Figure 11.2 Some rules about *Arrays*.

Here are a few things you need to remember about arrays:

0. All elements of an array are of the same type, say "**T**".
1. An element of an array behaves *exactly* like an ordinary variable of the array's element type, **T**.
2. An array is an object.
3. The number of elements in an array cannot change once you have instantiated the array.

Mainly due to rules 0 and 3, arrays are both harder to use and more efficient to process than Vectors.

Here's how to declare an array:

int[] A; // **A** *is of type "array of* **int***", or* **int[]**, *but not* **int**!
Student[] section; // **section** *is of type "array of* **Student***"*

Note by rule 2, **A** and **section** are just references to array objects, and not actual arrays. Before we have an array there is another step … we must ______________ it:

A = new int[50]; // *now we can use* **A[0] … A[49]**
int classSize;
classSize = Integer.parseInt(keyb.readLine());
section = new Student[classSize];

Note from the **section** example that the size of an array can be determined at run time. Note also that we can't use the **section** array until we instantiate *each* of the **Student** variables contained in that array. We can do this in a loop:

for (int s = 0; ______________ **; s++)**
    section[s] = new Student();

This follows from rule 1; namely that each array element behaves like an ordinary variable of the array element's type.