CS 134:
Java (3)

Slide content based on http://www.cs.cmu.edu/~mjs/courses/121-F14-W/Java4Python.pdf

Announcements & Logistics

- **HW 9** due tonight @ 11pm
  - Covers "advanced" topics from recent lectures
  - Review special methods, iterators, efficiency
- **Lab 10 Selection Sort in Java**: today/tomorrow in lab
  - Due Wed/Thurs @ 10 pm
  - Hope most of you will start and finish during your lab session
- **Final exam reminder: May 22 @ 9:30 am**
  - Another (early) option: May 18 @ 1pm. Submit Google form!
  - Practice problems: we will release later this week
  - Review session and office hours next week: details TBD
- **Course evals on Friday**: bring a laptop to class if possible

Last Time

- Discussed an example of how to read input in Java, do basic arithmetic and print the output
- Introduced **data types** in Java:
  - **Strings**
  - **ArrayLists** and **Arrays** (like Python lists)
  - **HashMaps** (like Python dictionaries)
- Briefly talked about **conditional statements**: very similar to Python!

Booleans

- **Boolean** (or **boolean**) values in Java:
  - **true** and **false** (no capitalization)
  - Example: **Boolean b = true**
- **Boolean** operators in Java**:**
  - **&&** - and
  - **||** - or
  - **!** - not
  - Most other operators (**<**, **>**, **==**, etc) are the same as Python

Conditional Statements

- **Conditional** (if-else) statements in Python and Java are very similar

if elif else in Python:

```
if condition:
    statement1
    statement2
    ...
elif condition:
    statement1
    statement2
    ...
else:
    statement1
    statement2
    ...
```

Nested if else if in Java:

```
if (condition) {
    statement1;
    statement2;
    ...
} else if (condition) {
    statement1;
    statement2;
    ...
} else {
    statement1;
    statement2;
    ...
}
```

Java does not have an elif equivalent

Conditional Statements

Python:

```
a = 1
b = 2
if a < b:
    print("a < b")
```

```
a < b
```

```
if a > b:
    print("a > b")
else:
    print("a < b")
```

```
a < b
```

```
c = 3
if a > b and a > c:
    print("a is largest")
elif b > a and b > c:
    print("b is largest")
else:
    print("c is largest")
```

```
c is largest
```

Java:

```
int a = 1;
int b = 2;
if (a < b) {
    System.out.println("a < b");
}
```

```
a < b
```

```
if (a > b) {
    System.out.println("a > b");
} else {
    System.out.println("a < b");
}
```

```
a < b
```

Notice the && (logical AND) operator

```
int c = 3;
if (a > b && a > c) {
    System.out.println("a is largest");
} else if (b > a && b > c) {
    System.out.println("b is largest");
} else {
    System.out.println("c is largest");
}
```

```
c is largest
```

Today

- Discuss **loops** in Java
    - More if else statements, for loops, while loops
    - Review Python syntax as well!
- Begin discussing **methods** and **return** types in Java

Programming Language Features

- **Basic features:**
  - Data Types
  - Reading user input
  - Loops
  - Conditionals
- **Advanced topics:**
  - Classes
  - Interfaces
  - Collections
  - Graphical User Interface Programming

Loops

- We studied two different kinds of loops this semester
  - *Indefinite* loops (runs indefinitely until condition turns false)
    - **While loops**

```
while condition:
    # do something
```

  - *Definite* loops (runs a specific number of times)
    - **For loops**

```
for el in seq:
    # do something
```

- We'll look at both of these in Java

While Loops

- While loops in both languages are exactly the same (except for `(){}`)

Python:

```
num = 10
while num > 0:
    print(num)
    num = num // 2
```

Java:

```
int num = 10;
while (num > 0){
  System.out.println(num);
  num = num / 2;
}
```

When dividing Integers, Java automatically performs integer division. (No // in Java)

For Loops and Range Review

- Recall Python's **range** type: `range(start,stop,step)`
  - Example: `range(100,-1,-5)`
    - Start at 100, stop at -1, count backward by 5
  - Often use **range** object as part of for loop
- Java does not have a **range** data type
- Java's for loop syntax captures start and stop conditions *explicitly*

```
for (start clause; stop clause; step clause) {
    statement1;
    statement2;
    ...
}
```

- Let's look at a few examples

For Loops

- **Python** for loops allow you to iterate directly over any **iterable**
- **Java** syntax is a bit different and there is no range equivalent

for loops in Python:

```
for i in range(10):
    print(i)
    ...

for el in seq:
    print(el)
    ...
```

for loops in Java:

```
for (int i = 0; i < 10; i++) {
    System.out.println(i);
    ...
}

for (int i : myArray) {
    System.out.println(i);
    ...
}
```

Called a **for each** loop in Java

Lecture 6 Revisited: First for loop

- **Python** for loops also allow you to iterate directly over an **iterable**
  - Without using indices or knowing the length of the sequence
  - Recall this simple example from Lecture 6
  - Now we also know what happens behind the scenes

```
word = "Williams"
for char in word:
    print(char)
```

```
try:
    it = iter(word)
    while True:
        char = next(it)
        print(char)
except StopIteration:
    pass
```

Lecture 6 Revisited: First for loop

- Java **for each** loops internally use iterators just like Python and are equivalent to Python **for** loops (aside from data type complications)
- **for each** loops can easily iterate over arrays and Collections in Java

Python:

```
word = "Williams"
for char in word:
    print(char)
```

Java (for each):

```
String word = "Williams";
for (char c: word.toCharArray()) {
    System.out.println(c);
}
```

Lecture 6 Revisited: First for loop

- Java **for loops** explicitly use indices and specify the stopping condition (length of sequence) ahead of time
- In Python, we can rewrite our for loop as shown below to use indices, length, and range
- After rewriting, it will be easier to convert to Java

```
word = "Williams"
for char in word:
    print(char)
```

```
word = "Williams"
size = len(word)
for i in range(size):
    print(word[i])
```

Lecture 6 Revisited: First for loop

- Java **for loops** explicitly use indices and specify the stopping condition (length of sequence) ahead of time
- Once rewritten, we can convert to Java easily

Python:

```
word = "Williams"
size = len(word)
for i in range(size):
    print(word[i])
```

Java (for):

```
String word = "Williams";
int len = word.length();
for (int i = 0; i < len; i++) {
    System.out.println(word.charAt(i));
}
```

Same as `i += 1` or `i = i + 1`

countVowels

- Recall the `countVowels` function from Lecture 6 that combined for loops and conditionals
  - Notice that our docstring specifies input & output types of our function, but this is just *convention* in Python (not required)

```
def countVowels(word):
    """Takes a str word and returns
    a the number of vowels in it (int)"""
    count = 0
    for char in word:
        if char.lower() in "aieou":
            count += 1
    return count
```

`countVowels` in Java

- Writing the same method in Java

Return type of method specified in header

Takes parameter word of type `String` as input

```
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

`countVowels` in Java

- Writing the same method in Java

```
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

Initializing accumulation variable (specify type!)

`countVowels` in Java

- Writing the same method in Java

```
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

Define vowel string & compute length of word

`countVowels` in Java

- Writing the same method in Java

```
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

charAt returns a `char` (primitive type), no equivalent in Python

`String.valueOf(letter)` is like `str(letter)` in Python and converts `char letter` to a `String`

`countVowels` in Java

- Writing the same method in Java

```java
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

Similar to `s in vowels` in Python

`countVowels` in Java

- Writing the same method in Java

```
public static int countVowels(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (int i = 0; i < len; i++) {
      char letter = word.charAt(i);
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

Can also say `count += 1`

`countVowels2` in Java

- Writing the same method in Java **using a for each loop**

```
public static int countVowels2(String word){
  int count = 0;
  String vowels = "aeiou";
  int len = word.length();

  for (char letter : word.toCharArray()) {
      String s = String.valueOf(letter);
      if (vowels.contains(s)){
        count++;
      }
  }
  return count;
}
```

Vowels Class

```java
public class Vowels {

  public static int countVowels(String word){
    int count = 0;
    String vowels = "aeiou";
    int len = word.length();

    for (int i = 0; i < len; i++) {
        char letter = word.charAt(i);
        String s = String.valueOf(letter).toLowerCase();
        if (vowels.contains(s)){
          count++;
        }
    }
    return count;
  }

  public static void main (String args[]) {
    String word = "Williams";
    System.out.println(countVowels(word));
  }
}
```

Linear Search

- Recall our linear search in Python
- Let's implement it in Java! (with ints, chars, etc, and using both types of for loops)

```
def linearSearch(aList, item):
    n = len(aList)
    for el in aList:
        if item == el:
            return True
    return False
```

```java
public class LinearSearch {

  public static boolean doSearch(int array[], int elem) {
    int length = array.length;
    for (int i = 0; i < length; i++) {
      if(array[i] == elem) {
        return true;
      }
    }
    return false;
  }

  public static void main(String args[]) {
     int [] array = new int[] {4, 6, 9, 1, 3};

     System.out.println("4 in array?: "+doSearch(array, 4));
     System.out.println("2 in array?: "+doSearch(array, 2));

  }
}
```