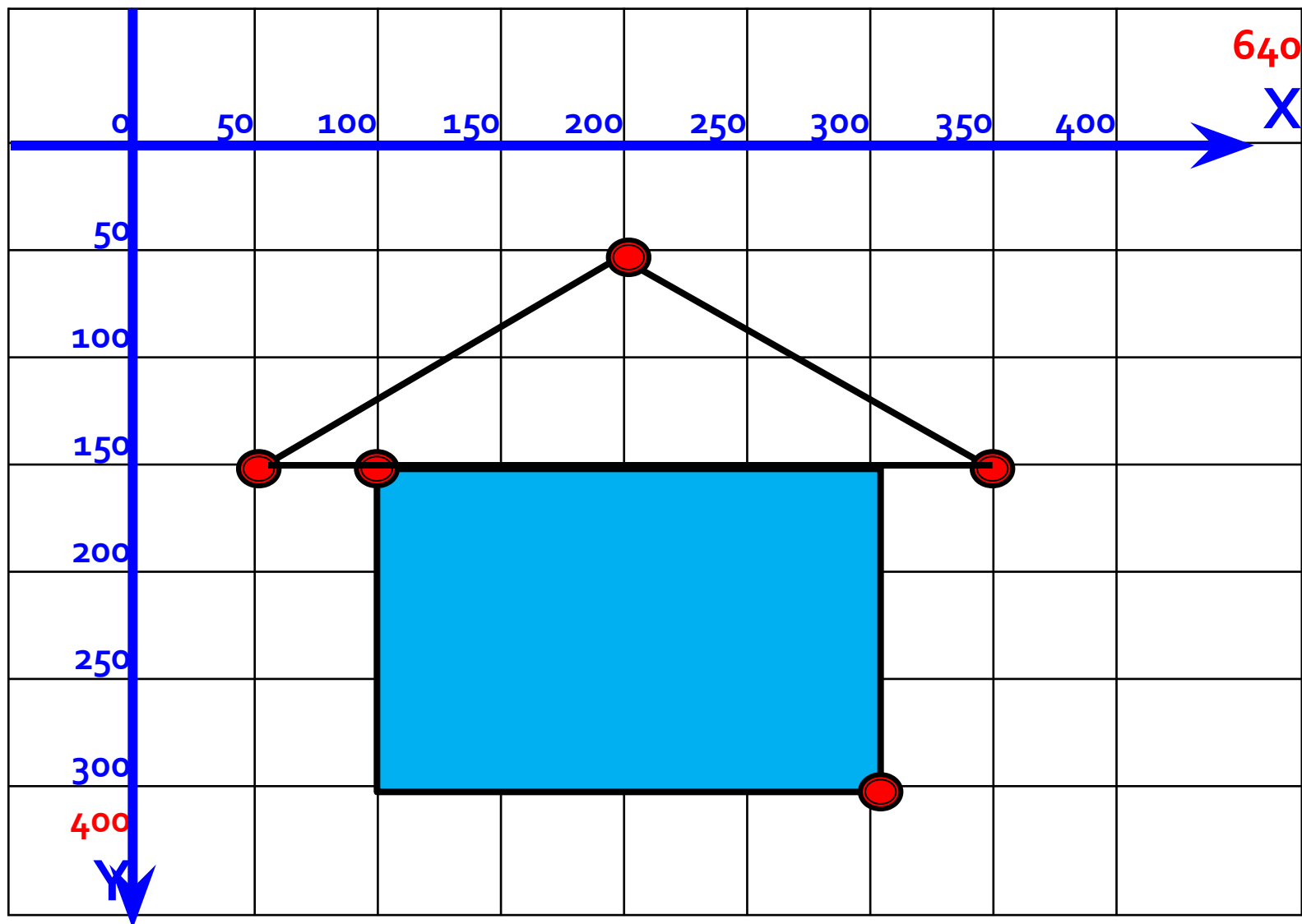


# Графические возможности языка программирования Pascal ABC

Тема урока



# Подключение дополнительных библиотек

Для работы в графическом режиме необходимо подключение модуля GraphABC.

Первой инструкцией программы должна быть инструкция

***uses GraphABC;***

```
Program clear;  
uses GraphABC;  
Begin  
  
End.
```

# Управление экраном

## ***SetWindowWidth(w)***

**Устанавливает ширину графического окна;**

## ***SetWindowHeight(h)***

**Устанавливает высоту графического окна;**

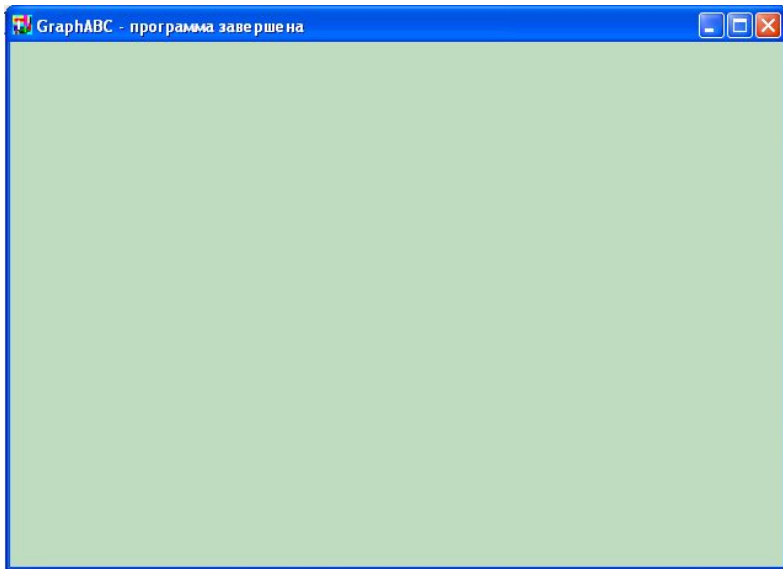
# Очистка графического окна

***ClearWindow;***

очищает графическое окно белым цветом.

***ClearWindow(color);***

очищает графическое окно указанным цветом.



Цвет зеленых денег

```
Program clear;  
uses GraphABC;  
begin
```

```
ClearWindow;  
ClearWindow  
(clMoneyGreen);
```

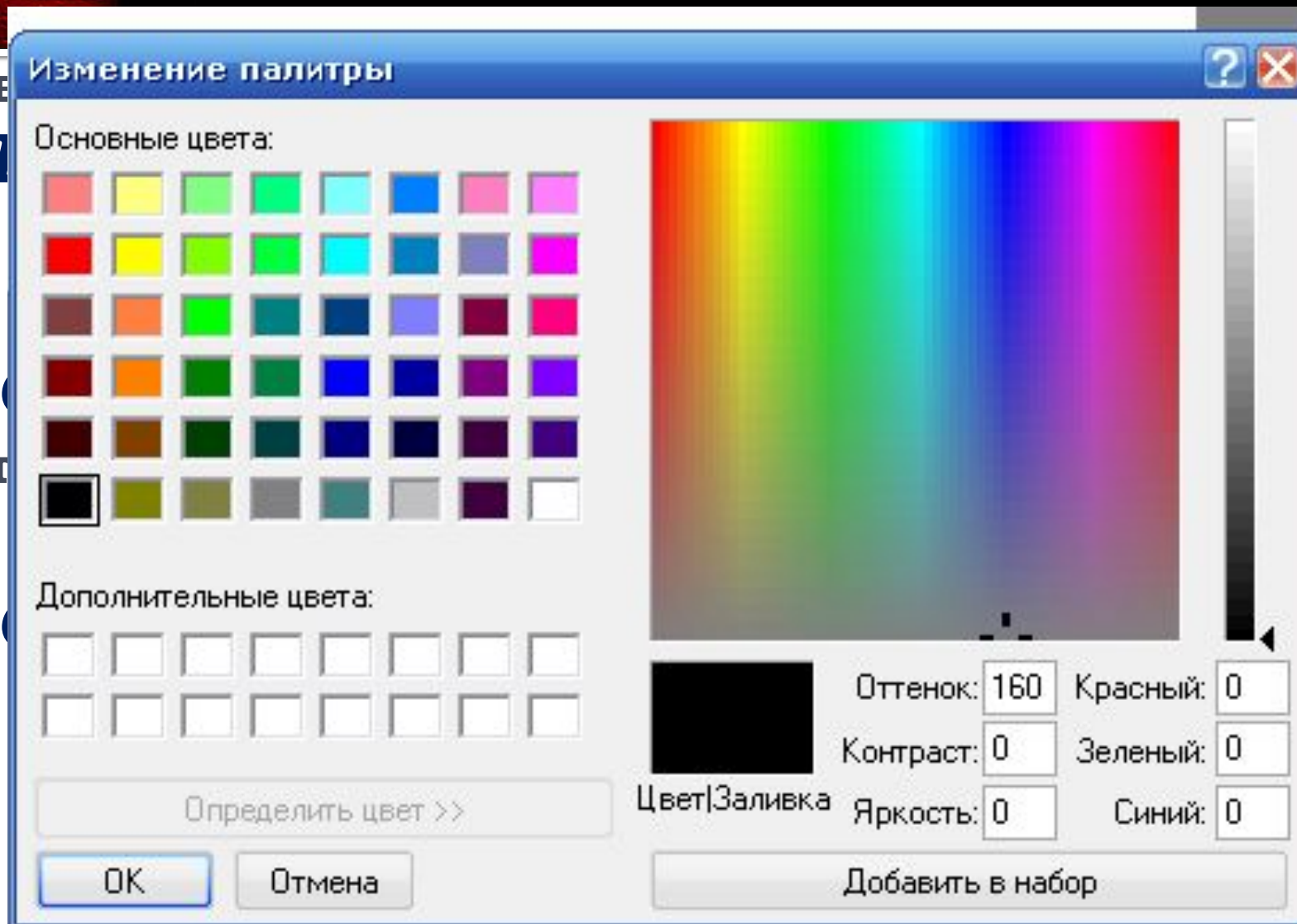
```
End.
```

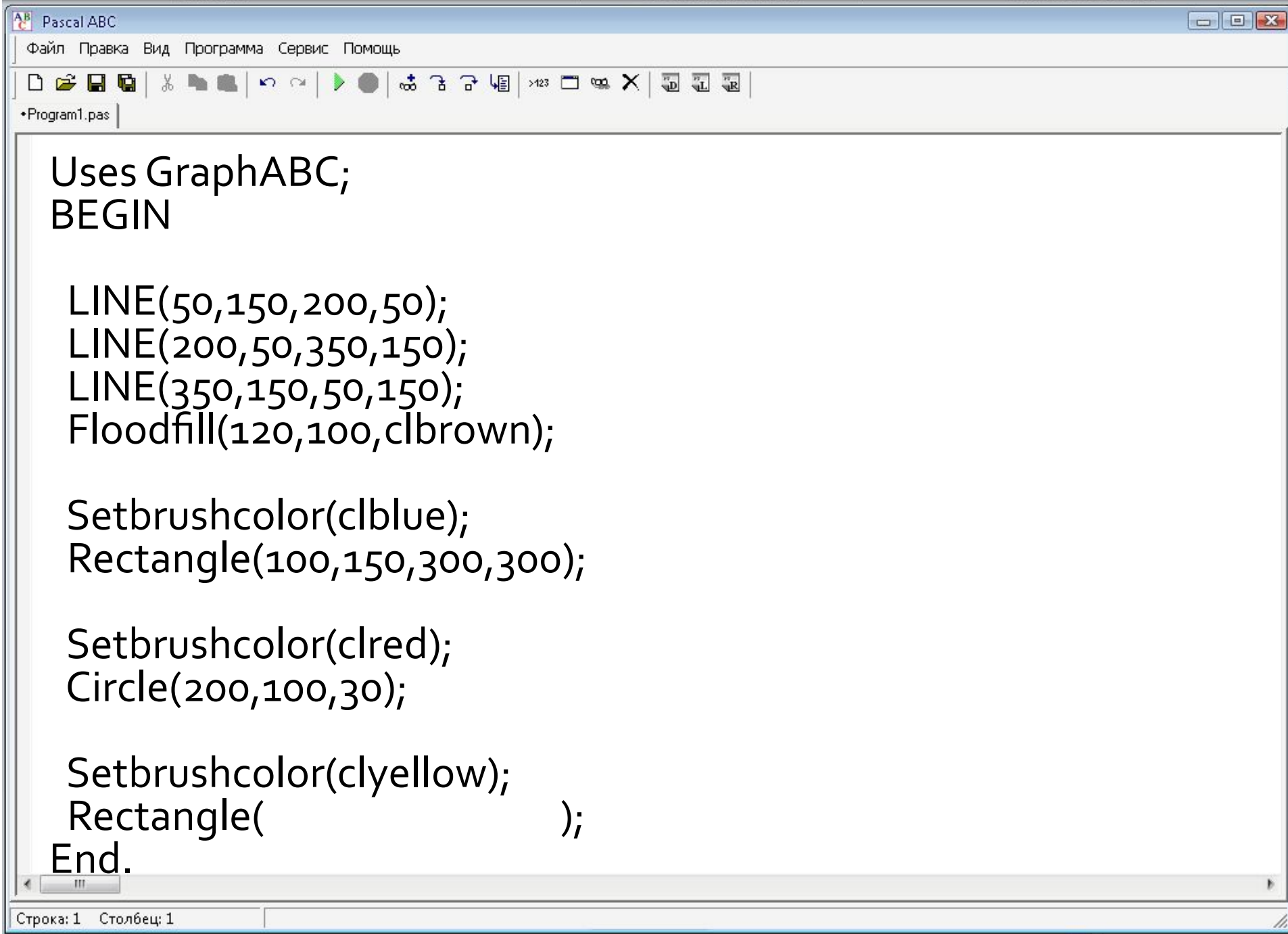
# Цвета

**clBlack** – черный  
**clPurple** – фиолетовый  
**clWhite** – белый  
**clMaroon** – темно-красный  
**clRed** – красный  
**clNavy** – темно-синий  
**clGreen** – зеленый  
**clBrown** – коричневый  
**clBlue** – синий  
**clSkyBlue** – голубой

**clAqua** – бирюзовый  
**clOlive** – оливковый  
**clFuchsia** – сиреневый  
**clTeal** – сине-зеленый  
**clGray** – темно-серый  
**clLime** – ярко-зеленый  
**clMoneyGreen** – цвет зеленых денег  
**clLtGray** – светло-серый  
**clDkGray** – темно-серый  
**clYellow** – желтый

# Используемые цвета



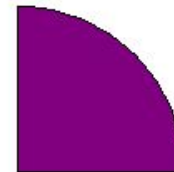
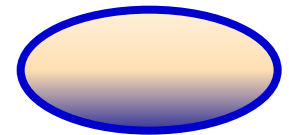
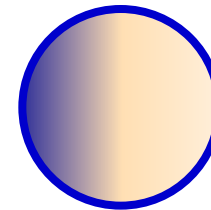
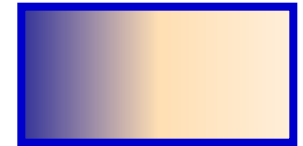
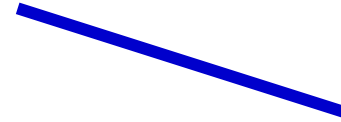




# Графические примитивы



1. Точка
2. Линия
3. Прямоугольник
4. Окружность
5. Эллипс
6. Сектор
7. Дуга



# Точка.

***SetPixel(x,y,color)***

Закрашивает один пиксел с координатами (x,y) цветом color



```
program точка;  
uses GraphABC;  
begin
```

```
SetPixel(300,200,clred)
```

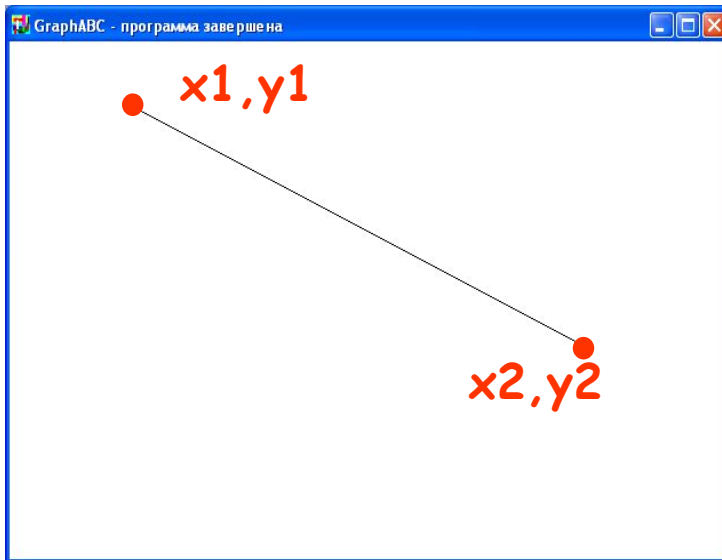
```
;
```

```
end.
```

# Линии

***Line(x1,y1,x2,y2)***

рисует отрезок с началом в точке (x1,y1) и концом в точке (x2,y2).

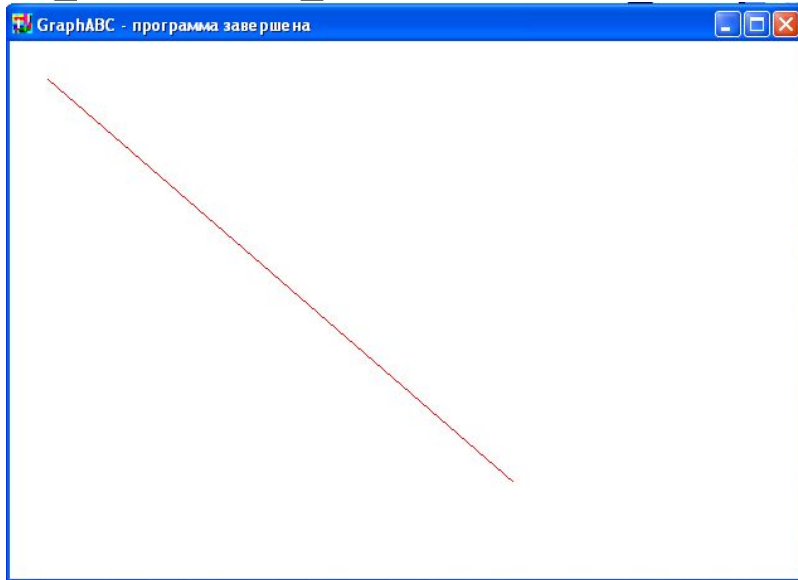


```
Program linia;  
uses GraphABC;  
begin  
line(100,50,500,250);  
end.
```

# Цвет линии

## ***SetPenColor(color)***

устанавливает цвет пера, задаваемый параметром **color**.

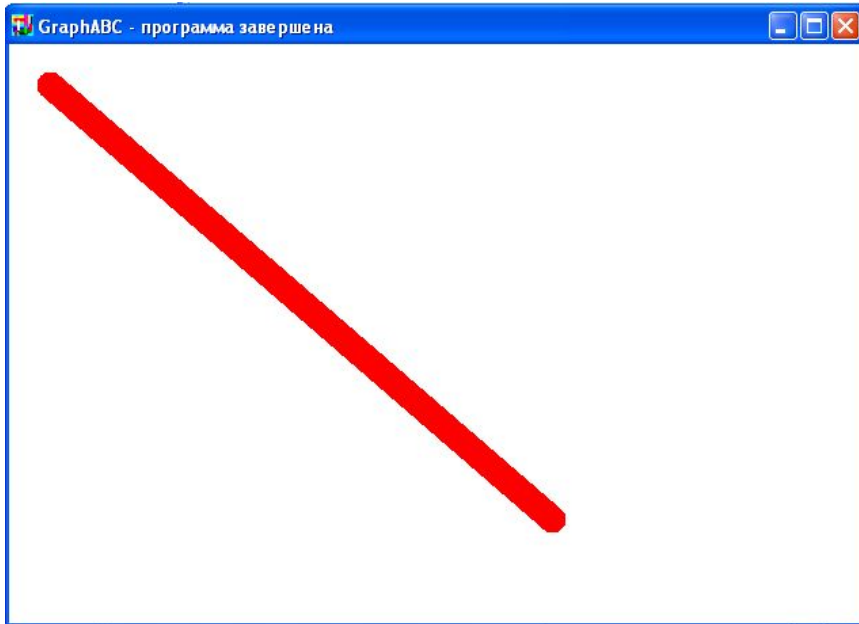


```
Program linia;  
uses GraphABC;  
begin  
    setpencolor(clred);  
    line(30,30,400,350);  
end.
```

# Толщина линии

## *SetPenWidth(n)*

устанавливает ширину (толщину) пера, равную *n* пикселям.

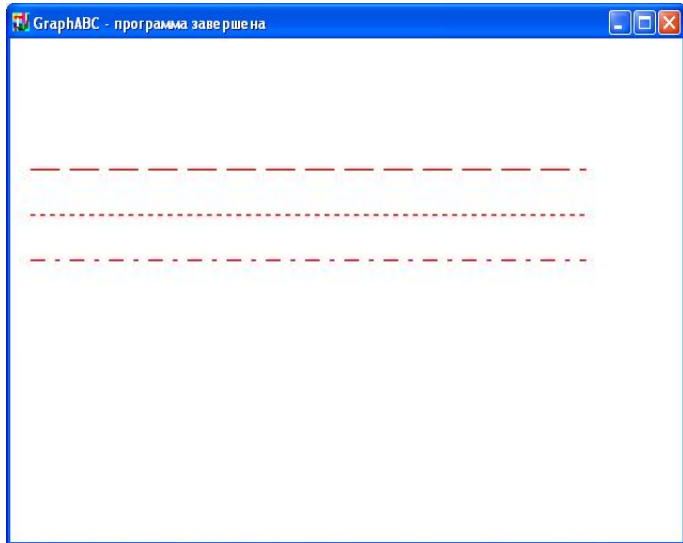


```
Program liniay;  
uses GraphABC;  
begin  
    setpenwidth(20);  
    setpencolor(clred);  
    line(30,30,400,350);  
end.
```

# Пунктирная линия

**SetPenStyle(<номер от 1 до 6>);** -

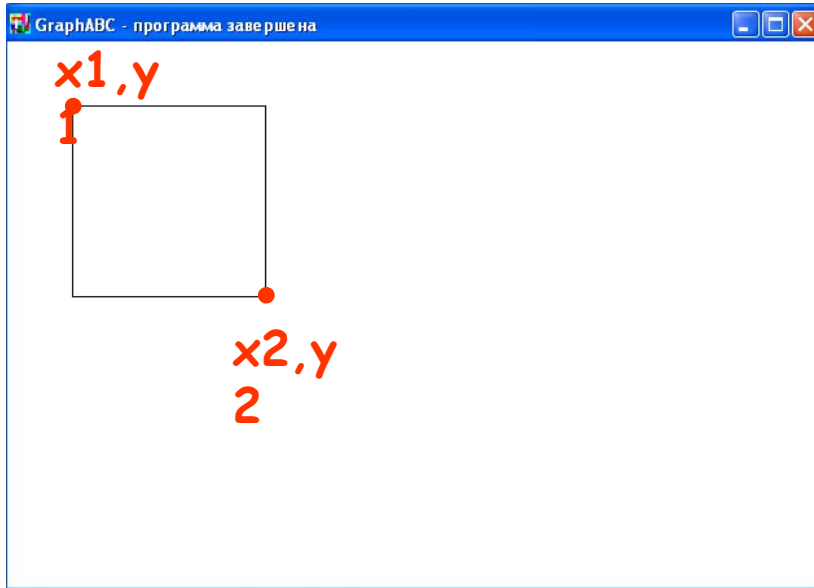
устанавливает стиль пера, задаваемый номером.



```
program prim;
uses GraphABC;
begin
  Setpencolor(clred);
  SetPenStyle(1); {1 - длинный штрих}
  Line(10,100,350,100);
  SetPenStyle(2); {2 - короткий штрих}
  Line(10,125,350,125);
  SetPenStyle(3); {3 - штрих-пунктир}
  Line(10,150,350,150);
end.
```

# Прямоугольник.

***Rectangle(x1,y1,x2,y2)*** - рисует прямоугольник, заданный координатами противоположных вершин (x1,y1) и (x2,y2).

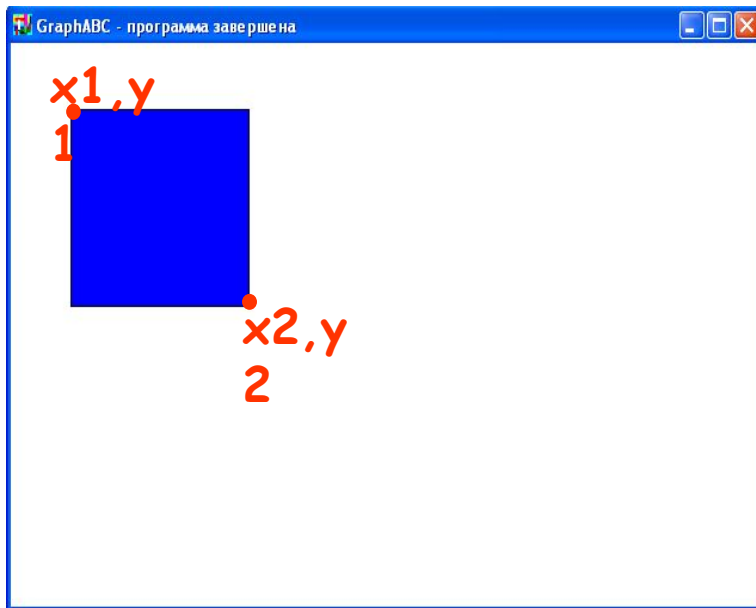


```
Program pryamougolnik;  
uses GraphABC;  
begin  
    Rectangle(50,50,200,200);  
end.
```

# Заливка цветом

## ***FloodFill(x,y,color)***

заливает область одного цвета цветом color, начиная с точки (x,y).



```
Program pryamougolnik;  
uses GraphABC;  
begin  
    Rectangle(50,50,200,200);  
    FloodFill(100,100,clBlue);  
end.
```

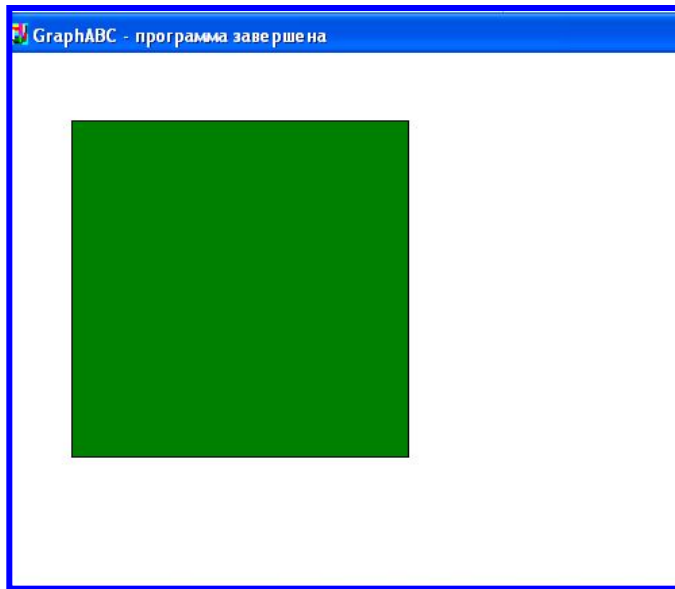


# Заливка кистью.

## ***SetBrushColor(color)***

устанавливает цвет кисти.

Заливка кистью распространяется на замкнутый контур, описание которого следует за процедурой установки цвета кисти.

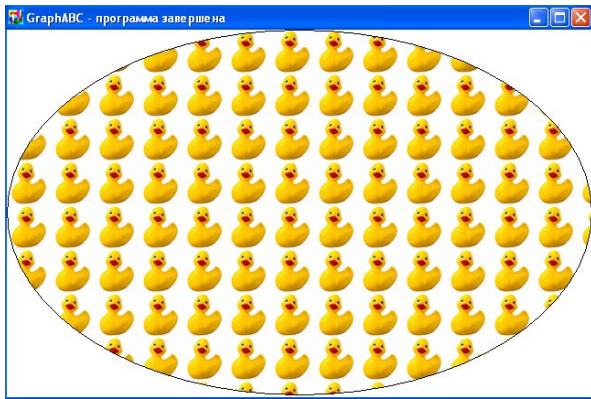
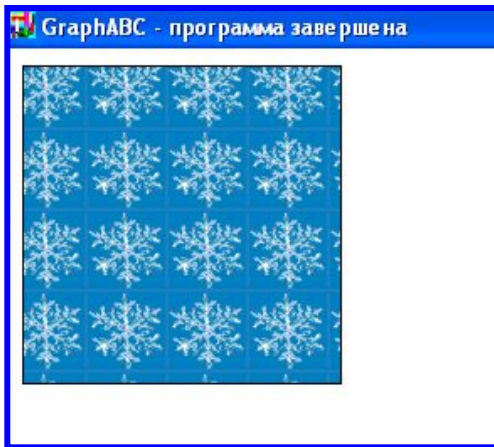


```
Program zalivka_kist;  
uses GraphABC;  
Begin  
SetBrushColor(clGreen);  
Rectangle(50,50,300,300);  
end.
```

# Заливка кистью

## ***SetBrushPicture('fname')***

устанавливает в качестве образца для закрашки кистью образец, хранящийся в файле `fname`, при этом текущий цвет кисти при закрашке игнорируется.



```
uses GraphABC;
```

```
begin
```

```
SetBrushPicture('brush4.bmp');
```

```
Ellipse(0,0,640,400);
```

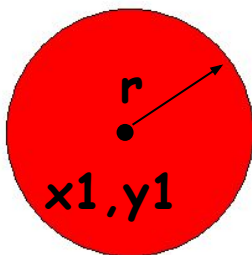
```
end.
```

# Окружность

***Circle(x,y,r)***

рисует окружность с центром в точке (x,y) и радиусом r.

GraphABC - программа завершена

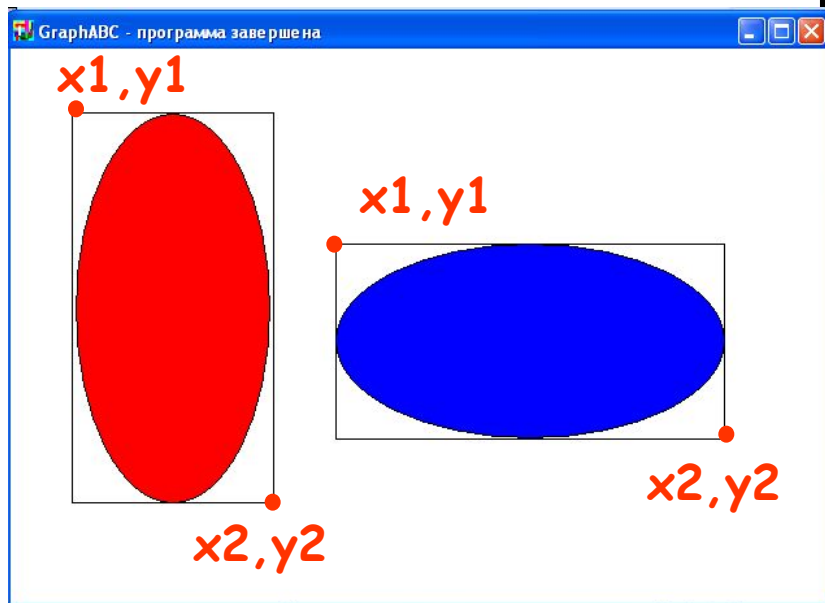


```
Program circle;  
uses GraphABC;  
begin  
    Circle(500,200,100);  
    FloodFill(500,200,clred);  
end.
```

# Эллипс

***Ellipse(x1,y1,x2,y2)***

рисует эллипс, заданный своим описанным прямоугольником с координатами противоположных вершин (x1,y1) и (x2,y2).



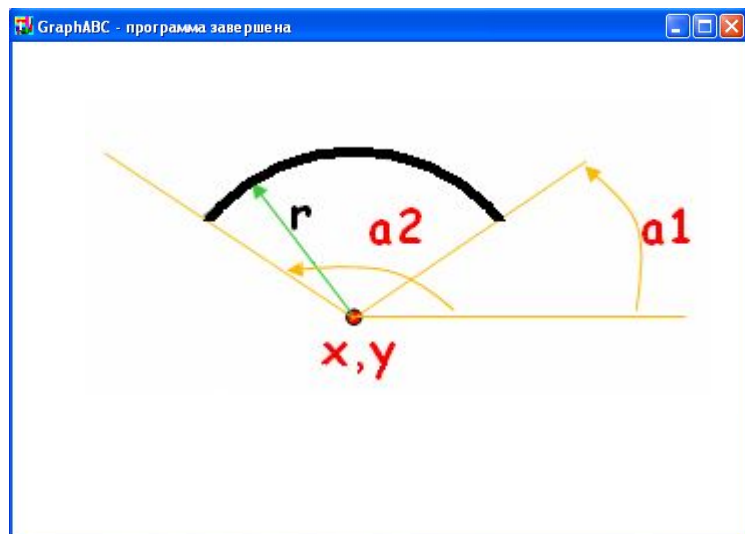
```
Program oval;  
uses GraphABC;  
Begin  
  Ellipse(50,50,200,350);  
  FloodFill(50+100,50+100,clred);  
  Ellipse(250,150,550,300);  
  FloodFill(250+100,150+100,clBlue);  
end.
```

# Дуга окружности

## ***Arc(x,y,r,a1,a2)***

Рисует дугу окружности с центром в точке (x,y) и радиусом r, заключенной между двумя лучами, образующими углы a1 и a2 с осью OX

(a1 и a2 – вещественные, задаются в градусах и отсчитываются против часовой стрелки).

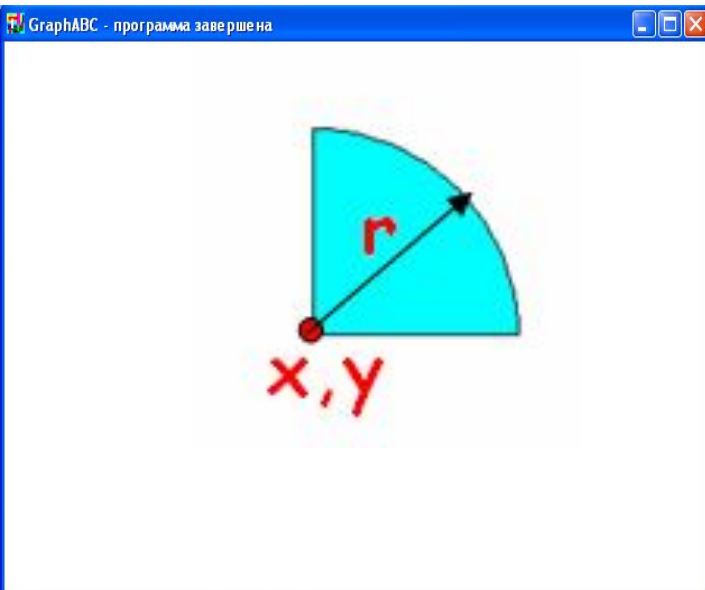


```
Program duga;  
uses GraphABC;  
Begin  
SetPenWidth(10);  
Arc(300,250,150,45,135);  
end.
```

# Сектор

***Pie(x,y,r,a1,a2)***

рисует сектор окружности, ограниченный дугой  
(параметры процедуры имеют тот же смысл, что и в  
процедуре Arc).



```
Program sector;  
uses GraphABC;  
begin  
Pie(300,200,100,0,90);  
FloodFill(300+10,200-10,clAqua);  
end.
```

# Вывод текста в графическое окно

***TextOut(x,y,'строка');***

**ВЫВОДИТ строку текста в позицию (x,y)**

**(точка (x,y) задает верхний левый угол прямоугольника, который будет содержать текст).**



```
Program text;  
uses GraphABC;  
begin  
TextOut(100,30,'Квадрат');  
Rectangle(50,50,200,200);  
FloodFill(55,55,clBlue);  
end.
```

# Вывод текста в графическое окно

Текст можно вывести с помощью операторов *Gotoxy(x,y)* и *Write('текст')*, подключив дополнительно модуль *Crt*.



```
Program text2;  
uses Crt, GraphABC;  
begin  
  clrscr;  
  hidescursor; {скрывает  
текстовый курсор}  
  gotoXY(12,3);  
  write('Квадрат');  
  Rectangle(50,50,200,200);  
  FloodFill(55,55,clYellow);  
end.
```



# Форматирование текста

***SetFontName('name')*** - устанавливает наименование шрифта.

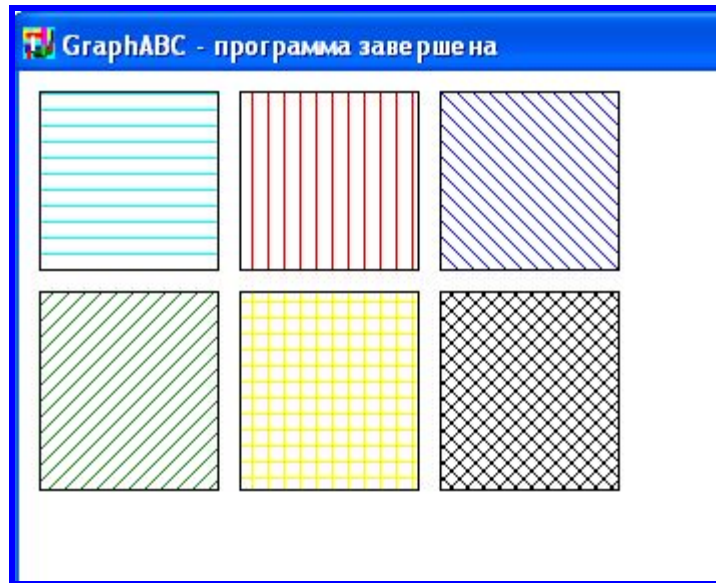
***SetColor(color)*** - устанавливает цвет шрифта.

***SetFontSize(sz)*** - устанавливает размер шрифта в пунктах.

***SetFontStyle(fs)*** - устанавливает стиль шрифта.

# Заливка кистью

**SetBrushStyle(номер от 0 до 7 или название)** - устанавливает стиль кисти, задаваемый номером или символической константой.



По умолчанию задается стиль 0 – сплошная заливка цветом.

```
Program p12_zalivka;  
uses GraphABC;  
Begin  
  SetBrushColor(clAqua);  
  SetBrushStyle(1);  
  Rectangle(10,10,100,100);  
  SetBrushColor(clRed);  
  SetBrushStyle(2);  
  Rectangle(110,10,200,100);  
  SetBrushColor(clBlue);  
  SetBrushStyle(3);  
  Rectangle(210,10,300,100);  
  SetBrushColor(clGreen);  
  SetBrushStyle(4);  
  Rectangle(10,110,100,210);  
  SetBrushColor(clYellow);  
  SetBrushStyle(5);  
  Rectangle(110,110,200,210);  
  SetBrushColor(clBlack);  
  SetBrushStyle(6);  
  Rectangle(210,110,300,210);  
end.
```

# Действия со шрифтом

**SetFontName('name')** - устанавливает наименование шрифта.

**SetFontColor(color)** - устанавливает цвет шрифта.

**SetFontSize(sz)** - устанавливает размер шрифта в пунктах.

**SetFontStyle(fs)** - устанавливает стиль шрифта.

# Название шрифта

По умолчанию установлен шрифт, имеющий наименование MS Sans Serif.

Наиболее распространенные шрифты – это Times, Arial и Courier New. Наименование шрифта можно набирать без учета регистра.

Пример:

```
SetFontName('Times');
```

# Стиль шрифта

Задается именованными константами:

**fsNormal** – обычный;

**fsBold** – жирный;

**fsItalic** – наклонный;

**fsBoldItalic** – жирный наклонный;

**fsUnderline** – подчеркнутый;

**fsBoldUnderline** – жирный подчеркнутый;

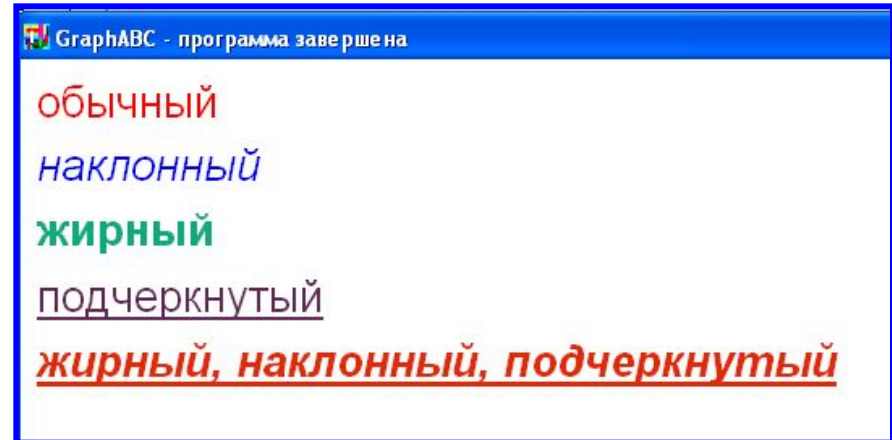
**fsItalicUnderline** – наклонный подчеркнутый;

**fsBoldItalicUnderline** – жирный наклонный  
подчеркнутый.

# Например,

Program text;  
uses GraphABC;  
Begin

```
SetFontName('Arial');  
SetFontSize(20);  
SetFontColor(clRed);  
TextOut(10,10,'обычный');  
SetFontStyle(fsItalic);  
SetFontColor(clBlue);  
TextOut(10,50,'наклонный');  
SetFontStyle(fsBold);  
SetFontColor(Random(16777215));  
TextOut(10,90,'жирный');  
SetFontStyle(fsUnderline);  
SetFontColor(Random(16777215));  
TextOut(10,130,'подчеркнутый');  
SetFontStyle(fsBoldItalicUnderline);  
SetFontColor(Random(16777215));  
TextOut(10,170,'жирный, наклонный, подчеркнутый');  
end.
```



# Загрузка готового рисунка

`LoadPicture(fname)`

**`n:=LoadPicture(fname)`** –

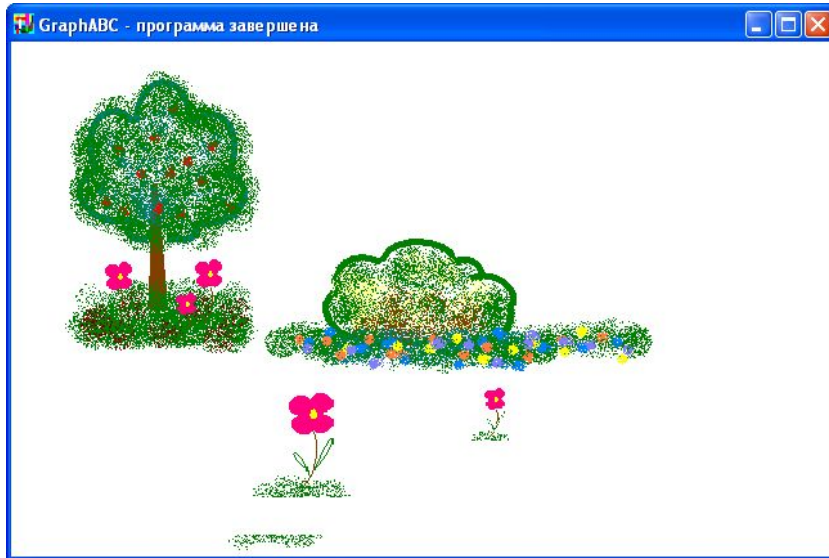
загружает рисунок из файла с именем `fname` в оперативную память и возвращает описатель рисунка в целую переменную `n`; если файл не найден, то возникает ошибка времени выполнения.

Загружать можно рисунки в формате `.bmp`, `.jpg` или `.gif`.

# Вывод рисунка в графическое ОКНО

## DrawPicture(n,x,y);

Выводит рисунок с описателем n в позицию (x,y) графического окна.



```
uses GraphABC;  
var pic: integer;  
begin  
    pic:=LoadPicture('demo.bmp');  
    DrawPicture(pic,10,10);  
    DestroyPicture(pic);  
end.
```



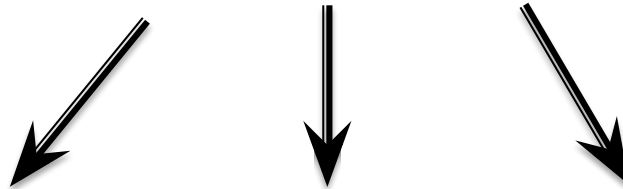
# Сохранение созданного рисунка

`SavePicture(n, 'fname')` -

**Сохраняет рисунок с описателем `n` в файл с именем `fname`. Рисунки можно сохранять в формате `.bmp`, `.jpg` или `.gif`.**

# Случайный выбор цвета

RGB (r,g,b)



Random(255)

Random(255)

Random(255)

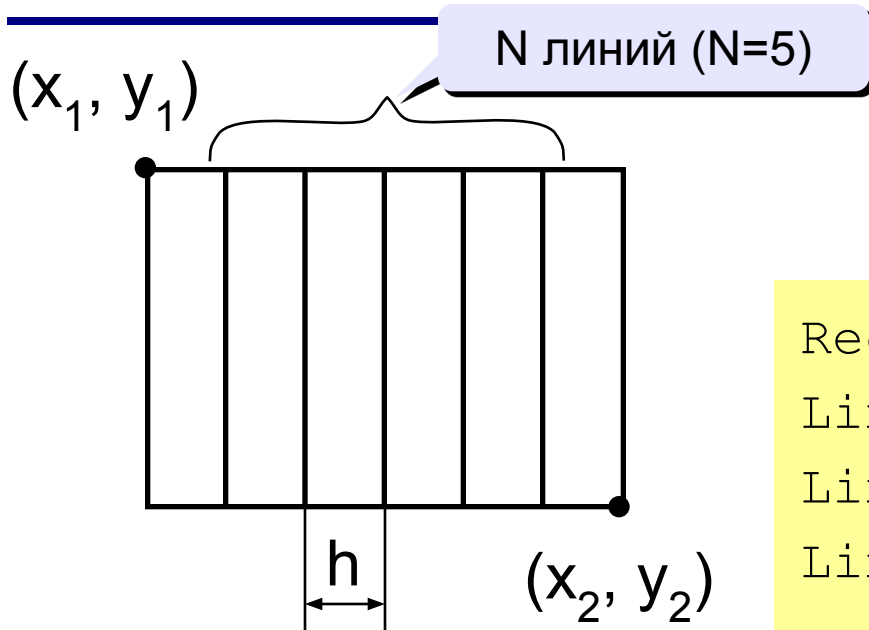
RGB (Random(255), Random(255), Random(255))

Или **CLRANDOM**,  
например,  
**Setbrushcolor(CLRANDOM)**

# Циклы в графике

```
For i:=1 to 10 do begin  
  SetBrushColor(clAqua);  
  SetBrushStyle(1);  
  Circle( i*10, 20,10);  
  
  SetBrushColor(clBlue);  
  SetBrushStyle(3);  
  Rectangle(i*10,50,300,100);  
End;
```

# Штриховка



$$h = \frac{x_2 - x_1}{N + 1}$$

```
Rectangle (x1, y1, x2, y2);  
Line ( x1+h, y1, x1+h, y2);  
Line ( x1+2*h, y1, x1+2*h, y2);  
Line ( x1+3*h, y1, x1+3*h, y2);  
...
```

x

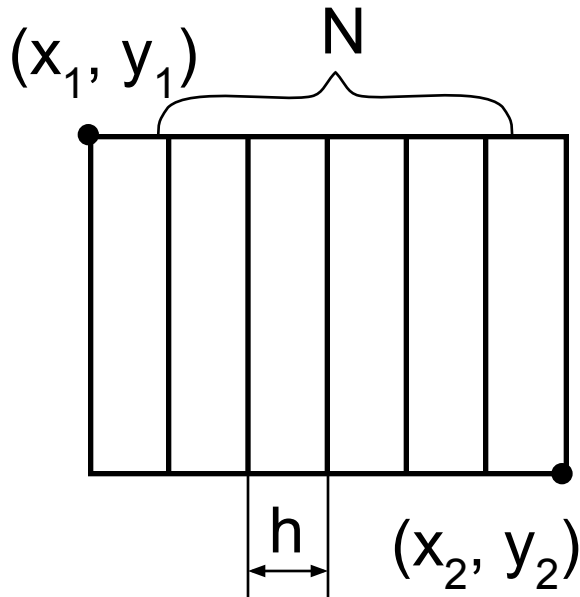
x

```
Rectangle (x1, y1, x2, y2);  
h := (x2 - x1) / (N + 1);  
x := x1 + h;  
for i:=1 to N do begin  
    Line( round(x), y1, round(x), y2);  
    x := x + h;  
end;
```

var x, h: real;

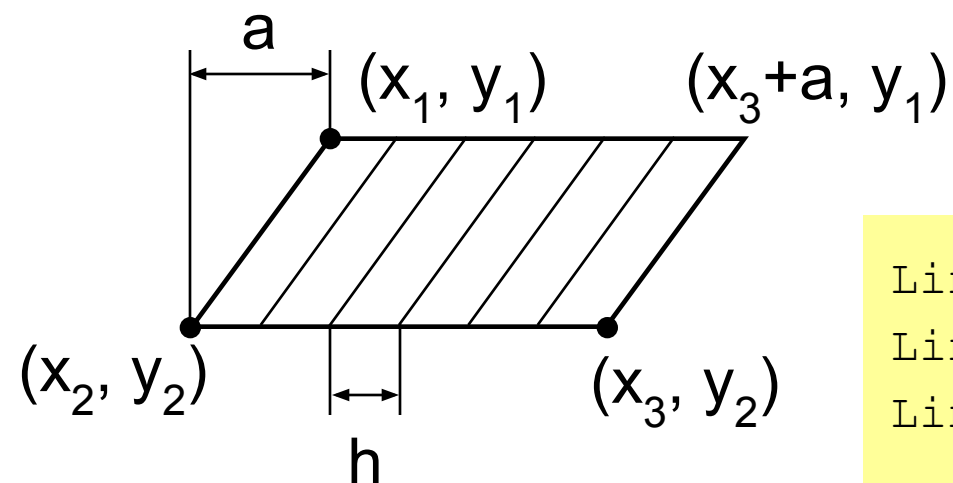
округление до  
ближайшего целого

# Штриховка (программа)



```
program qq;  
var i, x1, x2, y1, y2, N: integer;  
    h, x: real;  
begin  
    write('введите координаты  
прямоугольника и число штрихов');  
    Readln(x1, y1, x2, y2, N);  
    Rectangle (x1, y1, x2, y2);  
    h := (x2 - x1) / (N + 1);  
    x := x1 + h;  
    for i:=1 to N do begin  
        Line(round(x), y1, round(x), y2);  
        x := x + h;  
    end;  
end.
```

# Штриховка



$$a = x_1 - x_2$$

$$h = \frac{x_3 - x_2}{N + 1}$$

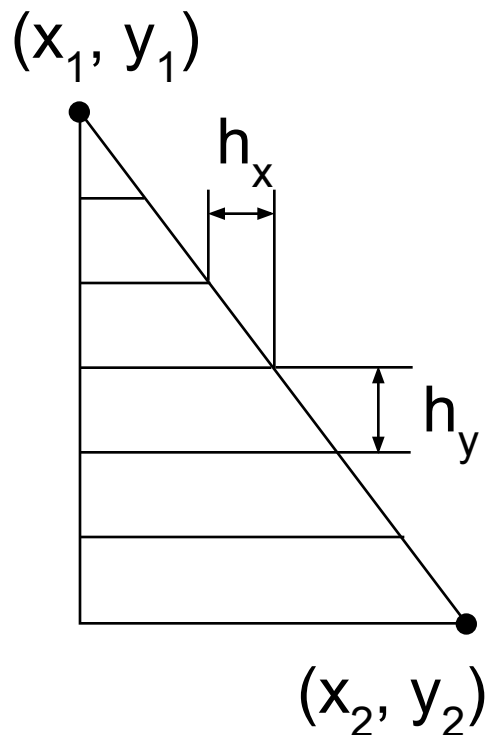
```
Line( x1+h, y1, x1+h-a, y2);  
Line( x1+2*h, y1, x1+2*h-a, y2);  
Line( x1+3*h, y1, x1+3*h-a, y2);  
...
```

$x$

$x-a$

```
h := (x3 - x2) / (N + 1);  
a := x1 - x2;  
x := x1 + h;  
for i:=1 to N do begin  
  Line(round(x), y1, round(x-a), y2);  
  x := x + h;  
end;
```

# Штриховка



$$h_x = \frac{x_2 - x_1}{N + 1}$$

$$h_y = \frac{y_2 - y_1}{N + 1}$$

```
Line( x1, y1+hy, x1+hx, y1+hy)
;
Line( x1, y1+2*hy, x1+2*hx,
y1+2*hy);
Line( x1, y1+3*hy, x1+3*hx,
y1+3*hy);
```

```
hx := (x2 - x1) / (N + 1);
hy := (y2 - y1) / (N + 1);
x := x1 + hx; y := y1 + hy;
for i:=1 to N do begin
    Line(x1, round(y), round(x), round(y));
    x := x + hx; y := y + hy;
end;
```