Eric Roberts and Jerry Cain
CS 106J
Handout #32
May 10, 2017

Section Handout #5: More Strings, A Bit of Arrays

Portions of this handout by Eric Roberts

Congratulations on finishing your midterm! ☺ Thank you all for working so hard these past weeks. We are all proud of how much you have learned and accomplished already, and we look forward to the rest of the quarter! – staff

Note: To simplify problems 1-4, you can assume all strings will only contain lower-case letters a-z . That said, feel free to make your functions more robust if you wish.

1. Duplicate characters

Write a function **`findDupChars(str)`** that takes a string and returns a list of all the duplicate characters and their counts (any order is okay). For example:

```
findDupChars ("apple") => "p: 2"
findDupChars ("banana") => "a: 3; n: 2"
findDupChars ("stanford") => ""
```

2. Remove duplicate characters

Now that you have a function to detect duplicate letters, can you remove all duplicate letters from a string? Write a function **`removeDupChars(str)`** that takes a string and returns the string with all duplicate characters removed. For example:

```
removeDupChars ("apple") => "ale"
removeDupChars ("banana") => "b"
removeDupChars ("stanford") => "stanford"
```

3. Is anagram

Write a function **`isAnagram(str1, str2)`** that takes two strings and returns `True` or `False` depending on whether the two are anagrams. An anagram is defined as a word that can be formed by rearranging the letters of another word. For example:

```
isAnagram("listen", "silent") => True
isAnagram("iamlordvoldemort", "tommarvoloriddle") => True
isAnagram("banana", "apples") => False
```

4. First non-repeating character

Write a function **`firstNonRepeatingChar(str)`** that takes a string and returns the first unique character in the string when reading from left to right. For example:

```
firstNonRepeatingChar ("aardvark") => "d"
firstNonRepeatingChar ("banana") => "b"
firstNonRepeatingChar ("stanford") => "s"
```

5. indexOf

We've used the **`str.indexOf(pattern)`** function previously when we learned about strings. Now, can you implement it yourself? Our syntax will be differ slightly since `str` will be passed as a parameter: **`indexOf(str, pattern)`**. For example:

```
indexOf("substrings", "string") => 3
indexOf("Stanford", "tree") => -1
indexOf("bananas", "an") => 1
```

6. Remove a specific word from string

Imagine you are J.K. Rowling, and you notice the night before you're supposed to publish *Harry Potter and the Sorcerer's Stone* that some unknown foreign entity has hacked into your computer and inserted the word "HACKED" everywhere in your manuscript.

It's 4 hours before it is due to the publisher, and you don't have time to read through the file and find every place where the hackers wrote "HACKED". Instead, since you took CS106J back in college, you decide to write a function **`removeUnwantedWord(manuscript, toRemove)`** that removes all instances of an unwanted word.

For example:

```
var manuscript = "Mr. and Mrs. Dursley, of number four, Privet Drive, were proud to say that they were perfectly HACKED normal, thank you very HACKED much. They were the last HACKED people you'd expect to be HACKED involved in anything strange or mysterious, because they just didn't hold with such HACKED nonsense... ..."

removeUnwantedWord(manuscript, "HACKED") =>

"Mr. and Mrs. Dursley, of number four, Privet Drive, were proud to say that they were perfectly normal, thank you very much. They were the last people you'd expect to be involved in anything strange or mysterious, because they just didn't hold with such nonsense... ..."
```