

# Strings

## Interview Skills

Richard Morrill

Fordham University CS Society

Wednesday, February 12th 2020



- The questions will be based on C-style strings.
  - You'll have to deal with them at some point.
  - It also means many skills will translate to arrays.
  - If you want to use fancy C++ stuff feel free.

- The questions will be based on C-style strings.
  - You'll have to deal with them at some point.
  - It also means many skills will translate to arrays.
  - If you want to use fancy C++ stuff feel free.
- Make sure you understand the given sample case, and try to come up with your own.

- The questions will be based on C-style strings.
  - You'll have to deal with them at some point.
  - It also means many skills will translate to arrays.
  - If you want to use fancy C++ stuff feel free.
- Make sure you understand the given sample case, and try to come up with your own.
- Go with the simplest solution you can think of, but keep efficiency in the back of your mind.

# Helpful Hints

- Just an array of `char`.
- The string variable is really just a pointer to the first character.
- There is a NULL (`'\0'`) character at the end of every string.
- Use `strlen(str)` to find the length (Won't include NULL).
- `'a' - 'a' == 0` and `'z' - 'a' == 25` Use this information wisely.
- Don't bother implementing simple functions such as `swap(a, b)` or `max(a,b)`
- C-Strings always pass by reference (pointer).



CS SOCIETY  
FORDHAM UNIVERSITY

# Problem 1: Reverse a String (5 minutes)

Use function signature `void reverse(char* str).`

Do I really have to spell this one out for you?

# Problem 1 solution

```
1 void reverse(char* str) {  
2     // Why did I store this value in a variable?  
3     int n = strlen(str);  
4  
5     for (int i = 0; i < n / 2; i++) {  
6         swap(str[i], str[n - i - 1]);  
7     }  
8 }
```



CS SOCIETY  
FORDHAM UNIVERSITY

# Problem 1.1: Palindrome (2 minutes)

Quick! Take your solution to problem 1 and make it tell me if a string is a palindrome.

Examples of Palindromes: racecar tacocat mom



## Problem 2: Longest Common Substring (12 minutes)

Given two strings, return the length of the longest common contiguous segment.

Examples:

- seminar seminarian  $\rightarrow 7$
- thomas massachusetts  $\rightarrow 3$
- abcafledt bbcaztledtor  $\rightarrow 4$

# Problem 2 Solution

```
1  int LCS(char* x, char* y) {
2      int xlen = strlen(x);
3      int ylen = strlen(y);
4
5      int result = 0;
6
7      for (int xstart = 0; xstart < xlen - 1; ++xstart) {
8          for (int ystart = 0; ystart < ylen - 1; ++ystart) {
9              int xi = xstart;
10             int yi = ystart;
11
12             int currLen = 0;
13             while(xi < xlen && yi < ylen && x[xi] == y[yi]) {
14                 ++currLen;
15                 ++xi;
16                 ++yi;
17             }
18
19             if (currLen > result) {
20                 result = currLen;
21             }
22         }
23     }
24     return result;
25 }
```



CS SOCIETY  
FORDHAM UNIVERSITY

# Follow Up for Problem 2

The solution I presented was incredibly inefficient.

How could we make this more efficient?



CS SOCIETY  
FORDHAM UNIVERSITY™

## Problem 3: Most Common Character (8 minutes)

Given an input string consisting only of lowercase letters, return the most commonly used character.

# Problem 3 Solution

```
1  char mostCommon(char* str) {
2      int n = strlen(str);
3      int buff[26];
4      buff[0] = 0;
5
6      for (int i = 0; i < n; ++i) {
7          ++buff[str[i] - 'a'];
8      }
9      int max = 0;
10     char maxChar = '\0';
11     for (int i = 0; i < 26; ++i) {
12         if (buff[i] > max) {
13             max = buff[i];
14             maxChar = i + 'a';
15         }
16     }
17     return maxChar;
18 }
```



CS SOCIETY  
FORDHAM UNIVERSITY

## Problem 4: Compression (12 minutes)

Given a string of lowercase letters, replace every repeating letter with that letter followed by the number of times it occurred.

### Examples

- aabccc → a2bc3
- fffffaaff → f5a2f2

# Problem 4 Solution

```
1  char* compress(char* input) {
2      int inputLen = strlen(input);
3      char* output = new char[inputLen];
4      int outputi = 0;
5
6      char currChar = input[0];
7      int currCharCount = 0;
8      for (int inputi = 0; inputi < inputLen; ++inputi) {
9          if (input[inputi] == currChar) {
10             ++currCharCount;
11         } else {
12             output[outputi] = currChar;
13             ++outputi;
14
15             currChar = input[inputi];
16             if (currCharCount > 1) {
17                 output[outputi] = currCharCount + '0';
18                 ++outputi;
19             }
20             currCharCount = 1;
21         }
22     }
23     output[outputi] = '\\0';
24     return output;
25 }
```



CS SOCIETY  
FORDHAM UNIVERSITY