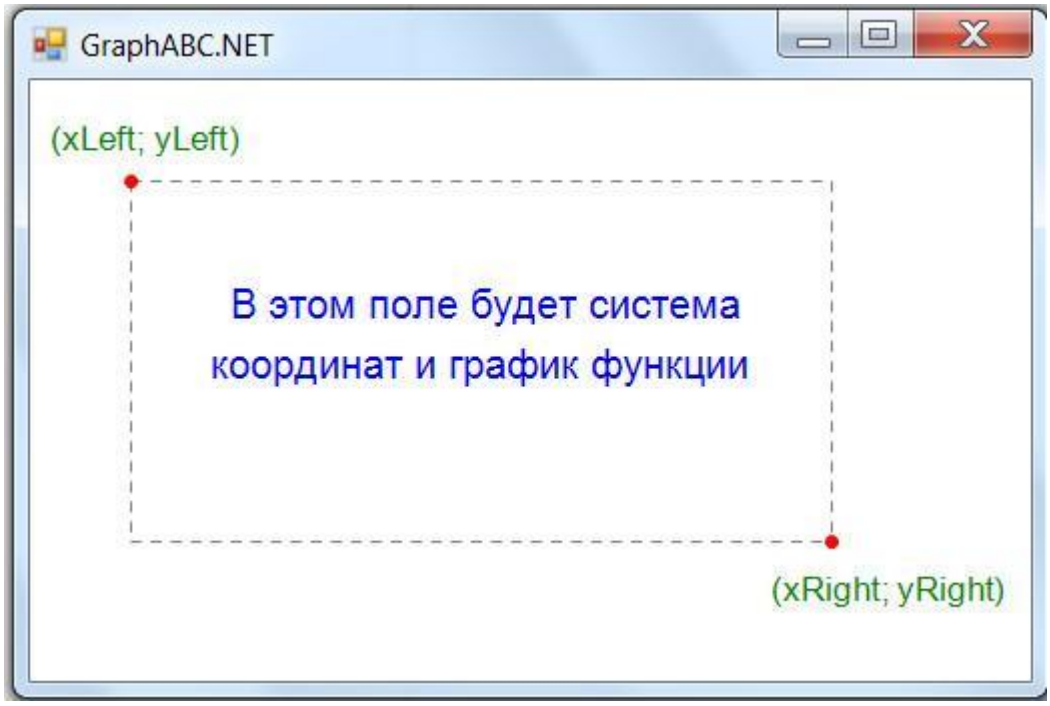


Графики функций в паскале

Построение графика функции происходит в два основных этапа: *построение системы координат* и, собственно, рисование самого графика. Кроме того, процесс создания системы координат тоже разбивается на несколько частей.

Систему будем строить с положительными и отрицательными значениями по обеим осям. Поскольку многие используют ещё турбо паскаль, то в конце страницы будет приведены две программы: одна – для PascalABC и PascalABC.Net, другая – для Turbo Pascal и Free Pascal.

Итак, для построения системы координат нам необходимо знать, в каких границах графического окна она будет находиться. Можно было бы опустить этот этап и строить декартову систему так, чтобы она занимала все графическое окно. Но это не очень удобно и просто некрасиво выглядит, когда график функции занимает всю область, не имея свободных полей слева-справа и сверху-снизу. Поэтому, чтобы задать прямоугольник, в котором будет находиться система координат, достаточно знать координаты левого верхнего и правого нижнего его углов.



Пусть (xLeft; yLeft) – координаты левого верхнего угла декартовой системы координат в графическом окне PascalABC.Net, (xRight; yRight) – соответственно координаты правого нижнего угла. Следующая задача – провести оси координат OX и OY. Будем считать, что нам нужны все четыре четверти координат. В этом случае обе оси будут иметь положительные и отрицательные значения. Чтобы правильно поставить центр координат (x_0; y_0), необходимо знать границы изменения аргумента x по оси OX и значения функции f по оси OY.

Итак, отложим по оси OX числа от a до b с интервалом dx, по оси OY – числа от fmin до fmax с разницей dy; причем обязательные условия: $a \le 0$, $b \ge 0$, $fmin \le 0$, $fmax \ge 0$. Для правильного отображения засечек на осях необходимо также, чтобы dx было делителем a и b, a dy было делителем fmin и fmax, и эти числа придется выбирать самостоятельно для каждого интервала. Но сначала нам придется познакомиться с таким понятием как *масштаб системы координат* в графическом окне паскаля. Что такое масштаб?

✎ **Масштаб** – это величина, или коэффициент, показывающий, сколько пикселей графического окна паскаля приходится на единицу оси системы координат. Например, по оси OX нужно расположить числа от -4 до 16 (всего 20 единиц), а ширина графического окна паскаля равна 1000 пикселей; тогда на единицу величины оси OX приходится 1000:20=50 пикселей/единицу. Это и есть масштаб по оси OX. Чтобы узнать, сколько пикселей содержат n единиц, надо просто умножить n на 50.

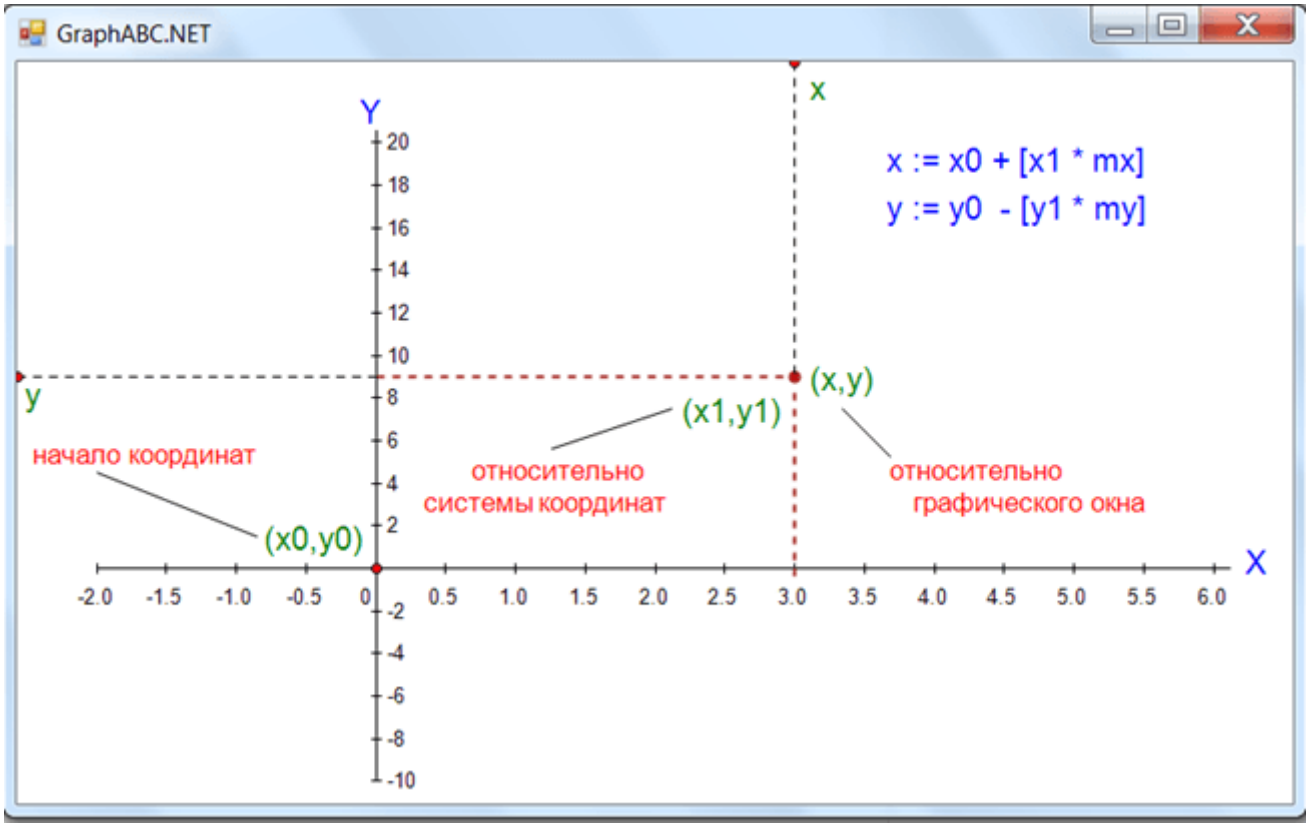
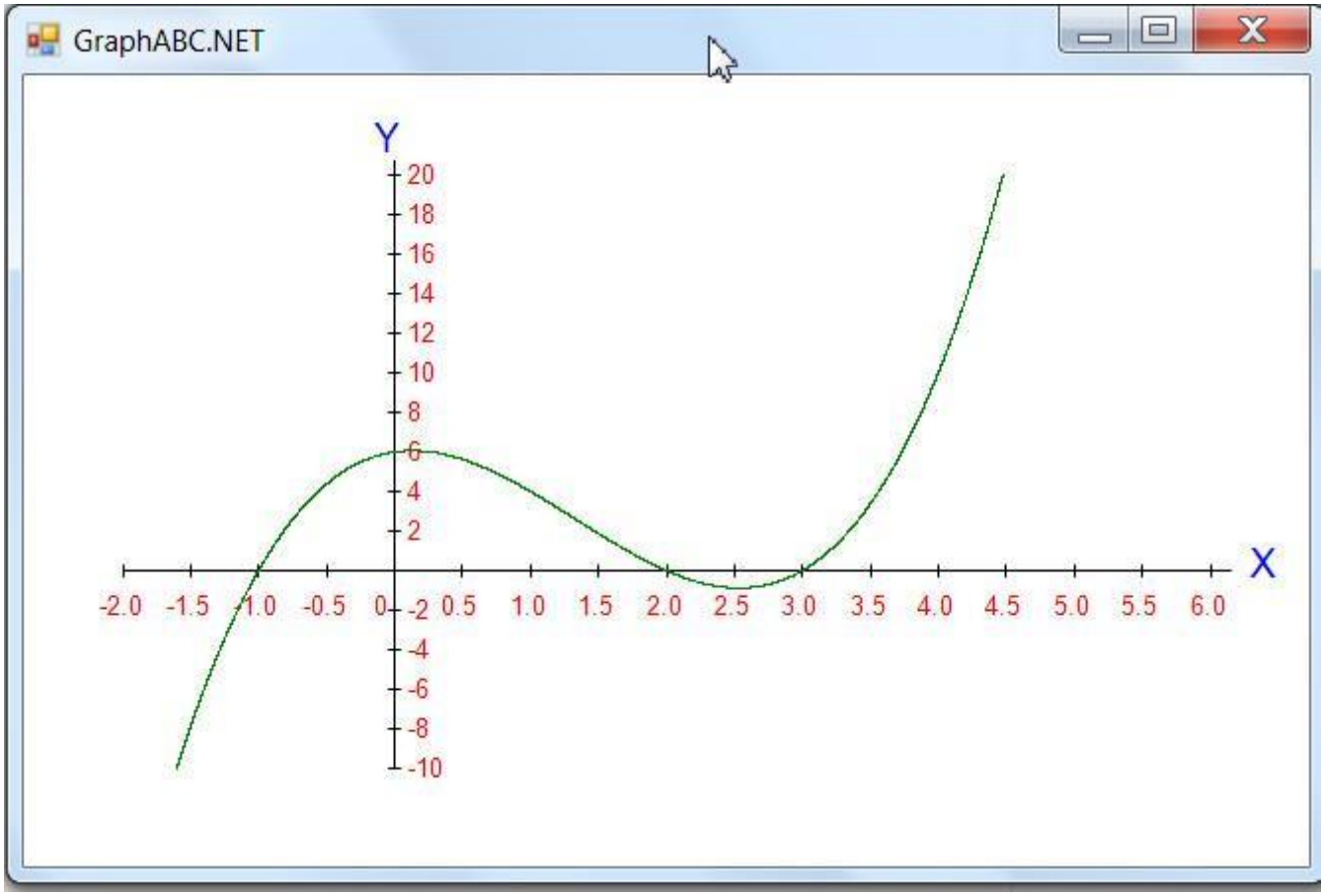


График функции будем строить по точкам, используя процедуру SetPixel(x, y, c), где x, y – координаты точки в графическом окне паскаля, c – цвет точки. Для рисования осей координат OX и OY воспользуемся процедурой Line(x1, y1, x2, y2), где (x_1; y_1) – координаты начальной точки, (x_2; y_2) – координаты конечной.

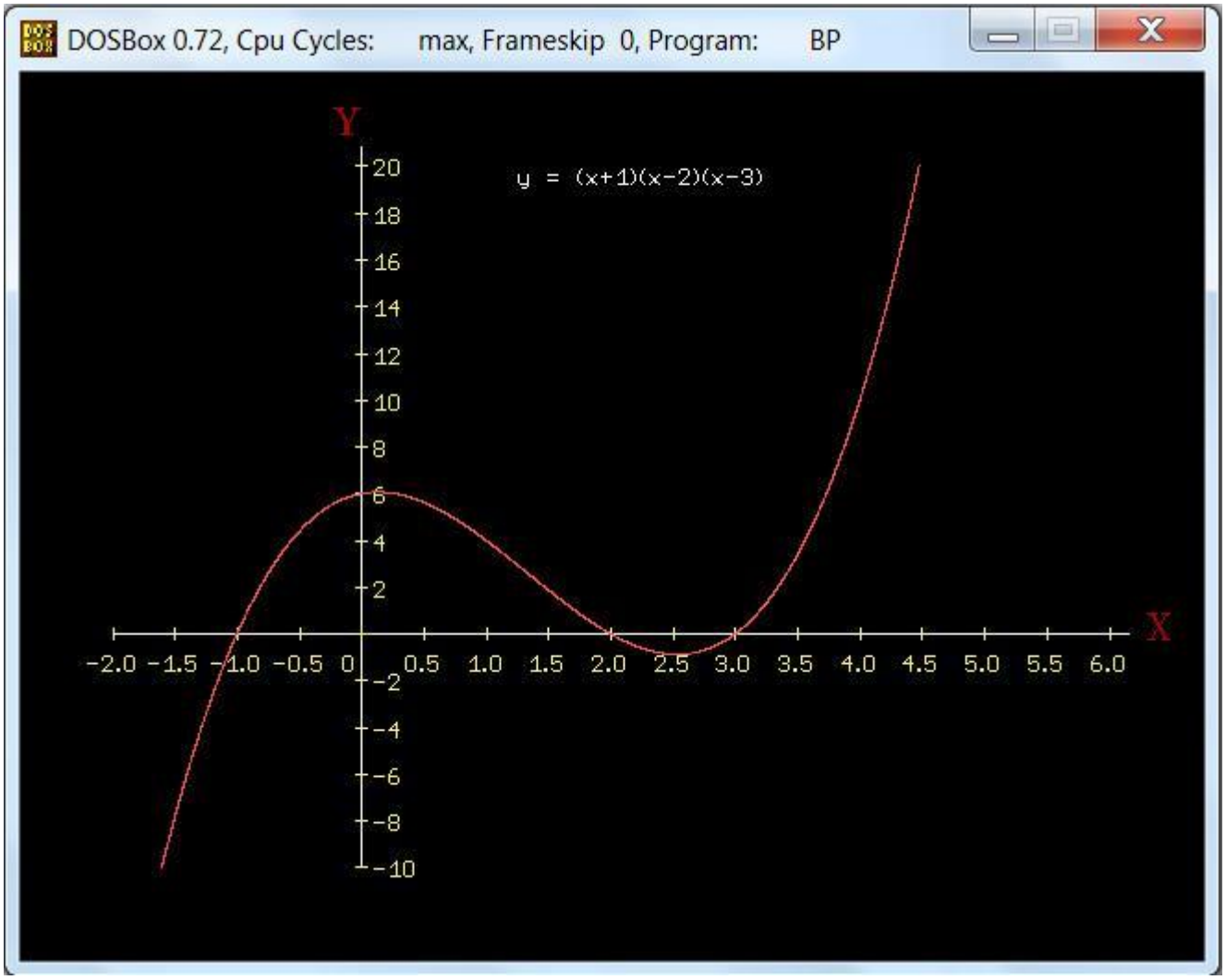
Последовательность такова: сначала строим систему координат, а после (в самом конце) вычисляем значения функции, вычисляем соответствующие координаты точки в графическом окне и ставим точку (x, y), закрашенную в зеленый цвет. Откройте **PascalABC** или **PascalABC.Net**, скопируйте следующий код и запустите программу:

```
uses
  graphABC; //Подключаем графический модуль

const
  W = 800; H = 500;//Размеры графического окна

function F(x: real): real;
begin
  F := (x + 1) * (x - 2) * (x - 3); //Функция
end;

var
  x0, y0, x, y, xLeft, yLeft, xRight, yRight, n: integer;
  a, b, fmin, fmax, x1, y1, mx, my, dx, dy, num: real;
  i: byte;
  s: string;

begin
  SetWindowSize(W, H); //Устанавливаем размеры графического окна
  //Координаты левой верхней границы системы координат:
  xLeft := 50;
  yLeft := 50;
```

```
  //Координаты правой нижней границы системы координат:
  xRight := W - 50;
  yRight := H - 50;
  //интервал по X; a и b должно нацело делится на dx:
  a := -2; b := 6; dx := 0.5;
  //Интервал по Y; fmin и fmax должно нацело делится на dy:
  fmin := -10; fmax := 20; dy := 2;
  //Устанавливаем масштаб:
  mx := (xRight - xLeft) / (b - a); //масштаб по X
  my := (yRight - yLeft) / (fmax - fmin); //масштаб по Y
  //начало координат:
  x0 := trunc(abs(a) * mx) + xLeft;
  y0 := yRight - trunc(abs(fmin) * my);
  //Рисуем оси координат:
  line(xLeft, y0, xRight + 10, y0); //ось OX
  line(x0, yLeft - 10, x0, yRight); //ось OY
  SetFontSize(12); //Размер шрифта
  SetFontColor(clBlue); //Цвет шрифта
  TextOut(xRight + 20, y0 - 15, 'X'); //Подписываем ось OX
  TextOut(x0 - 10, yLeft - 30, 'Y'); //Подписываем ось OY
  SetFontSize(8); //Размер шрифта
  SetFontColor(clRed); //Цвет шрифта
  { Засечки по оси OX: }
  n := round((b - a) / dx) + 1; //количество засечек по OX
  for i := 1 to n do
  begin
    num := a + (i - 1) * dx; //Координата на оси OX
    x := xLeft + trunc(mx * (num - a)); //Координата num в окне
    Line(x, y0 - 3, x, y0 + 3); //рисуем засечки на оси OX
    str(Num:0:1, s);
    if abs(num) > 1E-15 then //Исключаем 0 на оси OX
      TextOut(x - TextWidth(s) div 2, y0 + 10, s)
  end;
  { Засечки на оси OY: }
  n := round((fmax - fmin) / dy) + 1; //количество засечек по OY
  for i := 1 to n do
  begin
    num := fMin + (i - 1) * dy; //Координата на оси OY
    y := yRight - trunc(my * (num - fmin));
    Line(x0 - 3, y, x0 + 3, y); //рисуем засечки на оси Oy
    str(num:0:0, s);
    if abs(num) > 1E-15 then //Исключаем 0 на оси OY
      TextOut(x0 + 7, y - TextHeight(s) div 2, s)
  end;
  TextOut(x0 - 10, y0 + 10, '0'); //Нулевая точка
  { График функции строим по точкам: }
  x1 := a; //Начальное значение аргумента
  while x1 <= b do
  begin
    y1 := F(x1); //Вычисляем значение функции
    x := x0 + round(x1 * mx); //Координата X в графическом окне
    y := y0 - round(y1 * my); //Координата Y в графическом окне
    //Если y попадает в границы [yLeft; yRight], то ставим точку:
    if (y >= yLeft) and (y <= yRight) then SetPixel(x, y, clGreen);
    x1 := x1 + 0.001 //Увеличиваем абсциссу
  end
end.
```


GraphABC.NET
Y
X
20
18
16
14
12
10
8
6
4
2
-2
-4
-6
-8
-10
-2.0
-1.5
-1.0
-0.5
0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0

```
uses
  Graph; { Подключаем модуль }

function F(x: real): real;
begin
  F := (x + 1) * (x - 2) * (x - 3); { Функция }
end;

var
  Gd, Gm, x0, y0, x, y, xLeft, yLeft, xRight, yRight, n: integer;
  a, b, fmin, fmax, x1, y1, mx, my, dx, dy, num: real;
  i: byte;
  s: string;
begin
  x0 := 0;
  Gd := Detect;
  InitGraph(Gd, Gm, 'C:\tp7\bgi'); { Инициализируем графический режим }
  { Координаты левой верхней границы системы координат: }
  xLeft := 50;
  yLeft := 50;
  { Координаты правой нижней границы системы координат: }
  xRight := GetMaxX - 50;
  yRight := GetMaxY - 50;
  { интервал по X; a и b должно нацело делится на dx: }
  a := -2; b := 6; dx := 0.5;
  { Интервал по Y; fmin и fmax должно нацело делится на dy: }
  fmin := -10; fmax := 20; dy := 2;
  { Устанавливаем масштаб: }
  mx := (xRight - xLeft) / (b - a); { масштаб по X }
  my := (yRight - yLeft) / (fmax - fmin); { масштаб по Y }
  { начало координат: }
  x0 := trunc(abs(a) * mx) + xLeft;
  y0 := yRight - trunc(abs(fmin) * my);
  { Рисуем оси координат: }
  line(xLeft, y0, xRight + 10, y0); { ось OX }
  line(x0, yLeft - 10, x0, yRight); { ось OY }
  SetColor(4); { Цвет шрифта }
  SetTextStyle(1, 0, 1); { Устанавливаем стиль шрифта: }
  OutTextXY(xRight + 20, y0 - 15, 'X'); { Подписываем ось OX }
  OutTextXY(x0 - 15, yLeft - 35, 'Y'); { Подписываем ось OY }
  SetColor(14); { Цвет шрифта }
  { Засечки по оси OX: }
  n := round((b - a) / dx) + 1; { количество засечек по OX }
  for i := 1 to n do
  begin
    num := a + (i - 1) * dx; { Координата на оси OX }
    x := xLeft + trunc(mx * (num - a)); { Координата num в окне }
    Line(x, y0 - 3, x, y0 + 3); { рисуем засечки на оси OX }
    str(Num:0:1, s);
    if abs(num) > 1E-15 then { Исключаем 0 на оси OX }
      OutTextXY(x - TextWidth(s) div 2, y0 + 10, s)
  end;
  { Засечки на оси OY: }
  n := round((fmax - fmin) / dy) + 1; { количество засечек по OY }
  for i := 1 to n do
  begin
    num := fMin + (i - 1) * dy; { Координата на оси OY }
    y := yRight - trunc(my * (num - fmin));
    Line(x0 - 3, y, x0 + 3, y); { рисуем засечки на оси Oy }
    str(num:0:0, s);
    if abs(num) > 1E-15 then { Исключаем 0 на оси OY }
      OutTextXY(x0 + 7, y - TextHeight(s) div 2, s)
  end;
  OutTextXY(x0 - 10, y0 + 10, '0'); { Нулевая точка }
  { График функции строим по точкам: }
  x1 := a; { Начальное значение аргумента }
  while x1 <= b do
  begin
    y1 := F(x1); { Вычисляем значение функции }
    x := x0 + round(x1 * mx); { Координата X в графическом окне }
    y := y0 - round(y1 * my); { Координата Y в графическом окне }
    { Если y попадает в границы [yLeft; yRight], то ставим точку: }
    if (y >= yLeft) and (y <= yRight) then PutPixel(x, y, 12);
    x1 := x1 + 0.001 { Увеличиваем абсциссу }
  end;
  SetColor(15);
  OutTextXY(GetMaxX div 2 - 50, 50, 'y = (x+1)(x-2)(x-3)');
  readln
end.
```



Пятый вопрос: <u>Компьютерное моделирование геометрических и физических задач в графическом режиме.</u>

<u>Компьютерное моделирование падения тела.</u>

```
Uses graphABC,crt;
var X,Y: Integer;
Vx,Vy,K,Vx0: Real;
R:integer;
IsRunning:boolean;
Const
a = 1;
wid=1300;
he=650;
Begin
writeln('Введите радиус шара ');
readln(r);
writeln('Введите высоту падения шара ');
readln(Y);
writeln('Введите начальную скорость шара ');
readln(Vx0);
writeln('Введите коэффициент потери энергии (от 0.5 до 0.9) ');
readln(K);
SetWindowSize(wid,he);
IsRunning:= true;
Vy := 0;
Vx := Vx0;
X := R + 5;
Y := he-Y;
ClearWindow;
setpencolor(clBlack);
SetBrushColor(clRed);
MoveTo(R+5, Y + R);
LineTo(R+5, he);
setpencolor(clBlue);
Ellipse (X-R, Y-R, X + R, Y + R);
Sleep(500);
while IsRunning do
begin
setpencolor(clWhite);
SetBrushColor(clWhite);
ellipse(X-R, Y-R, X+R, Y+R);
X := X + Round(Vx);
Y := Y + Round(Vy);
if X > wid-R then
exit;
if Y > he-R then
begin
Y:= he-R;
Vy:= -Vy * K;
if abs(Vy) < 0.1 then exit;
end;
Vy := Vy + a;
setpencolor(clBlue);
SetBrushColor(clRed);
Ellipse(X-R, Y-R, X+R, Y+R);
Sleep(30);
end;
end.
```

Седьмой вопрос: Составление программ для графической интерпретации и исследования физических моделей (практическое занятие № 28, теоретическая часть, выполнение практического задания).

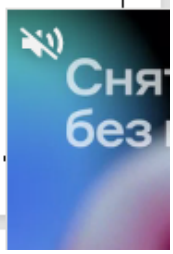
задача "Бросание мячика в стенку"
нужно построить график, который будет отражать траекторию полета мяча...
вот собственно сама программа:

PascalВыделить код

```
program myach;
var A,H,S,V0,g,L:real;
begin
g:=9.81;
readln(A,H,S,V0);
L:=S*Sin(A*Pi/180)/Cos(A*Pi/180)-G*Sqr(S)/(2*Sqr(V0*Cos(A*Pi/180)));
writeln(L);
if L<0 then
   writeln('Недолет')
else if L>H then
   writeln('Перелет')
else
   writeln('Попадание');
end.
```

2.1. Тело правильной формы (куб, шар)

	Этап моделирования	Результат
1	Постановка задачи	Цель: рассчитать плотность тела, поведение в жидкости, определить из какого вещества состоит тело. Оборудование: тело, весы, штангенциркуль; Исходные данные: масса, диаметр тела Результат: программа на языке Pascal, получающая на вход массу и диаметр тела, на выходе – плотность, поведение в жидкости, вещество.
2	Построение информационной модели	Плотность тела: $\rho = \frac{m}{V}$, m – масса тела , V – объем. Для шара $V = \frac{4}{3}\pi R^3 = \frac{\pi D^3}{6}$, где R, D – радиус и , шара.



		Для куба $V = a^3$, где a – ребро куба.
3	Алгоритм реализации компьютерной модели	1. Определяем количество входных данных, констант, задаем типы переменных; 2. Вводим: 1-если шар, 2- если куб, массу, диаметр шара (или ребро куба); 3. Используем алгоритм ветвления с использованием операторных скобок для выбора формы тела; 4. Переводим объем в $см^3$ для удобства расчетов; 5. Рассчитываем плотность pl в $кг/м^3$, получаем на выходе число типа real; 6. Для использования оператора case преобразовываем значение плотности в число типа integer, для чего используем функцию round. 7. Выводим: плотность, предполагаемый род вещества, поведение в воде.
4	Разработка компьютерной модели	См. Приложение 1
5	Проведение эксперимента	См. Приложение 2

2.1. Тело правильной формы (шар, куб)

```
program SHAR_KUB;
var m,d,v,k,vk,pl: real;
a,x:integer;
const pi=3.1415;
begin
writeln ('Программа для вычисления плотности однородного тела шарообразной или кубической формы');
writeln ('по измеренным значениям массы, диаметра, длине ребра');
writeln ('Если ШАР - введите 1, если КУБ - введите 2');
readln (x);
if x=1 then begin
writeln ('Введите массу шара в граммах:  ');
readln (m);
writeln ('Введите диаметр шара в миллиметрах:  ');
readln (d);
v:=(pi*d*d*d)/6000;     {Вычисляем объем тела в см3}
```

```
pl:=(m*1000)/v;        {Вычисляем плотность тела в кг/м3}
end
else begin
writeln ('Введите длину ребра куба в миллиметрах:  ');
readln (k);
writeln ('Введите массу куба в граммах:  ');
readln (m);
vk:=(k*k*k)/1000;    {Вычисляем объем куба в см3}
pl:=(m*1000)/vk;     {Вычисляем плотность тела в кг/м3}
end;
writeln ('_____');
writeln ('1) Плотность тела =',pl:0:3,' кг/м3');     {pl:0:4 - задаем количество знаков после запятой}
a:= round (pl);     {ROUND  - стандартная функция Паскаль, округляет вещественное число (REAL)}
                    {до целого числа (INTEGER). Необходима в случае с оператором CASE }
case a of
11200..11400: writeln ('2) Вероятно тело из свинца');
8800..9000: writeln ('2) Вероятно тело из меди');
```

```
8401..8600: writeln ('2) Вероятно тело из латуни');
7600..8400: writeln ('2) Вероятно тело из стали (железа)');
7200..7400: writeln ('2) Вероятно тело из олова');
7051..7190: writeln ('2) Вероятно тело из цинка');
6900..7050: writeln ('2) Вероятно тело из чугуна');
6300..6899: writeln ('2) Вероятно тело из карбида хрома');
2600..2800: writeln ('2) Вероятно тело из алюминия или мрамора');
2400..2590: writeln ('2) Вероятно тело из стекла');
2200..2390: writeln ('2) Вероятно тело из фарфора');
1100..1300: writeln ('2) Вероятно тело из оргстекла или легкой пластмассы (поликарбонат, капрон, полиуретан, полистирол, полипропилен)');
1301..1500: writeln ('2) Вероятно тело из плотной пластмассы (полиформальдегид, поливинилхлорид (ПВХ) и пр) или пластилиновый');
800..990: writeln ('2) Вероятно тело из парафина или льда');
600..790: writeln ('2) Вероятно тело из сухого дуба');
351..500: writeln ('2) Вероятно тело из сухой сосны');
150..350: writeln ('2) Вероятно тело из пробки');
```

```
5..100: writeln ('2) Вероятно тело из пенопласта');
end;
if pl<1000 then writeln ('3) Тело плавает на поверхности воды');        {Определяем условие плавания тел}
if pl=1000 then writeln ('3) Тело плавает в толще воды');
if pl>1000 then writeln ('3) Тело тонет в воде');
end.
```