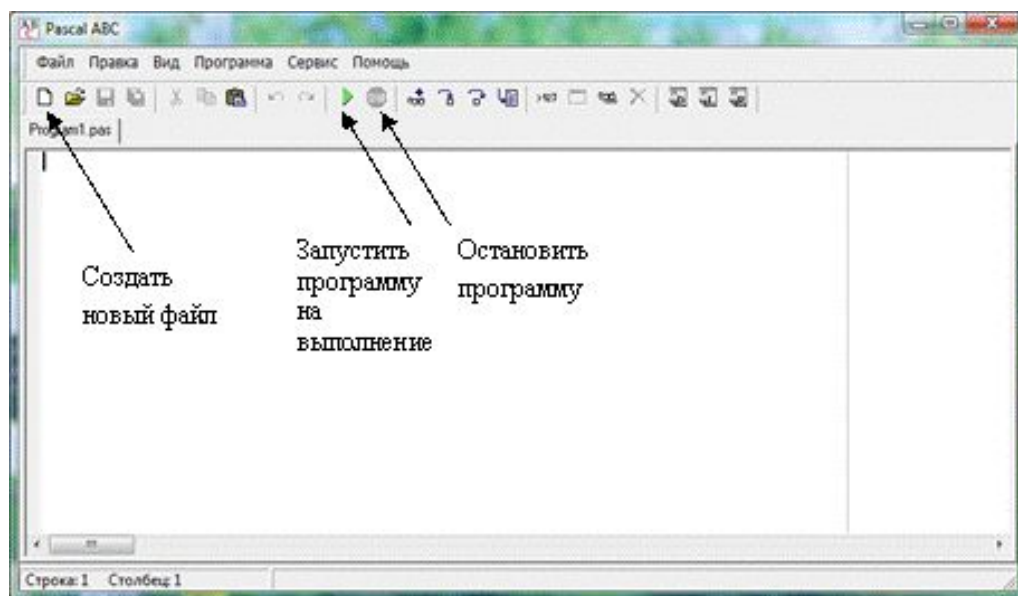


# Pascal ABC

## Модуль GraphABC

Для запуска Паскаль ABC необходимо запустить ярлык Pascal ABC. На экране появится среда программирования Паскаль ABC (оболочка). Среда программирования – это пакет взаимосвязанных файлов, которые позволяют набирать, редактировать, запускать и отлаживать программы.

После запуска ярлыка на рабочем столе открывается окно:



Первая строка экрана – меню интегрированной среды, следующая строка – панель инструментов, нижняя строка экрана – строка подсказки и состояния интегрированной среды. Между ними расположено окно редактирования – рабочее поле, в котором можно открывать несколько вкладок для разных программ.

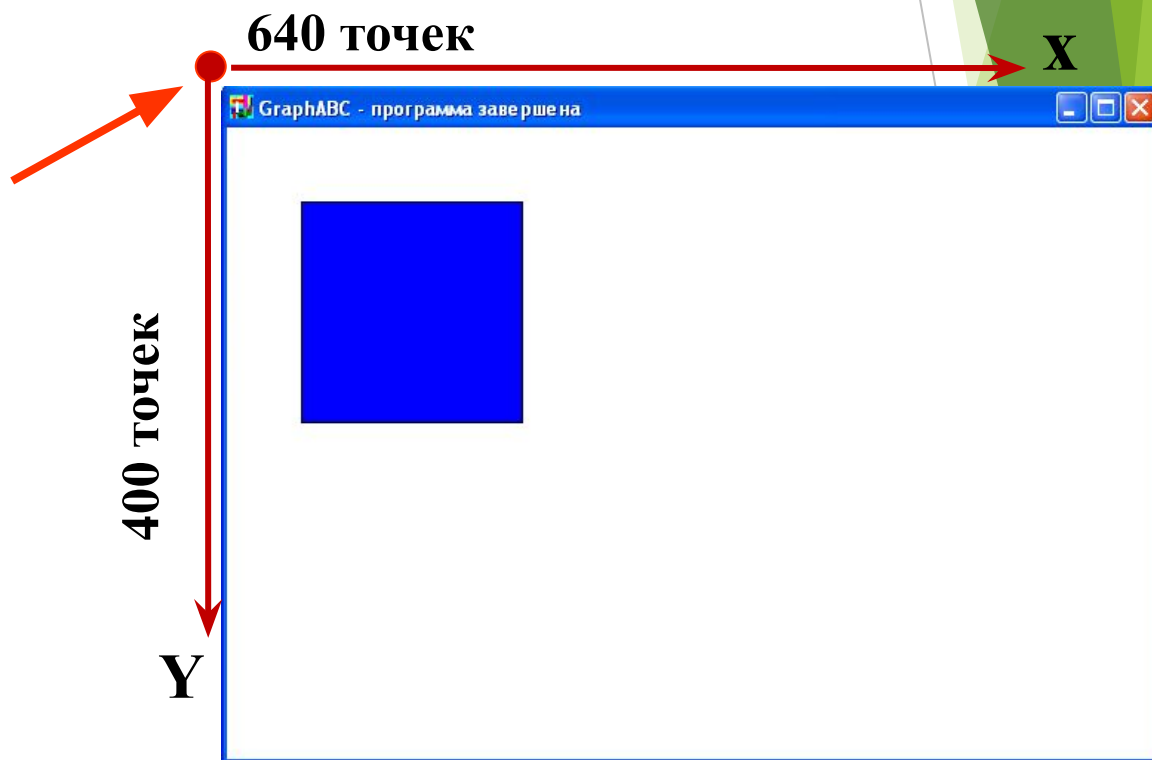
# Подключение дополнительных библиотек

Для работы в графическом режиме необходимо подключение модуля **GraphABC**.  
Первой инструкцией программы должна быть инструкция **uses GraphABC;**

# Графический режим

Графический экран PascalABC  
(по умолчанию) содержит **640 точек**  
по горизонтали и **400 точек**  
по вертикали.

Начало отсчета –  
левый верхний  
угол экрана



# Управление экраном

**SetWindowSize(x,y)** - Устанавливает ширину и высоту графического окна;

# Очистка графического окна

**ClearWindow;** - очищает графическое окно белым цветом.

**ClearWindow(color);** - очищает графическое окно указанным цветом.



Цвет зеленых денег

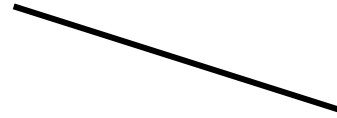
```
program clear;  
uses GraphABC;  
begin  
ClearWindow;  
ClearWindow  
(clMoneyGreen);  
end.
```

# Графические примитивы

1. Точка



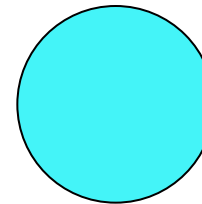
2. Линия



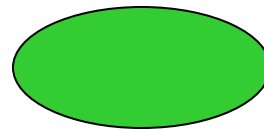
3. Прямоугольник



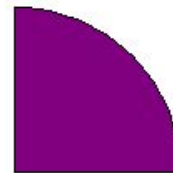
4. Окружность



5. Эллипс



6. Сектор



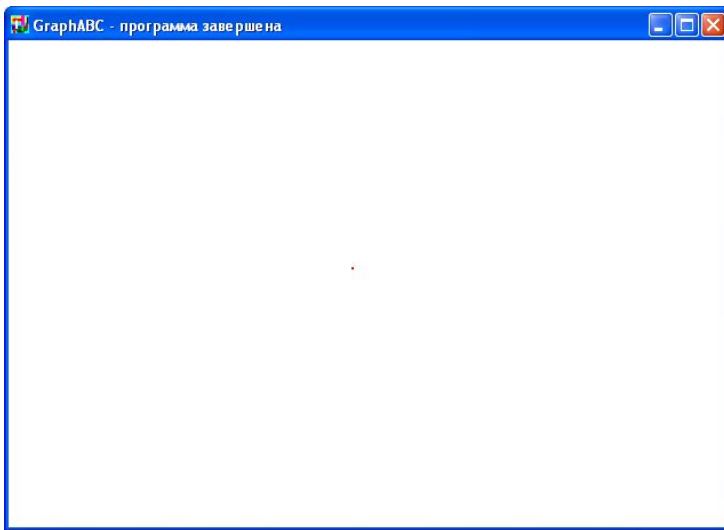
7. Дуга





# Точка

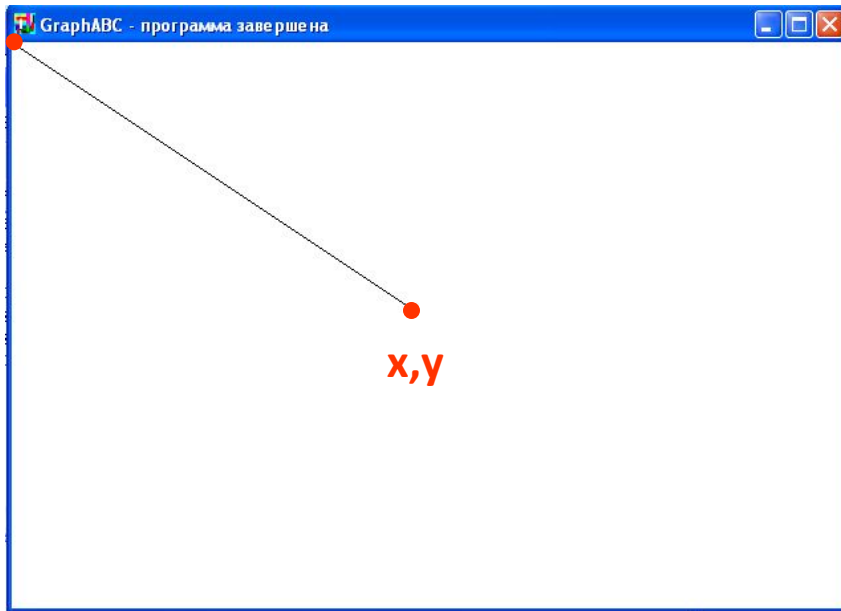
**SetPixel(x,y,color)** - Закрашивает один пиксел с координатами (x,y) цветом color



```
program точка;  
uses GraphABC;  
begin  
    SetPixel(300,200,clred);  
end.
```

# Линии

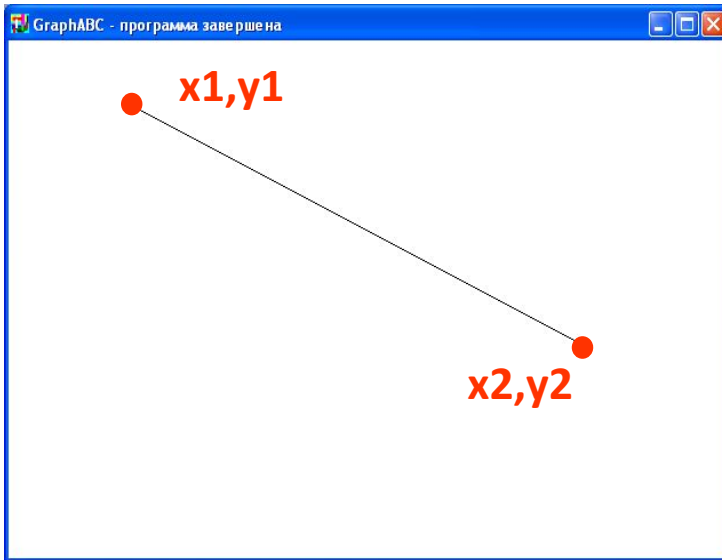
**LineTo(x,y)** - рисует отрезок от текущего положения пера до точки (x,y); координаты пера при этом также становятся равными (x,y).



```
Program liniay;  
uses GraphABC;  
begin  
LineTo(300,200);  
end.
```

# Линии

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



Program liniay;  
uses GraphABC;  
begin  
**line(100,50,500,250);**  
end.

# Используемые цвета

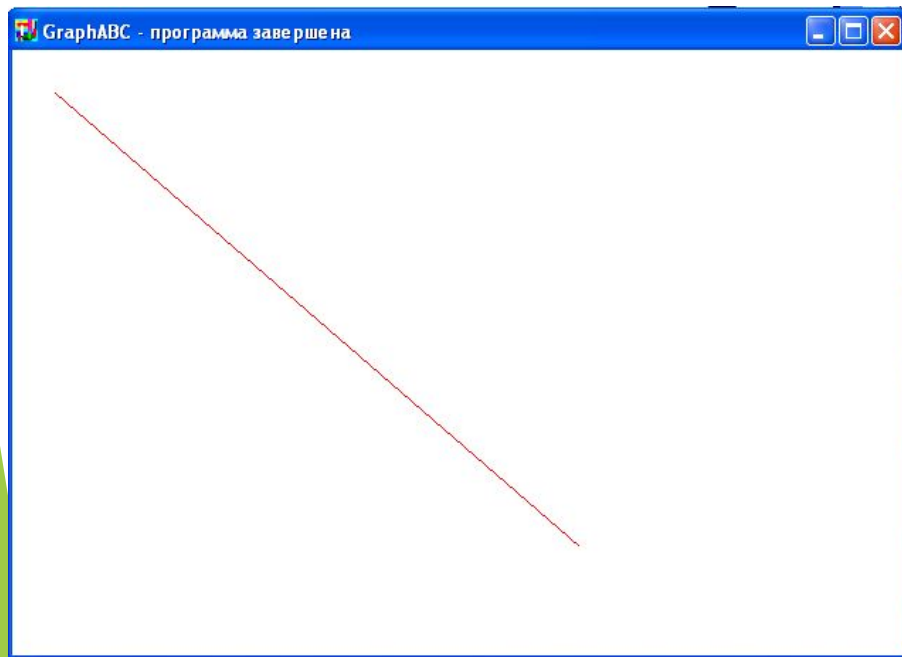
**clBlack** – черный  
**clPurple** – фиолетовый  
**clWhite** – белый  
**clMaroon** – темно-красный  
**clRed** – красный  
**clNavy** – темно-синий  
**clGreen** – зеленый  
**clBrown** – коричневый  
**clBlue** – синий  
**clSkyBlue** – голубой  
**clYellow** – желтый  
**clCream** – кремовый

**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  
**clMedGray** – серый  
**clSilver** – серебряный

**Random(16777215)** – случайный цвет  
из всей палитры цветов Паскаля

# Цвет линии

**SetPenColor(color)** - устанавливает цвет пера, задаваемый параметром **color**.



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

# Пунктирная линия

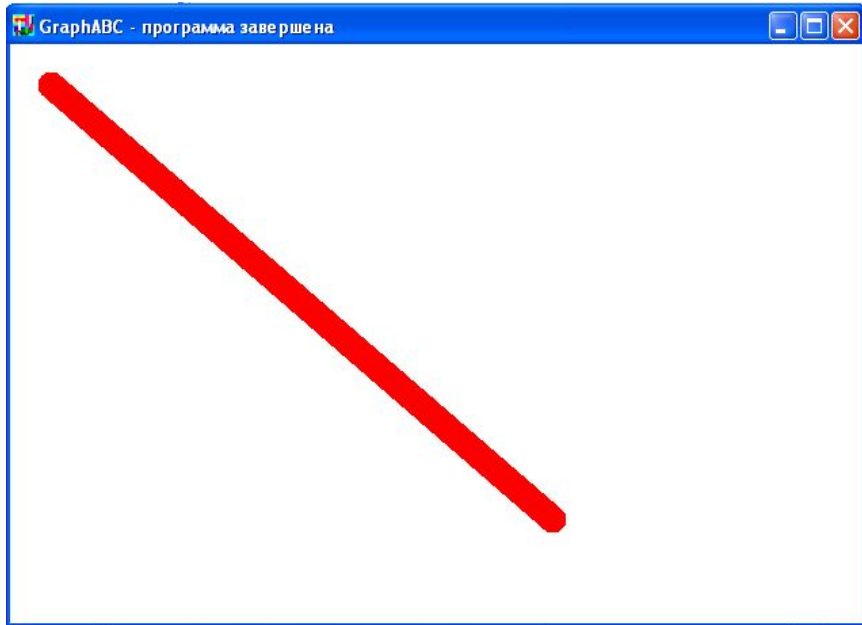
**SetPenStyle(<номер от 1 до 6>);** -  
устанавливает стиль пера, задаваемый номером.



```
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.
```

# Толщина линии

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

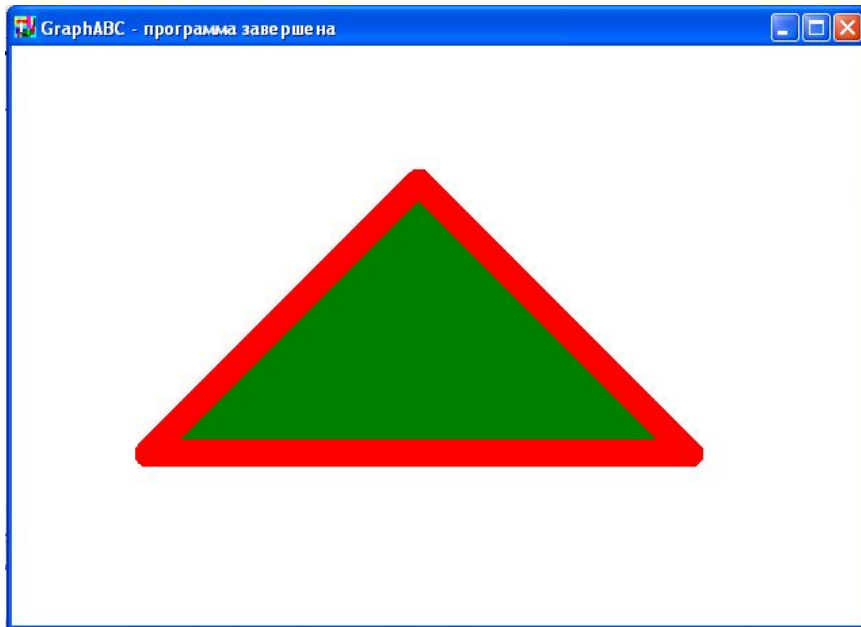


```
Program liniay;  
uses GraphABC;  
begin  
    setpenwidth(20);  
    setpencolor(clred);  
    line(30,30,400,350);  
end.
```

# Треугольник

Рисуется процедурами

**Line(x1,y1,x2,y2); LineTo(x,y);**

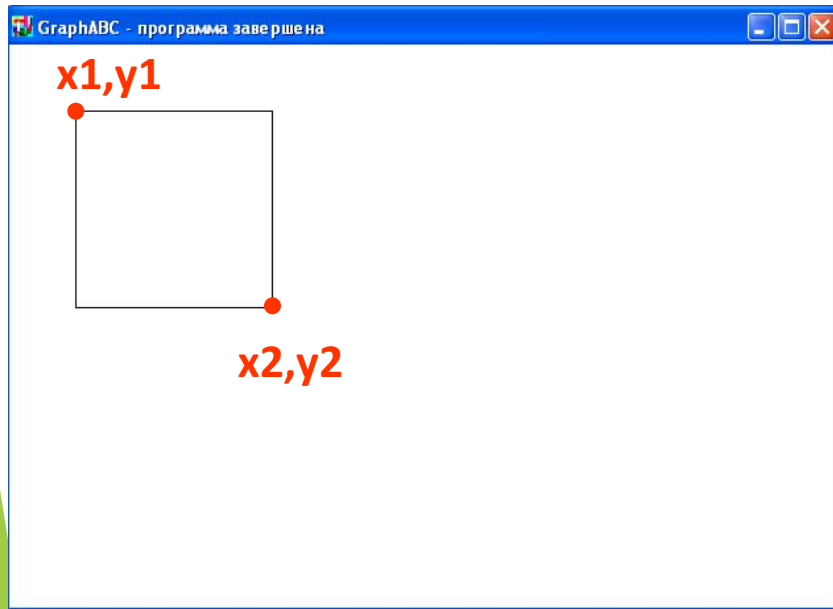


```
Program treugolnik;  
uses GraphABC;  
begin  
  setpenwidth(20);  
  setpencolor(clred);  
  line(300,100,500,300);  
  lineto(100,300);  
  lineto(300,100);  
  floodfill(300,200,clgreen);  
end.
```



# Прямоугольник

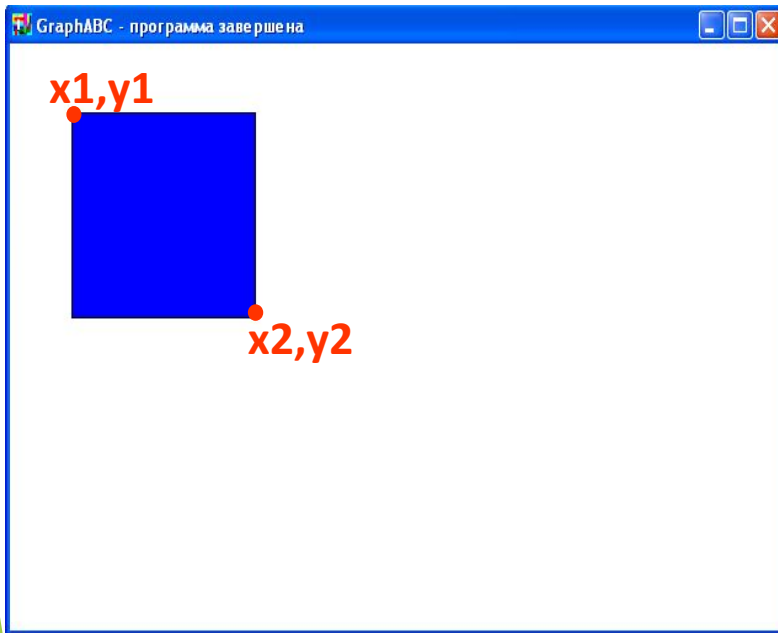
**Rectangle(x1,y1,x2,y2)** - рисует  
прямоугольник, заданный координатами  
противоположных вершин (x1,y1) и (x2,y2).



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

# Заливка цветом

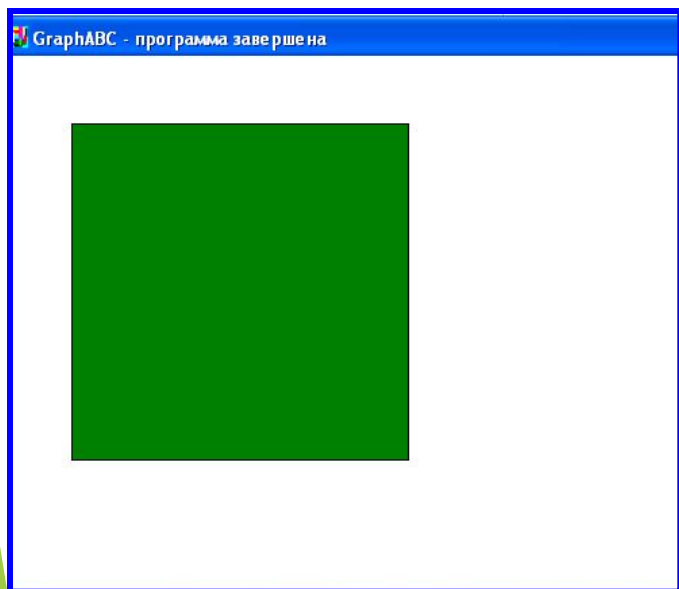
**FloodFill(x,y,color)** - заливает область одного цвета цветом color, начиная с точки (x,y).



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

# Заливка кистью

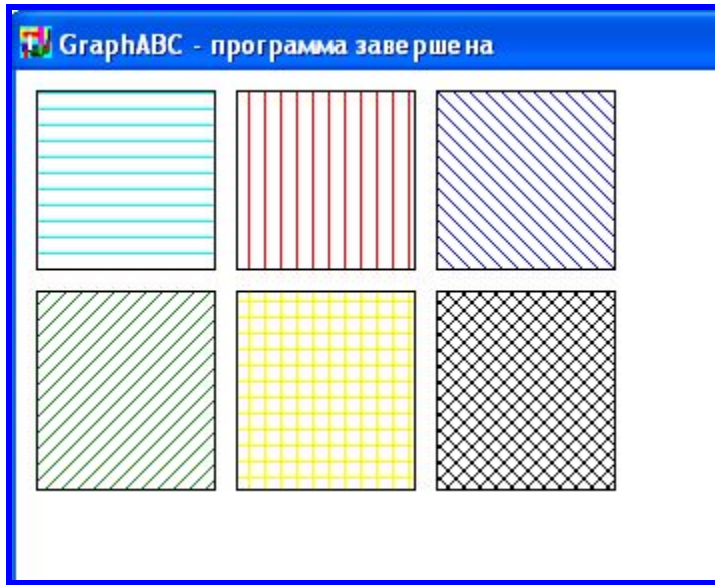
**SetBrushColor(color)** - устанавливает цвет кисти. Заливка кистью распространяется на замкнутый контур, описание которого следует за процедурой установки цвета кисти.



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

# Заливка кистью

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



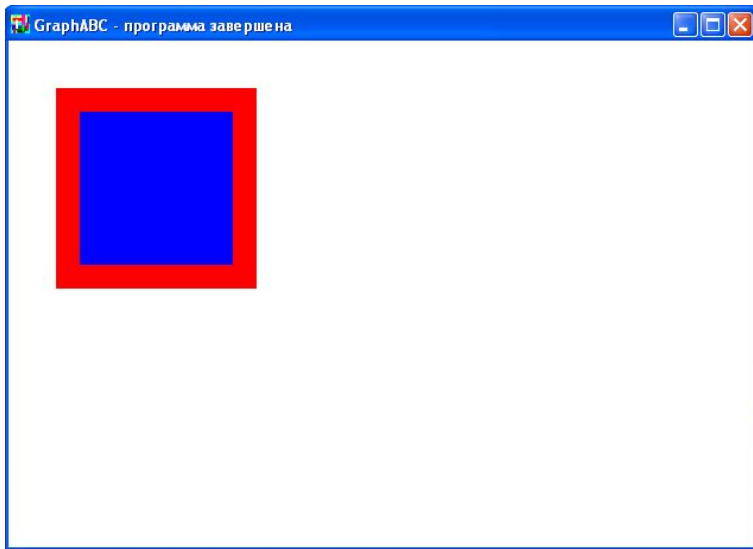
**По умолчанию задается стиль 0 – сплошная заливка цветом.**

```
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.
```

# Цвет и толщина контура

Задаются процедурами

**SetPenWidth(w); SetPenColor(color);**



```
Program pryamougolnik;  
uses GraphABC;  
begin
```

```
SetPenColor(clRed);
```

```
SetPenWidth(20);
```

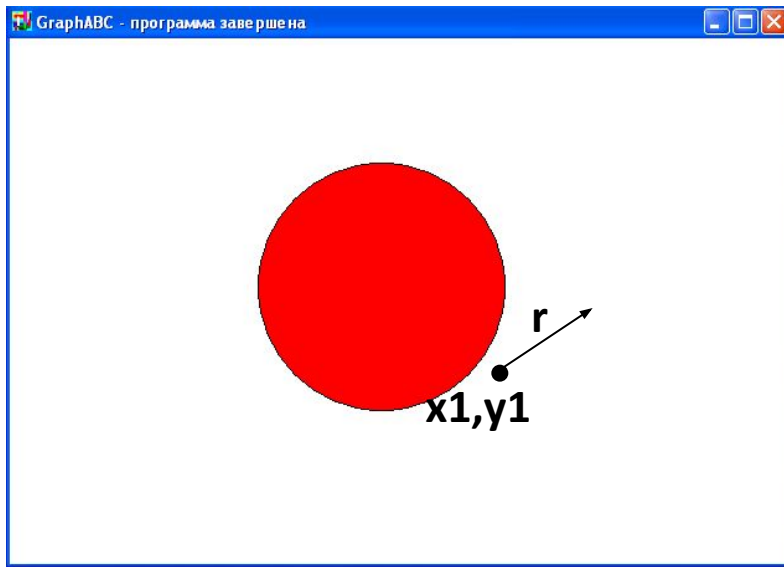
```
Rectangle(50,50,200,200);
```

```
FloodFill(100,100,clBlue);
```

```
end.
```

# Окружность

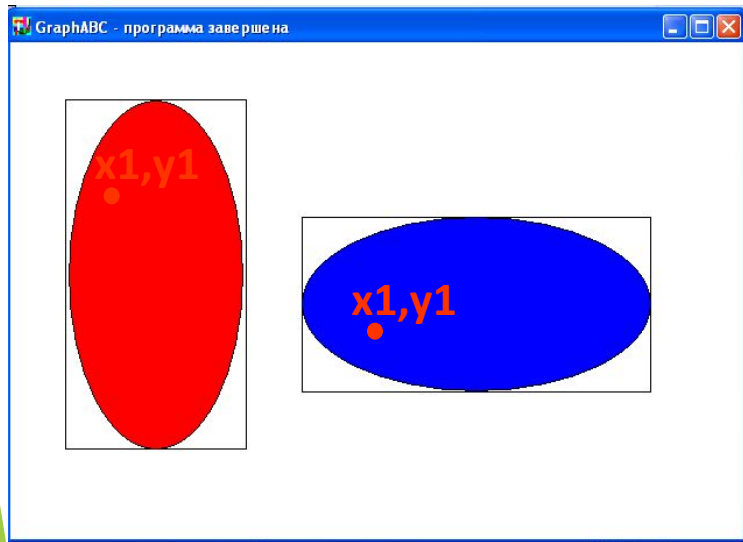
**Circle(x,y,r)** - рисует окружность с центром в точке (x,y) и радиусом r.



```
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).



x2,y2

x2,y2

Program oval;  
uses GraphABC;

Begin

**Ellipse(50,50,200,350);**

FloodFill(50+100,50+100,clred);

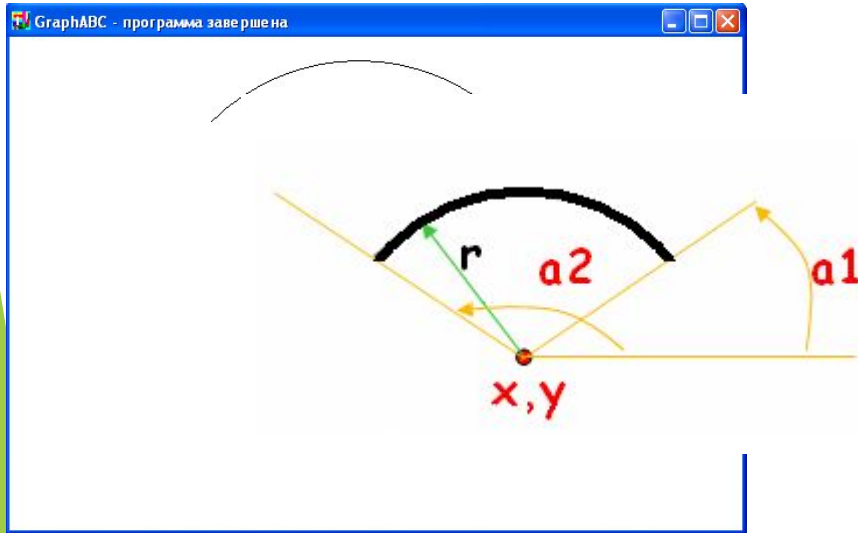
**Ellipse(250,150,550,300);**

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).

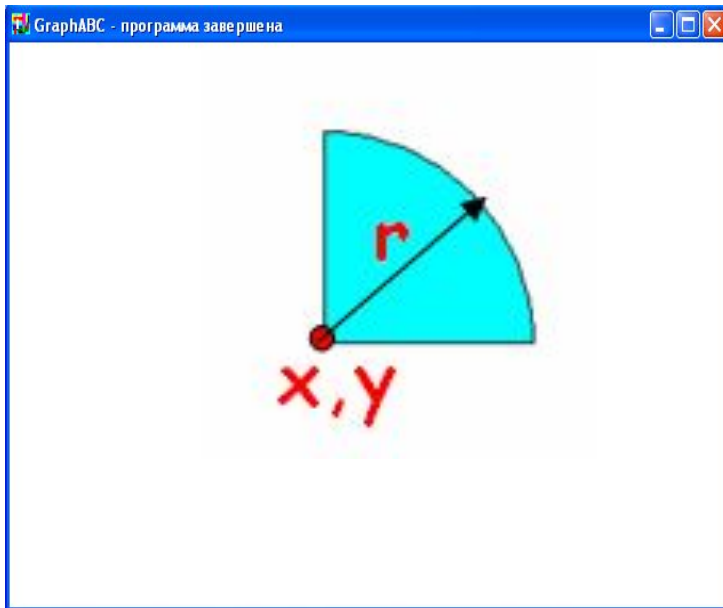


```
Program дуга;  
uses GraphABC;  
Begin  
SetPenWidth(10);  
Arc(300,250,150,45,135);  
end.
```



# 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,c1  
Aqua);  
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.
```

# 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));
```

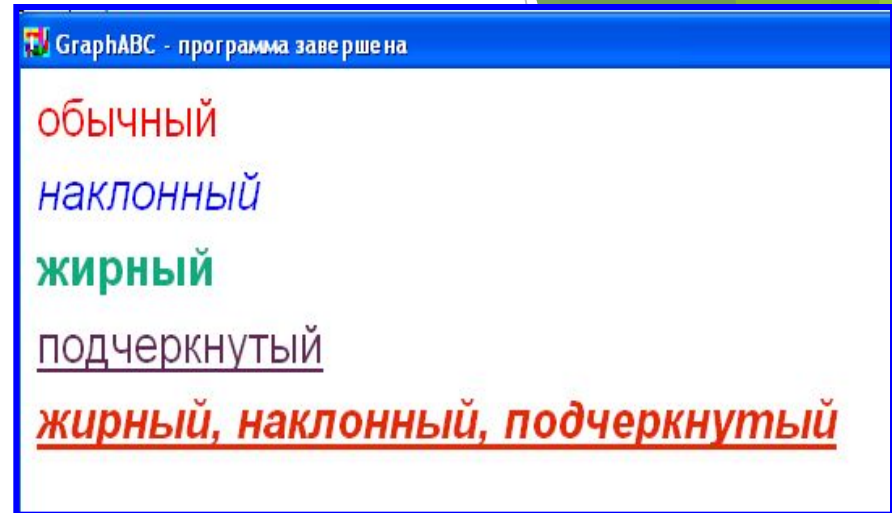
```
TextOut(10,130,'подчеркнутый');
```

```
SetFontStyle(fsBoldItalicUnderline);
```

```
SetFontColor(Random(16777215));
```

```
TextOut(10,170,'жирный, наклонный, подчеркнутый');
```

end.



# Используемые цвета

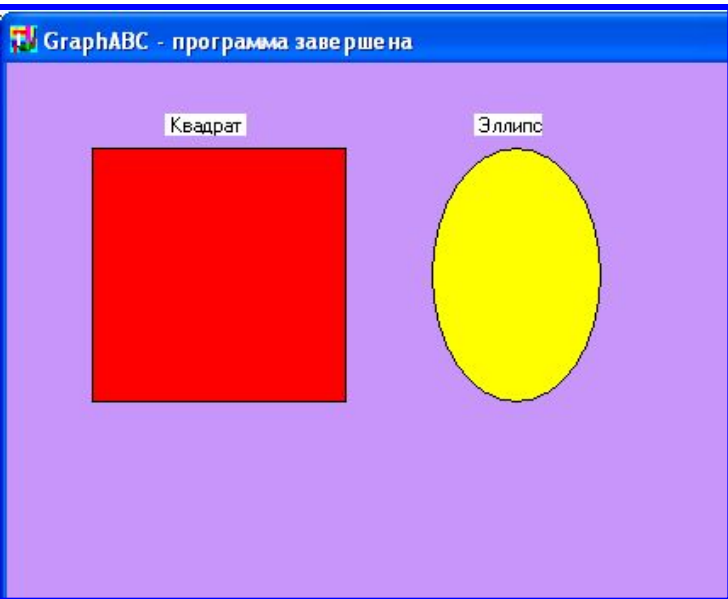
Цвет можно задавать и с помощью функции **RGB(r,g,b)** где r, g и b – целые числа в диапазоне от 0 до 255.

Функция возвращает целое значение, являющееся кодом цвета, который содержит красную, зеленую и синюю составляющие с интенсивностями r, g и b соответственно (0 соответствует минимальной интенсивности, 255 – максимальной).

**RGB(255,255,255)** – соответствует белому цвету.

**RGB(0,0,0)** – соответствует черному цвету.

# Например,



```
Program color;  
uses GraphABC;  
begin
```

```
Clearwindow(rgb(200,150,250));
```

```
TextOut(93,30,' Квадрат ');
```

```
Rectangle(50,50,200,200);
```

```
FloodFill(55,55,clRed);
```

```
TextOut(275,30,' Эллипс');
```

```
Ellipse(250,50,350,200);
```

```
FloodFill(250+50,50+50,clYellow);
```

```
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;  
  hidecursor; {скрывает текстовый  
курсor}  
  gotoXY(12,3); write('Квадрат');  
  Rectangle(50,50,200,200);  
  FloodFill(55,55,clYellow);  
end.
```

# Загрузка готового рисунка

LoadPicture(fname)

**n:=LoadPicture(fname) –**

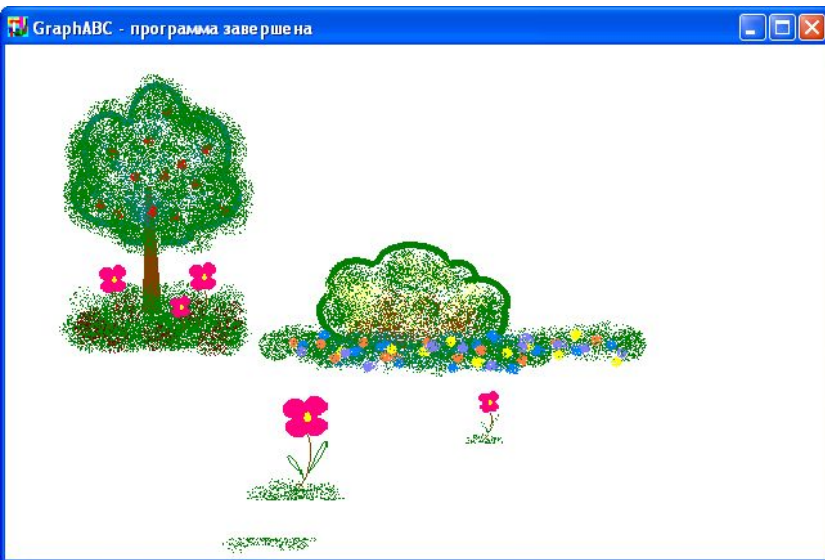
загружает рисунок из файла с именем fname в оперативную память и возвращает описатель рисунка в целую переменную n; если файл не найден, то возникает ошибка времени выполнения.

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

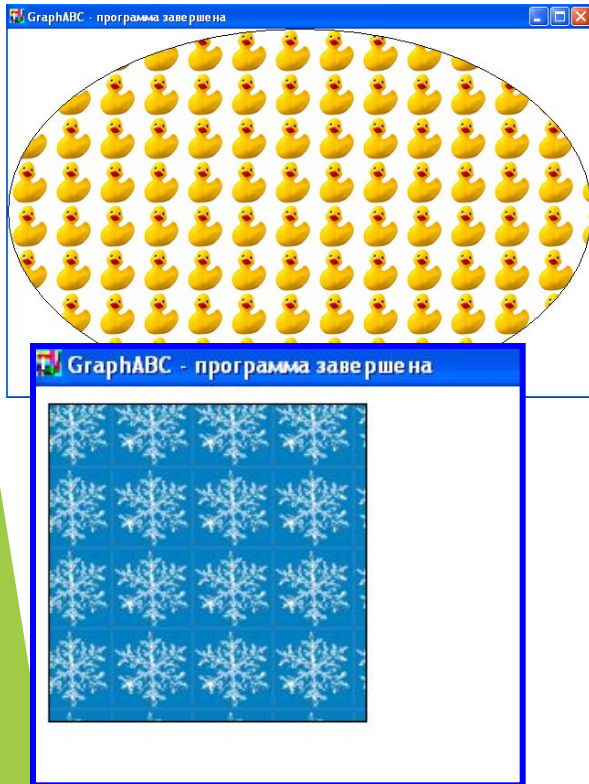
**SavePicture(n, 'fname')** -

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

# Заливка кистью

`SetBrushPicture('fname')` -

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



uses GraphABC;

begin

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

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

end.