

## CS 1044, Summer II 2000 Homework #2 Solution

1) Write the C++ code to declare an array of integers of some constant size, `MAX_SIZE`. Initialize each element in the array with the number of its index. For example, the first slot in the array would get a 0, the second slot would get a 1, and so on.

```
int Array [ MAX_SIZE ];  
    for ( int index = 0 ; index < MAX_SIZE ; index ++ )  
        Array [ index ] = index ;
```

2) What is the error in the following code, and what will be the result:

```
char Array [10];  
Array[10] = '\0';
```

The valid range of index values is 0 through 9. In our case, the second statement uses an array index that is out of bounds. This is a logical error in processing arrays. The result is that a memory location outside the array is accessed; destroying whatever value was contained there. C++ does not check for out-of-bounds array indexes either at compile time or at run time, so we should be careful with it.

3) Assume that X and Y are arrays of type int. Which of the following operations on arrays are valid:

|                |                     |           |
|----------------|---------------------|-----------|
| Assignment:    | X = Y;              | Non Valid |
| Equality Test: | X == Y;             | Non Valid |
| I/O:           | cout << X;          | Non Valid |
| Arithmetic:    | X + Y;              | Non Valid |
| Return:        | return (X);         | Non Valid |
| Parameter:     | SomeFunction(X, Y); | VALID     |

The only thing we can do with an array as a whole is to use it as a parameter to a function.

4) What should the **minimum** size of the array be? Explain your answer.  
`char String1[SIZE] = "Hello World!";`

|   |   |   |   |   |  |   |   |   |   |   |   |    |
|---|---|---|---|---|--|---|---|---|---|---|---|----|
| H | e | l | l | o |  | W | o | r | l | d | ! | \0 |
|---|---|---|---|---|--|---|---|---|---|---|---|----|

The minimum size of the array should be 13. The dimension of the char array should always be one greater than the length of the string to be stored. This happens because when the specified array initialization is done, a special string termination character called NULL character is inserted automatically.

5) Assume an input filestream, inFile (it has been declared and opened) that contains a last name and a first name in the format *Last^Firs^Middle* (this is based on the HL7 message standard used in healthcare information systems). Write the C++ code to extract each name into it's own array. Then, combine the names into a single array in the form *First MiddleInitial. Last* (with spaces in between the names). Keep in mind that the input will contain the full middle name, but the output should only contain the first initial of the name, followed by a "." (period). You can assume a MAX\_SIZE for all arrays of 256.

```
const int MAX_SIZE=256;           // The maximum size of the arrays
char Last[MAX_SIZE];             // Array for the last name
char First[MAX_SIZE];           // Array for the first name
char Middle[MAX_SIZE];          // Array for the middle name
char FullName[MAX_SIZE];        // Array that contains the combined name

inFile.get ( Last, MAX_SIZE, '^'); // Read the last name
inFile.ignore ( 200, '^' );       // Ignore the character '^'
inFile.get( First, MAX_SIZE, '^' ); // Read the last name
inFile.ignore ( 200, '^' );       // Ignore the character '^'
inFile.get ( Middle, MAX_SIZE );  // Read the middle name

// The following creates a C string that contains only the first character of the
// middle name

Middle[1] = '\0';                // Assign the null character

strcpy (FullName, First);        // Copy the first name into the full name
strcat (FullName, " ");         // Concatenate the space onto the end of full
strcat (FullName, Middle);      // Concatenate the initial of middle name
strcat (FullName, ". ");       // Concatenate a period and a space
strcat (FullName, Last);        // Concatenate the last name

cout << FullName << endl;
```