Strings

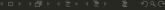
Interview Skills

Richard Morrill

Fordham University CS Society

Wednesday, Febuary 12th 2020

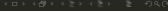




General Info

- 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.
 - \bullet If you want to use fancy C++ stuff feel free.





General Info

- 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.

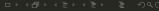




General Info

- 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).





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

```
void reverse(char* str) {
    // Why did I store this value in a variable?
    int n = strlen(str);

for (int i = 0; i < n / 2; i++) {
        swap(str[i], str[n - i - 1]);
    }
}</pre>
```



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



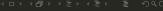
Problem 2: Longest Common Substring (12 minutes)

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

Examples:

- seminar seminarian \rightarrow 7
- ullet thomas massachusets ightarrow 3
- abcafledt bbcaztledtor \rightarrow 4





Problem 2 Solution

```
int LCS(char* x, char* y) {
    int \times len = strlen(x);
    int \text{ ylen = strlen(y)};
    int result = 0;
    for (int xstart = 0; xstart < xlen - 1; ++ xstart) {
        for (int ystart = 0; ystart < ylen - 1; ++ ystart) {</pre>
             int \times i = xstart:
             int vi = vstart:
             int currlen = 0:
             while (xi < xlen && yi < ylen && x[xi] == y[yi]) {
                ++currLen:
             if (currLen > result) {
                 result = currLen:
    return result:
```



Follow Up for Problem 2

The solution I presented was incredibly inefficient.

How could we make this more efficient?



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

```
char mostCommon(char* str) {
    int n = strlen(str);
    int buff[26];
    buff[0] = 0;

for (int i = 0; i < n; ++i) {
        ++buff[str[i] - 'a'];
    }
    int max = 0;
    char maxChar = '\0';
    for (int i = 0; i < 26; ++i) {
        if (buff[i] > max) {
            max = buff[i];
            maxChar = i + 'a';
    }
}
return maxChar;
```



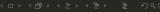
Problem 4: Compression (12 minutes)

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

Examples

- $aabccc \rightarrow a2bc3$
- $\bullet \ \, \mathsf{fffffaaff} \to \mathsf{f5a2f2}$





Problem 4 Solution

```
char* compress(char* input) {
    int inputLen = strlen(input);
    char* output = new char[inputLen];
    int output = 0:
    char currChar = input[0]:
    int currCharCount = 0:
    for (int inputi = 0; inputi < inputLen; ++inputi) {</pre>
        if (input[inputi] == currChar) {
            ++currCharCount:
            output[outputi] = currChar:
            ++outputi;
            currChar = input[inputi];
            if (currCharCount > 1) {
                output[outputi] = currCharCount + '0';
                ++outputi;
            currCharCount = 1:
    output[outputi] = '\0';
    return output;
```

