Открытые массивы

При описании открытого массива (в разделе type или var) указывается тип элементов, из которых он состоит (например,
real
,
char
и др.), но не указываются границы индексов. Например:

mas1: array of real;

mas2: array of integer;

В результате получаются как бы безразмерные массивы. Их размер может задаваться и меняться в программе при ее выполнении. Это так называемое динамическое распределение памяти, а переменные открытых массивов представляют собой ничто иное, как указатели на динамически выделяемую область памяти. Т.е. в переменных открытых массивов будут содержаться адреса начала массива, а не сам массив.

Особенностью открытых массивов является то, что их индексы всегда начинаются с нуля (а не с единицы, которая чаще всего используется для обычных массивов).

Чтобы в программе выделить память под открытый массив, следует воспользоваться процедурой setlength, которая принимает два фактических параметра – имя открытого массива и устанавливаемое количество элементов в нем. В результате работы
setlength
в памяти выделяется столько байт, сколько необходимо для хранения
n

-го количества элементов определенного типа. Так, если массив ранее описан как
real
и задано 5 элементов, то процедура
setlength
выделит под него 40 байт, т.к. для хранения каждого числа типа
real
требуется 8 байт памяти (хотя не обязательно 8, это может зависеть от компилятора).

Функция high принимает в качестве параметра имя массива и возвращает индексный номер последнего элемента массива. Например, выделяется память под десять элементов открытого массива; значит, индекс последнего будет равен 9 (т.к. индексация начинается с 0), что и вернет функция high.

Чтобы освободить, выделенную под массив память, используется оператор nil.

Обычно открытые массивы используются для передачи в подпрограмму массивов переменных размеров. Это позволяет с помощью одной и той же подпрограммы обрабатывать массивы произвольной длины. Без использования открытых массивов пришлось бы для каждого массива иной длины писать собственную подпрограмму.

Рассмотрите программу ниже и запустите ее на выполнение. Вам станет более понятно описанное выше.

Примечание. Функция sizeof возвращает количество памяти (в байтах), отведенное под переменную.

```
var

   a: array[1..10] of real;
```

```
    b: array of real;

    i, n: integer;

    sum: integer;

begin

     writeln('Переменная a занимает ', sizeof(a),' байт  памяти.');

    writeln('Переменная  b занимает ', sizeof(b),' байт  памяти.');

     write(' : ');

    readln(n);

    setlength(b,n);

    writeln('Индекс последнего элемента  массива  ', high(b));

     sum  := 0;

    for i:=0 to high(b) do begin
```

```
    sum := sum + sizeof(b[i])

  end;

  writeln('Массив  b занимает в памяти ', sum, ' байт(а);');

  writeln('но  переменная b по-прежнему ', sizeof(b),'.');

   b := nil;

  sum := 0;

  for i:=0 to high(b) do

    sum := sum + sizeof(b[i]);

  writeln('Сейчас  массив b занимает в памяти ', sum, ' байт,');

  writeln('т.к.  память была освобождена с помощью nil.');

readln
```

end.