

## Модификация движения шарика, массивы.

### Массивы

Массив - это группа пронумерованных переменных одного типа. Массивы бывают одномерные и многомерные (двумерные и т.д.). Для начала остановимся на одномерных массивах: это множество переменных, пронумерованное одним индексом, то есть по сути - строка пронумерованных переменных. Каждая переменная называется *элементом* массива.

Массивы, как и переменные, создаются командой **var**, после которой пишутся имена массивов, двоеточие, ключевое слово **array**, размер массива и тип элементов массива. Например приведённые ниже команды создают несколько переменных, 3 массива целых чисел одинакового размера (1000 элементов), массив вещественных чисел (100 элементов) и массив строк (500 элементов):

```
var i, j, k: integer;
    arr, b, massiv: array [1..1000] of integer;
    x, y: real;
    str: array [1..500] of string;
    r: array [1..100] of real;
```

С каждым элементом массива можно работать как обычной переменной, имя которой состоит из названия массива и его номера в нём, взятого в квадратные скобки, например: **arr[312]** или **arr[k]**, где **k** - целочисленная переменная, значение которой должно быть от 1 до 1000. Если при обращении к **k**-му элементу массива **k** выйдет из этого диапазона, то получится так называемый выход за пределы массива, который приведёт к сбоям в работе программы.

Двумерные массивы отличаются тем, что их элементы пронумерованы не одним, а двумя индексами. Это можно представлять, как таблицу с числами, в которой первый индекс определяет номер строки, а второй - номер столбца, на пересечении которых и лежит элемент. При этом можно считать и наоборот, что первый индекс нумерует столбцы, а второй - строки, но важно в пределах одной программы придерживаться одного из этих вариантов, не меняя индексы местами. Я буду стараться придерживаться первого варианта (хотя например в программе, написанной на занятии со второй полугруппой - был второй вариант нумерации).

Для создания двумерного массива надо в квадратных скобках через запятую указать пределы, в которых могут лежать значения соответствующих индексов. Например здесь создаётся два массива вещественных чисел  $30 \times 40$  и массив целых чисел  $100 \times 100$ :

```
var r1, r2: array [1..30, 1..40] of real;
    a: array [1..100, 1..100] of integer;
```

Для обращения к элементу массива надо в квадратных скобках указать через запятую его индексы в нужном порядке: **a[10, 20]** - элемент в 10-ой строке и 20-ом столбце, **a[i, j]** - элемент в *i*-ой строке и *j*-ом столбце. При обращении по переменным, как и в одномерном случае, нужно не выйти за пределы массивов. Кроме того можно перечислять индексы не через запятую, а в отдельных квадратных скобках: **a[i][j]**.

Массивы больших размерностей объявляются и используются аналогично (добавление индексов через запятую). Трёхмерный массив можно представлять как числа в кубике. Массивы больших размерностей уже нельзя представить геометрически (для этого надо представлять как выглядит четырёхмерное (и более) пространство, однако их смысл в том, что каждый их элемент характеризуется более чем тремя индексами).

Так например строка книги - одномерный массив букв, и каждая буква имеет один номер - её положение в строке. Страница книги - двумерный массив букв, добавляется ещё один индекс - номер строки. Вся книга - трёхмерный массив, третий индекс - номер страницы. Ряд книг стоящих на полке - четырёхмерный массив, с четвёртым индексом - номером книги на полке. Шкаф книг - пятимерный массив, пятый индекс - номер полки. Библиотека - шестимерный массив. Ну и так далее.

## Изменение движения

Для дальнейшего написания удобно работать с вещественными, а не целыми скоростями. Для этого нужно переменные **x**, **y**, **vx** и **vy** сделать типа **real**, изменять **x** и **y** так же как и раньше, но перед рисованием (и закрашиванием) - округлять с помощью функции **round(x)**. Скорости при этом можно определять так же, как и раньше, однако с учётом дальнейшего развития программы удобнее сделать по-другому. Для этого надо использовать чуть-чуть тригонометрии, которой вы пока что не знаете, однако если вас это не слишком сильно пугает - попробуйте. Раньше скорость шарика определялась проекциями на оси *X* и *Y*, однако можно определить её другими двумя параметрами - её величиной **v** и направлением. Направление будем задавать углом  $\varphi$  от оси *X* против часовой стрелки. Введение вещественных скоростей позволит легче рассчитывать столкновение с кубиком, а введение угла и величины скорости позволяет реализовать возможность задавать фиксированную скорость, определяющую сложность игры, но сохранить случайное направление.

Тогда если введена вещественная переменная угла **phi** и величина скорости **v** (если скорость задаётся ползунком **TrackBar**, то её разумно сделать целочисленной), то используемые для расчёта движения скорости по осям **vx** и **vy** можно вычислить так:

$$vx = v * \cos(phi);$$

$$vy = -v * \sin(phi);$$

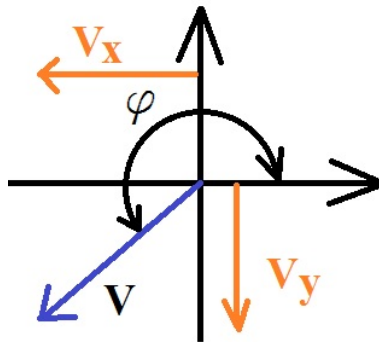


Рис. 1: Картинка с обозначениями.

Здесь угол **phi** указывается в радианах. Полёту вверх соответствует угол от 0 до  $\pi$  ( $180^\circ$ ), полёту в произвольную сторону - от 0 до  $2\pi$  ( $360^\circ$ ) (в формулах учтено, что на канве ось *Y* направлена вниз). Поэтому чтобы при фиксированной скорости полететь в случайном направлении вверх, нужно сгенерировать случайное значение **phi**, для чего используется функция **random()** без указания параметра в скобках, которая возвращает случайное вещественное число от 0 до 1:

```
phi = Pi*random();
```

## Домашнее задание

Усовершенствовать предыдущую программу, изменив движение шарика и добавив рисование блоков, параметры которых хранятся в массиве. Так как первой полугруппе я не успел рассказать двумерные массивы, то пока что можно сделать только один ряд блоков. По желанию - несколько рядов, кроме того можно сделать ползунок регулировки скорости (**TrackBar**).

Об изменении принципов движения написано выше; для рисования блоков нужно завести массивы координат верха, левого края, низа и правого края каждого блока (хотя в принципе достаточно только двух из них). Координаты блока лежащего в **i**-ой строке и **j**-ом столбце можно вычислить например так (**t** - top, верх; **b** - bottom, низ; **l** - left, левый край; **r** - right, правый край; это четыре целочисленных двумерных массива):

```
t[i,j] := 32*j - 25;  
b[i][j] := 32*j;  
l[i][j] := 50* i -40;  
r[i,j] := 50*i;
```

Когда координаты заданы - блоки рисуются с помощью вложенных циклов, аналогично программе "Дождь".