Array

Array is the most useful data structures, in fact, this data structure will almost always used in all contest problems...

Lets look at the good side first: if index is known, searching an element in an array is very fast, O(1), this good for looping/iteration. Array can also be used to implement other sophisticated data structures such as stacks, queues, hash tables.

However, being the easiest data structure doesn't mean that array is efficient. In some cases array can be very inefficient. Moreover, in standard array, the size is fixed. If you don't know the input size beforehand, it may be wiser to use vector (a resizable array). Array also suffer a very slow insertion in ordered array, another slow searching in unordered array, and unavoidable slow deletion because you have to shift all elements.

Example:

```
int scores[10000]; // integer array of 10000 elements...

// delete item at index 20
for (int i=20; i<10000; i++) // shift one to the left
  scores[i] = scores[i+1];
scores[9999] = 0; // delete the last one

// insert an item X at index 20
for (int i=21; i<10000; i++) // shift one to the right
  scores[i] = scores[i-1];
scores[20] = X; // insert the new item here
```

	No duplicate allowed	Duplicate items allowed
Search	O(n/2) comparisons	O(n) comparisons
Insertion	No comparison, O(1)	No comparisons, O(1) move
Deletion	O(n/2) comparisons O(n/2) moves	O(n) comparisons more than O(n/2) moves

We commonly use one-dimensional array for standard use and two-dimensional array to represent matrices, board, or anything that two-dimensional. Three-dimensional is rarely used to model 3D situations, but higher dimension is very hard to visualize and beyond the capabilities of human brain (unless you are very very genius).

For 2D array, there are two types of accessing two-dimensional array, please keep this in mind, this will speed up your code considerably.

Row major, cache-friendly, use this method to access all items in array sequentially.

```
for (i=0; i<numRow; i++)    // ROW FIRST = EFFECTIVE
  for (j=0; j<numCol; j++)
    array[i][j] = 0;
```

Column major, NOT cache-friendly, DO NOT use this method or you'll get very poor performance. Why? you will learn this in Computer Architecture course. For now, just take note that computer access array data in row major.

```
for (j=0; j<numCol; j++)    // COLUMN FIRST = INEFFECTIVE
  for (i=0; i<numRow; i++)
    array[i][j] = 0;
```

Taken from the site:
http://www.comp.nus.edu.sg/~stevenha/programming/prog_datastructures1.html