Computer Science 1MC3
Lab 5 – Arrays

ARRAYS

An array is a collection of similar items or objects stored under the same name. The individual elements of the array are called the array elements. Arrays allow us to store similar data more conveniently under one name. Arrays have a couple qualities which we should be aware of:

1. An array can only store one type of data, i.e. integers, floats, characters. . .
2. When referencing the elements of an array in C, the first element is stored at 0 **not** 1.

DECLARTION

Just like any other variable you use in c, an array has to be declared at the beginning of your program in order for the compiler to reserve the appropriate amount of memory. The declaration looks as follows

```
data_type array_name[size_of_array];

      ~or~

data_type array_name[]={array elements separated by commas -> ","}
```

For example an integer array declaration filled with the first ten numbers would look like this:

```
int array[]={0,1,2,3,4,5,6,7,8,9};
```

However had we not filled the array initially we could also do this:

```
int array[10];

for(index=0;index<10;index++) {
      array[index]=index;
}
```

REFERENCING

Now that we have created an array it would be nice to access each individual element. This is very easy, an individual array element is accessed like `array_name[index];` and can be treated exactly as a variable. So say we would like to make one of our array elements 5 we could do so like this:

```
array_name[3]=5;
```

We can even make array elements equal to other array elements as such:

```
array_name[2]=array_name[3]+5;
```

SIZE OF AN ARRAY

As covered in class to determine the length of an array we do:
`sizeof(array)/sizeof(array element);`
Please refer to lecture notes 10 for an explanation why.

STRINGS

You may recall that it is possible to have characters in C. A character is a single letter like ‘a’ or ‘F’. A group of characters, or a word, is called a string in C.

DECLARATION

There is actually no data type reserved for strings in C. Instead we create arrays of character which serve the same purpose. We would declare a word, or sentence as follows:

```
char string_name={"Sentence or word contained in quotes"};
```

The last element of this array actually contains a null terminator (`\0`), which is something the computer puts in by default to indicate that the string is over. If we didn’t have a null terminator we would have no way to tell if a sentence was finished since putting in blanks, or spaces, at the end of you sentence is completely acceptable. So this said keep in mind when finding the length of a string, that the string will be one longer then expected because of this null terminator.

USAGE

You may use a string in exactly the same fashion in which you use an array, specific elements may still be referenced, changed, or tested. However `%s` will be introduced to use with the `printf` command. `%s` much like its cousins `%d` or `%f` indicates to the `printf` command to print out a variable. You may have guessed the ‘s’ in `%s` stands for string, you would be correct, here is an example on how to use `%s`:

```
char word[]={"hello world"};

printf("%s",word);
```

would print to the screen: `hello world`

HOMEWORK

1. Design a program to take two arrays of equal length , A1 and A2, and point wise add them into A3. For example [1,2,3]+[4,5,6]=[5,7,9]. Print your results to the screen.

2. Take the `sizeof` the following arrays and put the solution one the line.

	YOUR GUESS	CHECK W/ COMPUTER
`char y[]={"hello world"}`		
`flot z[]={4.67,3.8};`		
`int x[]={1,2,3,4};`		

3. Using `%s` output your name to the screen.

4. Create a program to count the amount of a’s, b’s. . .z’s in a string. Store the word frequency in array then output to the screen your results. You may assume large case letter are not used.

   ```
   char string[]={"this program was a tough one to do"};
   ```