CS 1428
Lab 8 Sections L06 and L19

Jared Wallace

## 1-D Arrays and Character Arrays

1 dimensional arrays and character arrays (often called "C-strings") are good examples of the different types of array processing you will encounter. The first example is when we know the size of the array (I hope a bell just rang in your head about which loop to use).

Using a for loop with the size of the array as the limit will allow us to visit each element of the array one at a time and do whatever we want to with it.

```
const int SIZE = 15;
int arr[SIZE];

for (int i = 0; i < SIZE; ++i)
   cout << arr[i] << " \n";
```

C-strings utilize what is known as a sentinel value, specifically a null character ('\0'). With a while loop we can watch for that null terminator and know when to stop.

```
char arr[] = "C-string";          // ['C' '-' 's' 't' 'r' 'i' 'n' 'g' '\0']
int index = 0;

while(arr[index] != '\0') {
   cout << arr[index];
   index++;
}

// This can also be done with a for loop
for(int i = 0; arr[i] != '\0'; ++i)
   cout << arr[i];
```

## C Strings

I want you to declare a C-string with the contents "I am a C-string". You will create several new strings that contain manipulations of this original string but at the end of the program this string should be still intact. You might find it helpful to declare a constant that holds the size of the array. You may NOT use this size as a limit when there is a sentinel value but it will be useful for declaring new arrays. Each of the following tasks should be performed on the original string, not on the result of the previous task.

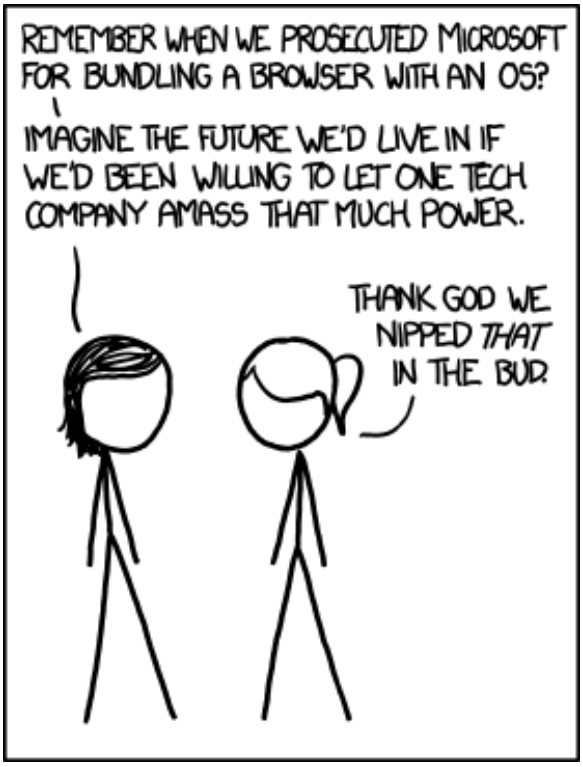
1. First I want you to create a new copy of the string that is in reverse. Recall that to be a valid C-string the null terminator (which is not considered part of the content) must be at the end of the string before we declare the operation complete.

2. Recall during the discussion of switch statements when we discussed that characters are stored in the computer as integers. When you assign a character to an integer the integer takes the value of that ASCII value. Create an integer array (SIZE - 1 since there is no null terminator) that contains the integer values of each character.

3. By checking the if the integer value of a particular character is within a certain range $97 \leq$ char $\leq 122$ we can determine if it is a lowercase character. We can then convert it to uppercase by subtracting 32 to that value. Visit each item in the original array and set the corresponding positions in a new array with the upper-case character that corresponds.

Print each of these arrays out to the screen in readable format including labeling what each one is.

**Upload your source code through the homework upload utility. Place a printout of you source code behind this page and staple before turning it in at my desk**




Facebook, Apple, and Google all got away with their monopolist power grabs because they don't have any S's in their names for critics to snarkily replace with $'s