# Strings Manipulation

Colorado State University

Announcements

TODO Reminders:

Readings are due **before** lecture

- Reading 11 (zybooks) – you should have already done that ☺
- Lab 07 - go to your lab to have the participation points
- Reading 12 (zyBooks) - you should have already done that ☺
- Lab 08 – go to your lab to have the participation points
- Reading 13 (zybooks) - you should have already done that ☺
- RPA 6

Keep practicing your RPAs in a spaced and mixed manner ☺
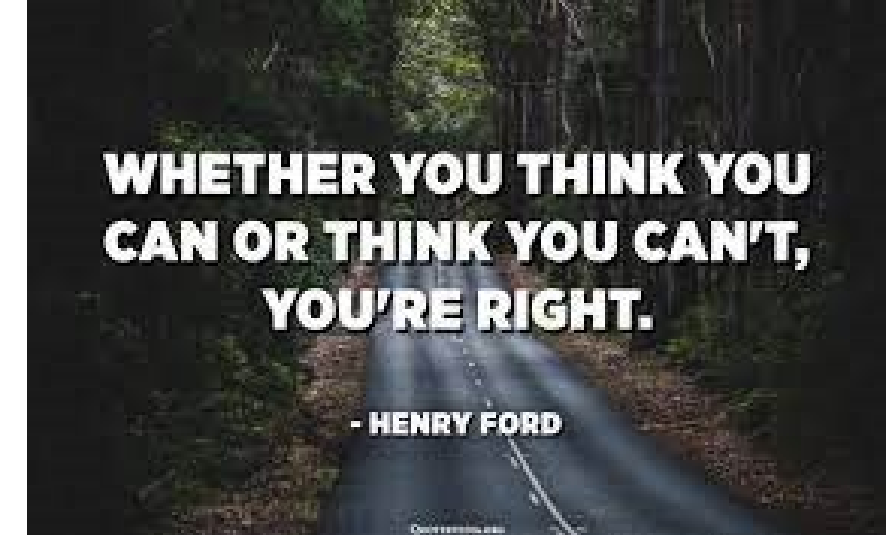
Help Desk

Day	Time : Room
Monday	3 PM - 5 PM : CSB 120
Tuesday	6 PM - 8 PM : Teams
Wednesday	3 PM - 5 PM : CSB 120
Thursday	6 PM - 8 PM : Teams
Friday	3 PM - 5 PM : CSB 120
Saturday	12 PM - 4 PM : Teams
Sunday	12 PM - 4 PM : Teams

Colorado State University

Recall Activity

- **Grab a paper, write your name, as it is in Canvas, and your answers to the following questions. Turn this as your attendance for today's lecture.**

- Explain what is a String, describe at least three methods you can use over Strings and how those methods work.

Strings

- A String is a collection of ordered characters
  - It has data
  - It has functionality (methods)
  - It is also **immutable** (can't be directly modified)
    - Every method that builds a String, returns a copy
    - Java does this for memory efficiency
- Example
 - String mascot = "CAM";

```
String mascot
```

0	C
1	A
2	M
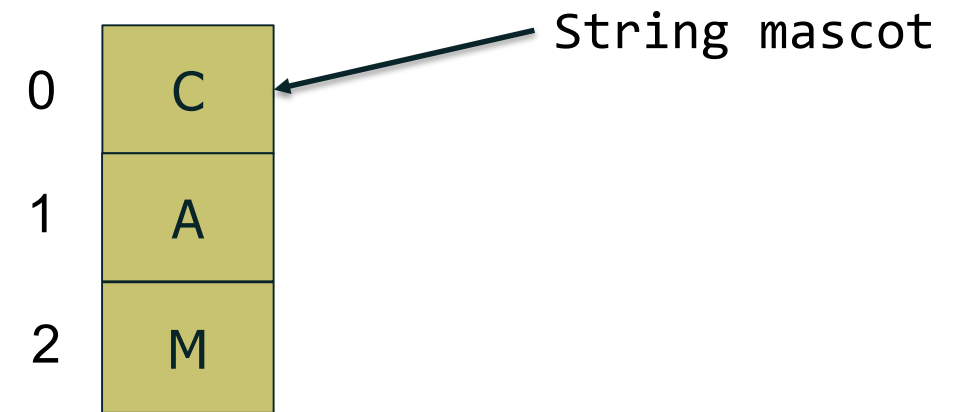
Common Strings Methods

- .charAt(int) – gives the character at a location

- .indexOf(char) – gives the location of character

- .indexOf(String) – overloaded option, gives the location of the start of the String that matches

- .indexOf(char, int) or indexOf(String, int) – same as above, but changing starting location

- .lastIndexOf(char) – gives the location starting at the end and working down (also has a String version)

- .substring(int start, int end) – returns a substring start-end (including start and excluding end)

- .toLowerCase() – returns the lower case version of the String

- .toUpperCase() – returns the upper case version of the String

Finding the index

0	k
1	i
2	n
3	n
4	i
5	k
6	i
7	n
8	n
9	i
10	k

stops searching, returns 1

stops searching, returns 2
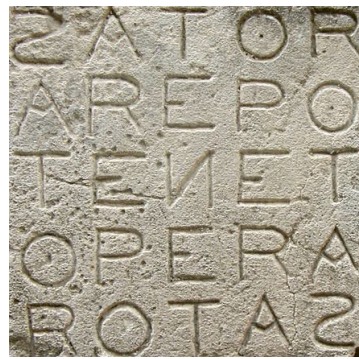
What are the outputs for the following instructions?

```
String palindrome  = "kinnikinnik";

palidrome.length(); // returns 11

palidrome.charAt(0);  // return 'k'

palidrome.charAt(palidrome.length()-1); // return 'k'

palidrome.indexOf('i');     // return 1

palidrome.indexOf('n');     // return 2

palidrome.indexOf('niki'); // return 3
```

Colorado State University

lastIndexOf

0	k
1	i
2	n
3	n
4	i
5	k
6	i
7	n
8	n
9	i
10	k

- String palindrome = "kinnikinnik";

- palidrome.lastIndexOf('n');

What if we wanted the middle 'k'?

Practice Loop Review

- Write a loop that :

  - Looks at each character in a String (hint: charAt(i))

  - Returns the index of that character (don't use 'indexOf')

    - Return -1 if the loop ends but the character was not found.

  - Now modify it to start at the end of the word, not the start!

  Complete the In Class Activity

```
public class YourProgram {

  public static int find(String str, char ch) {
      return find(str, ch, 0);
  }



  public static int find(String str, char ch, int start) {
      // LOOP here
      return 0; // change this
  }


  public static void main(String[] args) {
      System.out.printf("TEST: the index is: %d", find("SATOROTAS", 'A'));
      System.out.printf("TEST: the index is: %d", find("SATOROTAS", 'T'));
      System.out.printf("TEST: the index is: %d", find("SATOROTAS", 'Z'));
  }
}
```

indexOf is overloaded

0	k
1	i
2	n
3	n
4	i
5	k
6	i
7	n
8	n
9	i
10	k

- indexOf can take in two parameters
  - Second parameter is the **start** location

```
String palindrome = "kinnikinnik";

palindrome.indexOf('k'); // returns 0

palindrome.indexOf('k', 1); // returns 5
```

Substring

0	k
1	i
2	n
3	n
4	i
5	k
6	i
7	n
8	n
9	i
10	k

- Returns a portion of the string

- substring(start, end)
  - includes start
  - excludes end! (end is optional, defaults to .length())

```
String palindrome = "kinnikinnik";

String sub = palindrome.substring(0, 6); // sub is "kinnik"
```

```
int start = palindrome.indexOf("k");

int end = palindrome.indexOf("k", start+1) + 1;

String sub = palindrome.substring(start, end);
```

Exploring Patterns

- What pattern can be observe in the following Strings?

```
Fort Collins,40°35'6.9288"N,105°5'3.9084"W
Denver,39°44'31.3548"N,104°59'29.5116"W
Boulder,40°0'53.9424"N,105°16'13.9656"W
```

City,Latitude,Longitude

- How can we parse that String to have each one of those elements individually?

Using substring method combined with indexOf method!

Colorado State University

Exploring Patterns – Getting the City

- What pattern can be observe in the following Strings?

```
Fort Collins,40°35'6.9288"N,105°5'3.9084"W
Denver,39°44'31.3548"N,104°59'29.5116"W
Boulder,40°0'53.9424"N,105°16'13.9656"W
```

- Example: getting the city

Complete the In Class Activity

```java
public class Test {
    public static void main(String args[]){
        String coord = "Fort Collins,40°35'6.9288\"N,105°5'3.9084\"W";
        String city = coord.substring(0, coord.indexOf(","));
        System.out.println(city);
    }
}
```

Colorado State University

Substring Practice

- Write a method that returns all characters after a given character
    - Example: sub("SATOROTAS", 'O') // return ROTAS
    - Example: sub("SATOROTAS", 'A') // returns TOROTAS

- Think about the problem that you need to solve

- Write a sequence of steps to solve that problem – your algorithm

- Translate your algorithm into a Java program

Take-away

- All Strings have indices from 0 to length-1

  - indexOf(char) – finds the index of a character or substring

  - charAt(int) - gives you the character at an index

  - substring(int,int) – gives you a portion of the string

  - length() – don't forget to use it!

Additional Reading

Read more about the String Class : Here
Read more about the Character Class : Here

Colorado State University