The String Class
Instantiating String Objects

- `String name = new String(“Bob”);`
  - could leave blank to initialize with empty string
  - can also be initialized with StringBuffer variable

- `String name = “Bob”;`
- `String name;`
  - `name = “Bob”;`
Comparing Strings

Since String objects store addresses, we cannot compare them with ==

- equals(Object o)
  - name1.equals(name2);
  - name.equals(“Bob”);

- equalsIgnoreCase(Object o)
Comparing Strings (cont’d)

- `compareTo(Object o)`
  - returns 0 if strings hold same value
  - negative if calling object is less than argument
  - positive if calling object is greater than argument
  - `callingObject.compareTo(argument)`

Ex:  `String s1 = “Jane”;`
     `String s2 = “Bob”;`
     `int x = s1.compareTo(s2);  //x is positive`
     `int y = s2.compareTo(s1);  //y is negative`
Other String Methods

- `charAt(int index)`
  - `name.charAt(1) → 'o'`

- `indexOf(String s)`
  - Can take one char, a char and int, a String, or a String and an int
  - If not found, -1 is returned
  - Ex: `String s1 = "jumping"; System.out.print(s1.indexOf("ing")); //4`

- `endsWith(String s)`
  - Takes a String as argument and returns boolean value of true if the calling object ends with that argument
More Methods

- `replace(char c1, char c2)`
  - takes two parameters – both characters
  - replaces all occurrences of first argument with second argument

- `toUpperCase()`
  - Ex: String s2 = "bob";
  - System.out.print(s2.toUpperCase()); //BOB

- `toLowerCase()`
Substring

- substring(int from)
- substring(int from, int to)
  - may have one or two arguments
  - One argument gives beginning index only, two gives beginning and the index following where to stop

Ex: String s1 = "jumping";
    System.out.print(s1.substring(2,5));  // mpi
What’s on the AP?

- `compareTo(Object other)`
- `equals(Object other)`
- `length()`
- `substring(int from)`
- `substring(int from, int to)`
- `indexOf(String s)`
- `+`