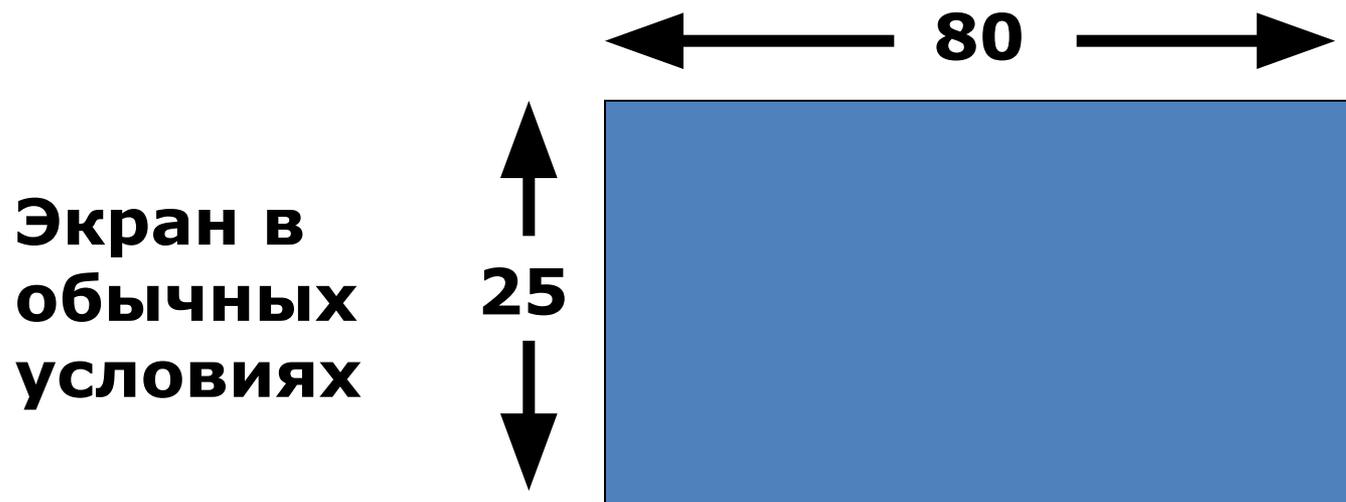


Графические возможности Turbo Pascal

- У компьютерного монитора два режима работы - **текстовый** и **графический**.
- В текстовом режиме минимальным объектом, отображаемым на экране, является алфавитно-цифровой или какой-либо иной **символ**.

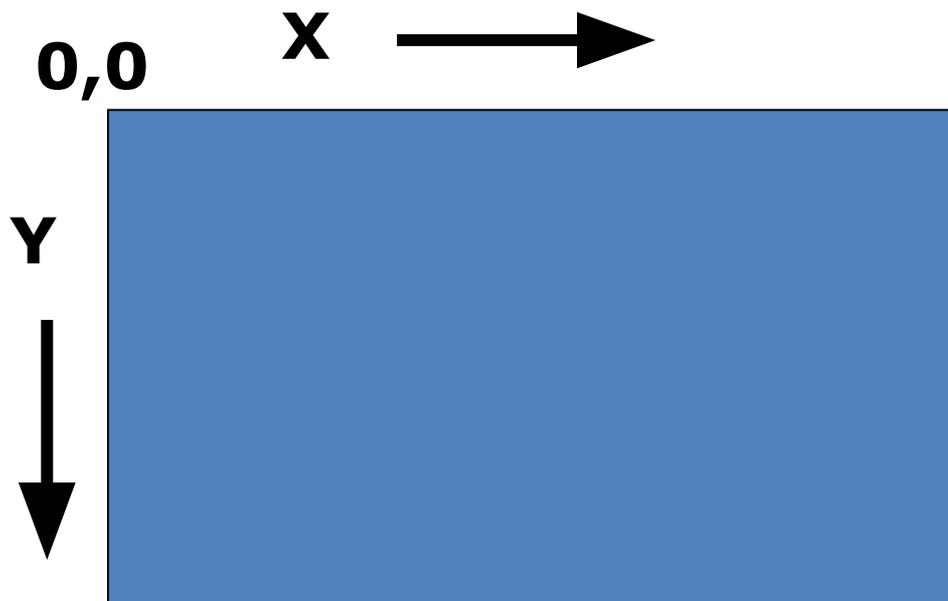


- В графическом режиме минимальным объектом, выводом которого может управлять программист, является **пиксель – графическая точка**.
- **Пиксель** имеет меньшие размеры по сравнению с символом, они определяются **разрешением монитора**.

- Графические координаты задают положение точки на экране дисплея.
- В качестве графических координат используется **порядковые номера пикселей**.

- **Точкой отсчёта** является верхний левый угол экрана. Значения x – координаты отсчитываются слева направо, а y – координаты – сверху вниз.

Экран в
графическом
режиме



- Для правильного отображения рисунков на экране необходимо учесть **различия между декартовой и графической системами координат:**

1. Графические координаты принимают только целочисленные значения;
2. Графические координаты принимают значения, ограниченные как **снизу(нулевым значением)**, так и сверху (**значением разрешения**);
3. Графическая координата у отсчитывается сверху вниз.

- Чтобы работа в графическом режиме была возможна, этот режим должен поддерживаться **видеоадаптером.**
- **Turbo Pascal** обеспечивает работу со следующими видеоадаптерами:
CGA, MCGA, EGA, VGA, Hercules, AT&T400, 3270 PC, IBM-8514.
- Видеоадаптером управляет специальная программа, которая называется **драйвером.**

- **Драйвер хранится** в отдельном файле на диске и содержит как исполняемый код, так и необходимые ему для работы данные.
- Файл с драйвером имеет расширение **.bgi**. Имя файла с драйвером соответствует типу видеоадаптера компьютера.
- Большинство видеоадаптеров могут работать в нескольких графических режимах. Эти режимы различаются разрешением и

Загрузка графического режима

- В пакет **Turbo Pascal** входит модуль **Graph**, который содержит процедуры, функции, а также встроенные типы и константы, предназначенные для работы в графическом режиме.
- Чтобы воспользоваться возможностями модуля **Graph**, в начале программы необходимо разместить оператор:
uses Graph;

Инициализация графического режима и выход из него

- Переключение в графический режим работы дисплея выполняется вызовом процедуры:

InitGraph (gd, gm, 'c:\tp\bgi');

Тип
адаптера

Видеорежим

Строка с
указанием
расположения
драйвера на
диске

- Для большинства современных видеоадаптеров можно использовать драйвер **egavga.bgi**.
- Пустая строка означает, что графический драйвер находится в том же каталоге, что и программа.
- **Инициализация графического режима** обычно сопровождается обработкой возможных ошибок инициализации с помощью функции **GraphResult** .

- Эти ошибки могут быть связаны с отсутствием графического драйвера или неправильными значениями параметров.
- При наличии ошибок функция **GraphResult** возвращает отличный от нуля результат - **код ошибки**.
- Завершение работы в графическом режиме производится с помощью процедуры **CloseGraph**, которая выгружает драйвер из памяти и восстанавливает предыдущий видеорежим.

- Тип видеоадаптера может быть задан путём присваивания соответствующего значения переменной **gd**.
- При автоматическом распознавании видеоадаптера в правой части оператора присваивания указывается встроенная константа **Detect (она имеет нулевое значение)**:
gd:= Detect;

- Чтобы задать определённый графический режим, следует присвоить значение переменной **gm**.
- По умолчанию **gm** равно 0.

Пример программы для инициализации графического режима и выхода из него

Program p1;

Uses graph;

Var gd, gm, Err : integer;

Begin

gd:=Detect;

initGraph (gd, gm, '');

Err:=GraphResult;

If Err=grOk Then

Begin

{графика}

Setbkcolor(6);

Setcolor(4);

Line (10, 50, 600, 300);

setFillStyle(1,5); {НОВЫЙ СТИЛЬ}

bar(100,100,200,200);

Setcolor(3);

circle(300,300,100);

Setcolor(13);

Circle (300,300,50);

Readln;

CloseGraph;

End

**Else WriteLn('Ошибка инициализации
графики:', GraphErrorMsg(Err));**

End.

Некоторые функции

- Функция GraphResult. Возвращает значение типа `integer`, в котором закодирован результат последнего обращения к графическим процедурам. Если ошибка не обнаружена, значением функции будет ноль, в противном случае – отрицательное число.
- Значению 0 соответствует встроенная константа `grOk`.

- Чаще всего причиной возникновения ошибки при обращении к процедуре `InitGraph`, является неправильное указание местоположения файла с драйвером графического адаптера (например, файла `CGA.BGI` для адаптера CGA).

- Для упрощения повторения примеров скопируйте файл, соответствующий адаптеру Вашему ПК, в текущий каталог.

- Процедура CloseGraph. Завершает работу адаптера в графическом режиме и останавливает текстовый режим работы экрана.

- Процедура RestoreCRTMode. Служит для кратковременного возврата в текстовый режим.
- В отличие от процедуры CloseGraph не сбрасываются установленные параметры графического режима и не освобождается память, выделенная для размещения графического драйвера.

- **Функция GetGraphMode.** Возвращает значение типа *Integer*, в котором содержится код установленного режима работы графического адаптера.
- **Процедура SetGraphMode.** Устанавливает новый графический режим работы адаптера.