1. An array is a data structure that allows you to store multiple values of the same data type. We have already seen an example since a C/C++ string is an array of characters. For example,

```c
const int SIZE = 50;
char name[SIZE] = "Johe Doe"; // allow up to 50 characters in the array name
char name2[SIZE];
```

Arrays are implemented as a contiguous block of memory which is referenced (pointed at) by the the name of the array.

```
const int SIZE = 50;
char name[SIZE] = "Johe Doe"; // allow up to 50 characters in the array name
char name2[SIZE];
```

Individual elements in an array are assigned subscripts/indexes starting at zero. Usually, index 0 holds the first data value with any empty elements being on the "right" end of the array. Individual elements can be accessed by specifying an index value in the "square-brackets," [ ].

```c
cout << name << endl;
```

a) What would get printed by the above "cout" statements?

We can assign individual elements new values using the same "square-brackets," [ ] syntax, e.g.,

```c
name[2] = 'z';
cout << name << endl;  // Jozn Doe
```

b) What assignment statements could we use to turn the name string into "Jane Doe"?

c) WARNING: You cannot assign the content of one array to another!!! For example,

```c
name2 = name;  // causes the error: '=' : left operand must be l-value
```

Assigning the contents of an array to another array must be done an element at a time using a loop. Write a loop to copy array name to array name2.
2. Arrays holding other data types (besides char’s) can also be initialized when defined using set-brackets, { }:

```cpp
const int MONTHS = 12;
```

The number of values in the set-brackets can be less than the array size, but not more. You can use implicit array sizing by not specifying the array size and letting the compiler create an array to exactly hold the initialized content:

```cpp
int misc[] = {0, 5, 2, 4}; // misc
char myName[] = "John";
```

a) How many elements are the above arrays?

- misc?

- myName?

b) Typically, arrays are used in situations where the programmer does not know exactly how many items will need to be stored in the array, but has an idea on an upper bound on the number of items to be stored. Therefore, the programmer must maintain a count of the number of items stored in the array. Write code to read in a set of scores to be averaged.