Графика в Бейсике

Содержание

Введение

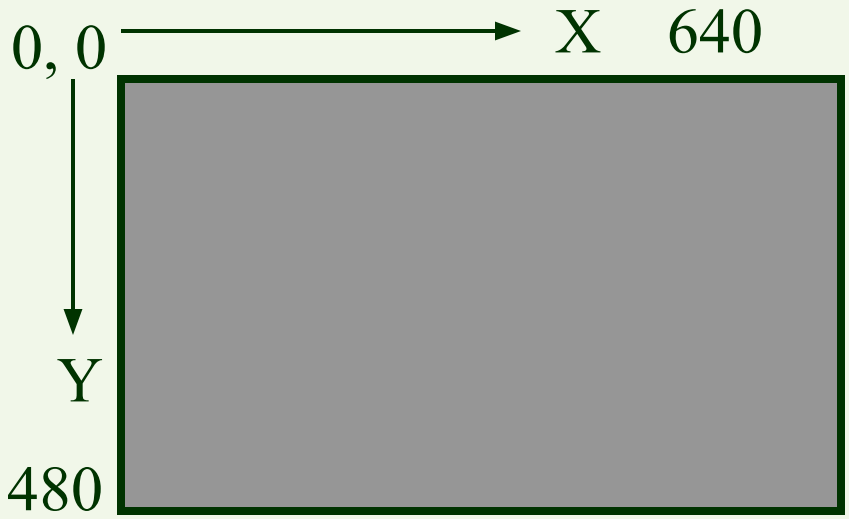
При входе в оболочку Бейсика по умолчанию включается текстовый режим, в котором можно производить вычисления и выводить результаты на экран, но если мы хотим пользоваться графическими возможностями языка, нам надо объяснить это компьютеру посредством включения графического режима командой **SCREEN 12**.

Графических режимов несколько, но именно режим № 12 обладает наибольшей разрешающей способностью и позволяет получать качественные графические объекты.

После включения графического режима мы можем задавать компьютеру команды рисования **графических примитив.**

Режим SCREEN 12

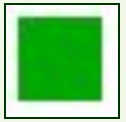
В режиме SCREEN 12 экран представляет собой координатную сетку с началом в верхнем углу, вправо от которого увеличивается координата X, а вниз - координата Y. Максимальное значение X на экране 640, а Y - 480.



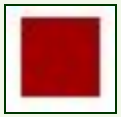
Основные цвета

0 - чёрный 

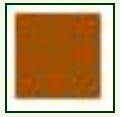
1 - синий 

2 - зелёный 

3 - голубой 

4 - красный 

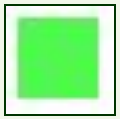
5 - фиолетовый 

6 - коричневый 

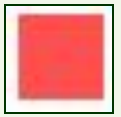
7 - светло-серый 

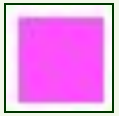
8 - тёмно-серый 

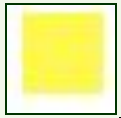
9 - ярко-синий 

10 - ярко-зелёный 

11 - ярко-голубой 

12 - ярко-красный 

13 - ярко-фиолетовый 

14 - жёлтый 

15 - белый 

Задание цвета

COLOR C

где C – цвет, которым далее будет окрашен текст или графическая фигура.

Пример программы:

```
10 SCREEN 12
20 Color 5
30 ? ” Меня зовут Аксинья”
```

Вывод текстовой информации в графике

Текстовая информация в графическом режиме выводится с помощью операторов LOCATE и PRINT.

При этом нужно помнить, что координаты в операторе LOCATE не графические (640x480), а текстовые (80x25).

LOCATE Y, X: ? ”Текст”

где X – номер позиции первой буквы текста;
Y – номер строки, с которой будет выводиться текст.

Пример программы:

```
10 SCREEN 12
20 Color 5
30 LOCATE y,x: ? ” Текст”
```

Графические примитивы

- Точка
- Отрезок прямой линии
- Прямоугольник со сторонами, параллельными экрану
- Закрашенный прямоугольник
- Окружность
- Эллипс
- Дуги окружностей
- Дуги эллипсов

Точка

PSET (X, Y), C

где X и Y - координаты точки на экране, а C - её цвет. Если цвет не указан, то точка будет изображена последним установленным цветом.

Пример программы:

```
10 SCREEN 12
20 PSET (320, 240), 4
```

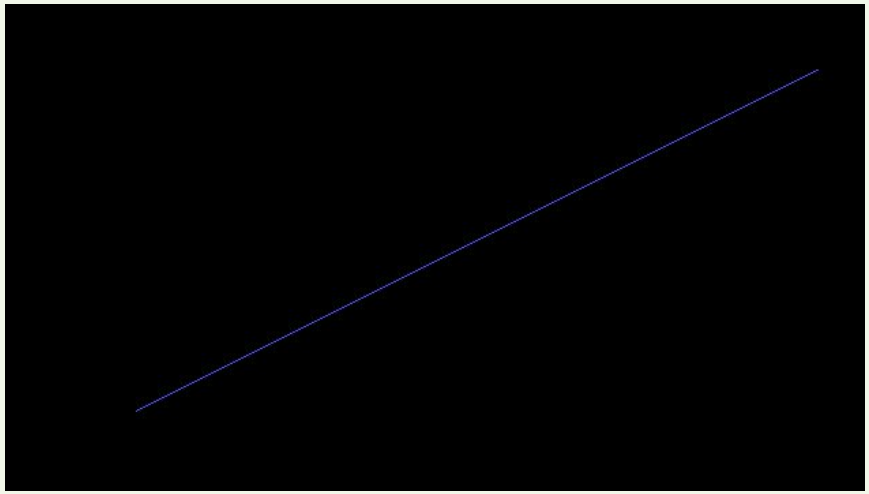
Отрезок прямой линии

LINE (X1, Y1) - (X2, Y2), C

где X1, Y1 - координаты начала отрезка;

X2, Y2 - координаты конца отрезка;

C - цвет.



Пример программы:

```
10 SCREEN 12
20 LINE (100, 300) - (600, 50), 9
```

Прямоугольник со сторонами, параллельными экрану

LINE (X1, Y1) - (X2, Y2), C, B

где X1, Y1 - координаты начала диагонали прямоугольника;

X2, Y2 - координаты конца диагонали прямоугольника;

C – цвет;

B (от английского BOX - коробка, ящик) - показатель того, что мы рисуем прямоугольник.

Пример программы:

```
10 SCREEN 12
20 LINE (100, 300) - (600, 50), 11, B
```

Закрашенный прямоугольник

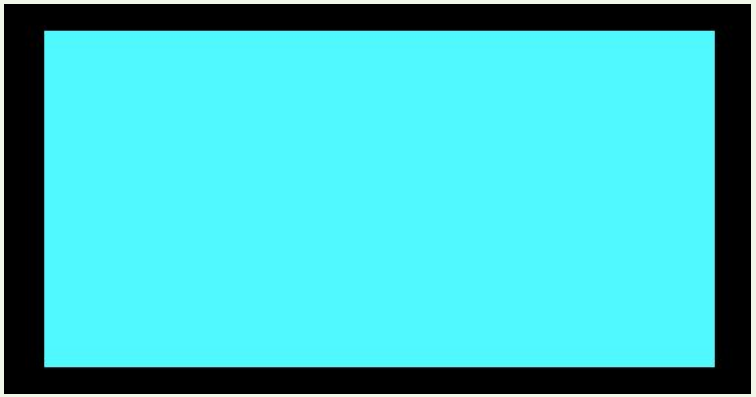
LINE (X1, Y1) - (X2, Y2), C, BF

где X1, Y1 - координаты начала диагонали прямоугольника;

X2, Y2 - координаты конца диагонали прямоугольника;

C – цвет;

BF (от английского BOX FULL - полная коробка) - показатель того, что мы рисуем закрашенный прямоугольник.



Пример программы:

```
10 SCREEN 12
20 LINE (100, 300) - (600, 50), 11, BF
```

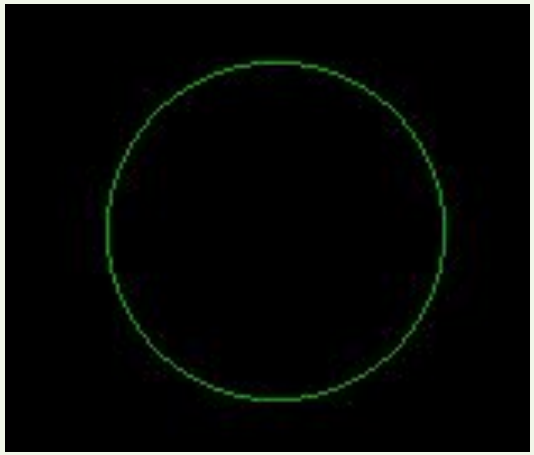
Окружность

CIRCLE (X, Y), R, C

где X, Y - координаты центра;

R - радиус (в экранных точках);

C - цвет.



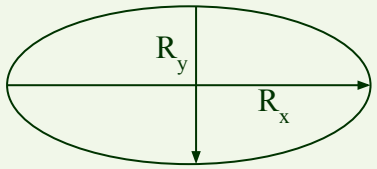
Пример программы:

```
10 SCREEN 12
20 CIRCLE (320, 175), 50, 2
```

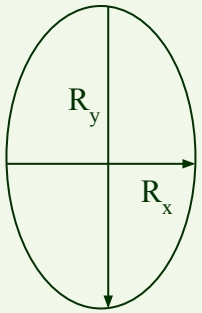
Эллипс

Эллипс - это окружность, сжатая по вертикали или горизонтали, у которой имеется два радиуса - по осям X и Y. Частное от деления R_y на R_x дает нам ***коэффициент сжатия:***

$$K = \frac{R_y}{R_x}$$



0<K<1



K>1

Таким образом, для эллипсов:

- сжатых по вертикали, коэффициент сжатия будет в пределах от 0 до 1;
- для эллипсов сжатых по горизонтали – **К > 1**;
- если же **K= 1**, то это окружность.

Эллипс

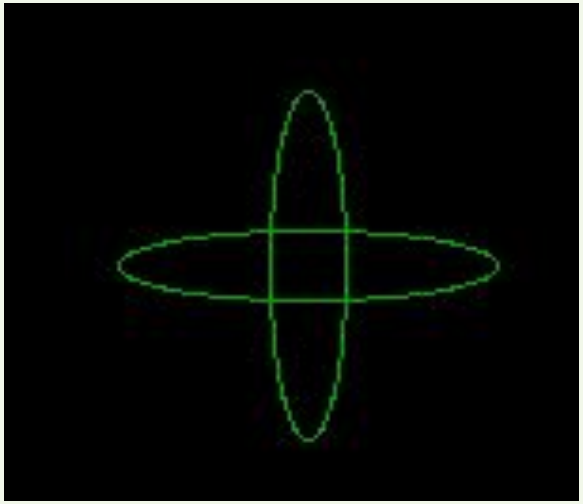
CIRCLE (X, Y), R, C, , , K

где X, Y - координаты центра эллипса;

R - радиус той окружности, из которой этот эллипс получился;

C - цвет;

K - численное значение коэффициента сжатия.



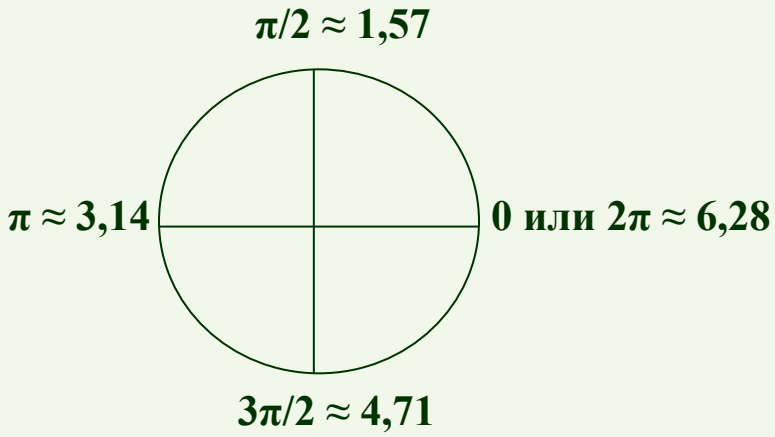
Пример программы:

```
10 SCREEN 12
20 CIRCLE (320, 175), 50, 2, , , 5
30 CIRCLE (320, 175), 50, 2, , , 0.2
```

Дуги окружностей

CIRCLE (X, Y), R, C, a, b

дуга строится от угла a к углу b против часовой стрелки.



Бейсик позволяет использовать в качестве операторов в своих командах арифметические выражения. Поэтому, если вы знаете угол начала дуги, например 30 градусов, но затрудняетесь определить его на тригонометрической окружности, то можете написать по формуле: **π * λ/180**

3.14*30/180

Дуги окружностей

Если в операторе построения дуг поставить знак минус перед значениями углов дуги, то автоматически будут проведены радиусы, соединяющие центр окружности с концами дуг.

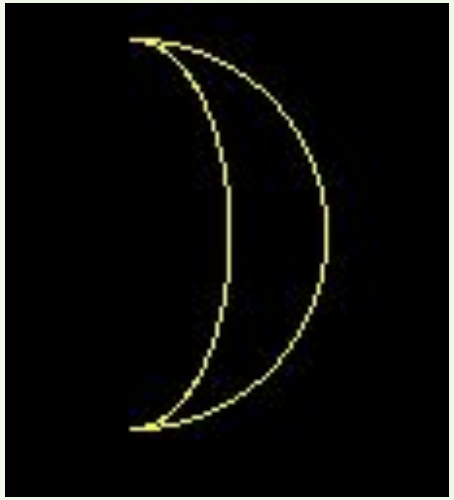
Пример программы:

```
10 SCREEN 12
20 CIRCLE (200, 120), 50, 15, -45*3.14/180, -315*3.14/180
```

Дуги эллипсов

CIRCLE (X, Y), R, C, a, b, K

В качестве примера нарисуем «светит месяц, светит ясный».



Пример программы:

```
10 SCREEN 12
20 CIRCLE (500, 70), 50, 14, 4.71, 1.57
30 CIRCLE (500, 70), 50, 14, 4.71, 1.57, 2
```

дуга окружности

дуга эллипса

Правила закраски

PAINT (X, Y), C1, C2

где X, Y - координаты любой (!) точки внутри закрашиваемого контура;

C1 - цвет, которым закрашивается контур;

C2 - цвет самого контура.

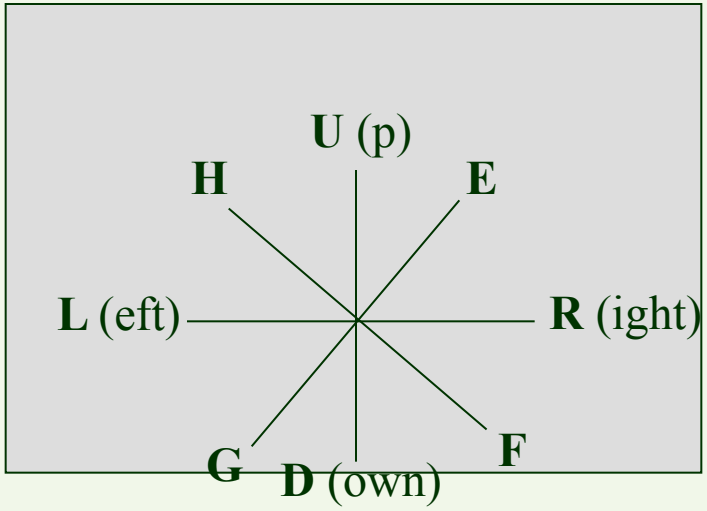
Если цвета совпадают, то достаточно указать один.

Правила закраски:

- контур должен быть ***замкнут***;
- контур должен быть ***одноцветен***;
- координаты точки закраски должны лежать ***внутри*** контура.

Макроязык GML

Для расширения возможностей машинной графики Бейсика был дополнительно разработан специальный макроязык GML (Graphics Macro Language). Он позволяет строить довольно сложные изображения и быстро выводить их на экран. Каждая команда языка представляет собой латинскую букву, после которой следует один или два числовых параметра (как правило целые числа).



Основные команды перемещения языка GML

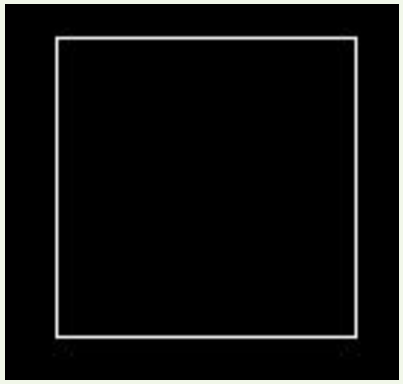
После команды указывается количество точек перемещения.

Макроязык GML

Команда	Действие
M x, y	Переместиться в точку с координатами x и y
M ±n,±m	Переместиться по отношению к текущей позиции на ±*n* точек по оси *X* и на ±*m* точек по оси *Y*. Знаки плюс и минус проставлять обязательно
At	Поворот изображения против часовой стрелки вокруг точки, с которой начиналось рисование, на 90º * *t* (*t* может принимать значения 0, 1, 2,3). Действует во всех дальнейших командах до нового назначения
Cn	Задание нового цвета. Действует во всех дальнейших командах до нового назначения
Sn	Расстояние, указанное в командах перемещения, умножается на *n / 4 (0≤ n ≤255)*
B	Переместиться на новую позицию при помощи следующих *B* команд, но рисования не производить. Отменяется установкой цвета *C*
N	Выполнить следующую команду перемещения и вернуться в исходную позицию; может предшествовать любой команде перемещения
Pc1, c2	Команда заполнения контура цветом: c1 – цвет заполнения, c2 – цвет контура

Макроязык GML

Для приведения в действие последовательности команд языка GML необходимо использование оператора **DRAW.**



Пример программы:

10 SCREEN 12

20 DRAW “R50 D50 L50 U50”

Программа изобразит квадрат, начиная от последней графической точки. Если такой точки не было, и вы сразу начинаете строить изображение с помощью оператора DRAW, то по умолчанию исходной точкой считается центр экрана.

Тема закончена

До свидания!

Выход