String Applications

Jerry Cain
CS 106AJ
October 24, 2018
slides courtesy of Eric Roberts and Jerry Cain

String Calisthenics

Let’s review some String methods you learned about last time:

- "AEIOUaeiou".length 10
- "ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(6) "G"
- "Harry Potter".indexOf("a") 1
- "Harry Potter".indexOf("a", 6) -1
- "Harry Potter".lastIndexOf("tt") 8
- "bumfuzzle".substring(3, 7) "fuzz"
- "cabotage".substring(1, 1) "m"
- "agelast".substring(3) "gest"

Generating Acronyms

- An acronym is a word formed by taking the first letter of each word in a sequence, as in "North American Free Trade Agreement" -> NAFTA
- "self-contained underwater breathing apparatus" -> scuba

- The text describes and implements two versions of a function acronym(str) that generates an acronym for str:
  - The first version searches for spaces in the string and includes the following character in the acronym. This version, however, fails for acronyms like scuba, in which some of the words are separated by hyphens rather than spaces.
  - The second version looks at every character and keeps track of whether the algorithm is scanning a word formed composed of sequential letters. This version correctly handles scuba as well as strings that have leading, trailing, or multiple spaces.

acronym, Take I

```java
function acronym(str) {
  let result = str.charAt(0);
  let sp = str.indexOf(" ");
  while (sp != -1) {
    result += str.charAt(sp);
    sp = str.indexOf(" ", sp + 1);
  }
  return result;
}
```

```
acronym("not in my back yard")
```

```
not in my back yard
```

```
"nimby" -1
```

```
triples: 4 matches
```

acronym, Take II

```java
function acronym(str) {
  const ALPHABET = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  function isLetter(ch) {
    ALPHABET.indexOf(ch.toUpperCase()) != -1
  }
  let result = "", sp = false, i = 0;
  while (isLetter(str[i])) {
    if (!sp) {
      if (isLetter(str[i + 1])) {
        result += str[i + 1] + "-";
        sp = true;
      } else {
        result += str[i] + "-";
        sp = false;
      }
    } else {
      if (isLetter(str[i + 1])) {
        result += str[i + 1] + "-";
      } else {
        result += str[i] + "-";
        sp = false;
      }
    }
    i++;
  }
  return result;
}
```

```
acronym("In My Humble Opinion")
```

```
"IMHO" 20 true "m"
```

Translating Pig Latin to English

Section 7.4 works through the design and implementation of a program to convert a sentence from English to Pig Latin. In this dialect, the Pig Latin version of a word is formed by applying the following rules:

1. If the word begins with a consonant, the wordToPigLatin function moves the initial consonant string to the end of the word and then adds the suffix ay, as follows:
   ```
   scram ➔ actrams
   ```

2. If the word begins with a vowel, the wordToPigLatin generates the Pig Latin version simply by adding the suffix way, like this:
   ```
   apple ➔ appleway
   ```

3. If the word contains no vowels at all, the wordToPigLatin returns the original word unchanged.
   ```
   "" ➔ ""
   ```
Translating Pig Latin to English

"stout plunder lover"

• `inWord` is true if and only if we’re in a word, and `start` is the index of the first character of the word we’re currently in (or -1 if we’re not in a word).
• `inWord` is now true and `start` is set equal to 0. We set assign the value of `i` to `start` at the same time `inWord` is transitioning from false to true, so we can remember where the current word of interest begins.

Pseudocode for the Pig Latin Program

```plaintext
Function toPigLatin(str) {
    Initialize a variable called result to hold the growing string.
    for (each character position in str) {
        if (the current character is a letter) {
            if (we're not yet scanning a word) Remember the start of this word.
            else {
                if (we were scanning a word) {
                    Call wordToPigLatin to translate the word.
                    Append the translated word to the result variable.
                }
                Append the separator character to the result variable.
            }
        }
    }
    if (we're still scanning a word) {
        Call wordToPigLatin and append the translated word to result.
    }
}
Function wordToPigLatin(word) {
    Find the first vowel in the word.
    If there are no vowels, return the original word unchanged.
    If the vowel appears in the first position, return the word concatenated with "way".
    Divide the string into two parts (head and tail) before the vowel.
    Return the result of concatenating the tail, the head, and the string "ay".
}
```

Simulating the Pig Latin Program

```plaintext
> toPigLatin("this is pig latin")
"isthay isway igpay atinlay"
```

The End