Announcements & Logistics

• **HW 9** due Mon 12/5 @ 10pm
  - Covers “advanced” topics from recent lectures (Python special methods, iterators, efficiency, Java basics)

• **Lab 10 Selection Sort in Java** (next Mon/Tue)
  - No pre-lab work
  - Hope most of you will start and finish during your lab session

• **Final exam reminder: Friday, Dec 16 @ 9:30 AM**
  - Final is cumulative, emphasis on new material since midterm
  - You won’t have to write Java code
  - Study guide on Glow

• Course evals next Friday 12/9 (bring a laptop to class if possible)

*Do You Have Any Questions?*
Last Time

• Discussed high level overview of Java vs Python
• Focused on main differences:
  • Java is a **compiled** language: code is compiled before it is executed!
  • Java is **statically typed**: variables must be explicitly declared
• Looked at “Hello World” in Java
• Started discussing a simple example which takes input and converts Fahrenheit to Celsius

```java
public class Hello {
    public static void main(String args[]) {
        System.out.println("Hello, World!");
    }
}
```

```
terminal% javac Hello.java
terminal% java Hello
Hello, World!
```
Why Java?!

- **Review** Python concepts from the entire semester!
- Explore topics we've mentioned throughout the semester even deeper!
  - Data Types
  - Public/Private/Protected
  - Dynamic vs. Static Arrays
- See what concepts are general to **computer science**, which are limited to specific programming languages
- Good preparation for **CSCI 136**
Today's Plan

• Break down the simple temperature example further
• Move on to more interesting **data types** in Java
  • **Strings**
  • **ArrayLists**
  • **Arrays**
  • **HashMaps**
• Talk about **conditional statements**: very similar to Python!
Recap:
Python vs. Java
Recap: Simple Example in Python

```python
def main():
    fahr = input("Enter the temperature in F: ")
    cel = (float(fahr) - 32) * 5.0/9.0
    print("The temperature in C is:", cel)

if __name__ == "__main__":
    main()
```

- Asks user to enter a temperature in Fahrenheit and converts the string input to float
- Does the computation to convert temperature to Celsius
- Prints result
Simple Example in Java

```java
import java.util.Scanner;

public class TempConv {
    public static void main (String args[]) {
        Double fahr;
        Double cel;
        Scanner in;

        in = new Scanner (System.in);
        System.out.print("Enter the temperature in F: ");
        fahr = in.nextDouble ();

        cel = ( fahr - 32) * 5.0/9.0;
        System.out.println("The temperature in C is: " + cel);
    } 
}
```

- Same program in Java: TempConv.java
Review: Python vs. Java

Java:

```java
in = new Scanner(System.in);
System.out.print("Enter the temperature in F: ");
fahr = in.nextDouble();

cel = (fahr - 32) * 5.0/9.0;
System.out.println("The temperature in C is: "+ cel);
```

Python:

```python
fahr = input("Enter the temperature in F: ")
cel = (float(fahr) - 32) * 5.0/9.0
print("The temperature in C is:" , cel)
```

- Step 1: Prepare to read input from user.
Review: Python vs. Java

Java:
```
in = new Scanner (System.in);
System.out.print("Enter the temperature in F: ");
fahr = in.nextDouble();

cel = ( fahr - 32 ) * 5.0/9.0;
System.out.println("The temperature in C is: " + cel);
```

Python:
```
fahr = input ("Enter the temperature in F: ")
cel = (float(fahr) - 32) * 5.0/9.0
print ("The temperature in C is:" , cel)
```

• Step 2: Prompt user for input.
Review: Python vs. Java

Java:

```java
in = new Scanner(System.in);
System.out.print("Enter the temperature in F: ");
fahr = in.nextDouble();

cel = (fahr - 32) * 5.0/9.0;
System.out.println("The temperature in C is: " + cel);
```

Python:

```python
fahr = input("Enter the temperature in F: ")
cel = (float(fahr) - 32) * 5.0/9.0
print("The temperature in C is:" , cel)
```

- Step 3: Read user input and convert to float/double (that is, a number with a decimal point).
Review: Python vs. Java

Java:
```java
in = new Scanner (System.in);
System.out.print("Enter the temperature in F: ");
fahr = in.nextDouble();

cel = ( fahr - 32) * 5.0/9.0;
System.out.println("The temperature in C is: " + cel);
```

Python:
```python
fahr = input("Enter the temperature in F: ")

cel = (float(fahr) - 32) * 5.0/9.0
print("The temperature in C is:" , cel)
```

- Step 4: Perform conversion to Celsius.
Review: Python vs. Java

Java:

```java
in = new Scanner (System.in);
System.out.print("Enter the temperature in F: ");
fahr = in.nextDouble ();

cel = ( fahr - 32 ) * 5.0/9.0;
System.out.println("The temperature in C is: "+ cel);
```

Python:

```python
fahr = input ("Enter the temperature in F: ")
cel = (float(fahr) - 32 ) * 5.0/9.0
print ("The temperature in C is:", cel)
```

• Step 5: Print result.
An Aside: Java GUIs

- Java has more built-in support for making GUIs and supporting graphical objects
- Here is a graphical version of our program

```java
import javax.swing.*;

public class TempConvGUI {
    public static void main (String args[]) {
        Double fahr, cel;
        String fahrString;

        fahrString = JOptionPane.showInputDialog("Enter the temperature in F: ");
        fahr = Double.valueOf(fahrString);

        cel = (fahr - 32) * 5.0 / 9.0;
        JOptionPane.showMessageDialog(null, "The temperature in C is " + cel);
    }
}
```
Data Type:

Strings
Data Type: Strings

- Strings in Java and Python are similar, slightly different **syntax**
- Like Python, Java Strings are also **immutable**
- Java Strings **do not support an indexing or slicing operator**
- Instead of indexing, we create **substrings** in Java
- Java strictly uses **method calls** (no operators); Java does not support operator overloading in general

<table>
<thead>
<tr>
<th>Python</th>
<th>Java</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str[3]</td>
<td>str.charAt(3)</td>
<td>Return character in 3rd position</td>
</tr>
<tr>
<td>str[2:5]</td>
<td>str.substring(2,5)</td>
<td>Return substring from 2nd to 4th</td>
</tr>
<tr>
<td>len(str)</td>
<td>str.length()</td>
<td>Return the length of the string</td>
</tr>
<tr>
<td>str.find('x')</td>
<td>str.indexOf('x')</td>
<td>Find the first occurrence of x</td>
</tr>
<tr>
<td>str.split()</td>
<td>str.split(&quot; &quot;)</td>
<td>Split the string on whitespace into a list/array of strings</td>
</tr>
<tr>
<td>str.split(',' )</td>
<td>str.split(',' )</td>
<td>Split the string at ',' into a list/array of strings</td>
</tr>
<tr>
<td>str + str</td>
<td>str.concat(str)</td>
<td>Concatenate two strings together</td>
</tr>
<tr>
<td>str.strip()</td>
<td>str.trim()</td>
<td>Remove any whitespace at the beginning or end</td>
</tr>
</tbody>
</table>
Strings

Java:

```java
String s = "Almost winter break";

s.substring(0,3);  // Returns "Alm"

s.substring(4,7);  // Returns "st"
```

Python:

```python
s = "Almost winter break"

s[:3]  # Returns 'Alm'

s[4:7]  # Returns 'st '
```
Strings

**Java:**

```java
String s = "Almost winter break";

s.substring(0,3);
Alm
s.substring(4,7);
st
s.toUpperCase();
ALMOST WINTER BREAK
s.toLowerCase();
almost winter break
String [] array = s.split(" ");
System.out.println(Arrays.toString(array));
[Almost, winter, break]
```

**Python:**

```python
s = "Almost winter break"

s[:3]
'Alm'
s[4:7]
'st '
s.upper()
'ALMOST WINTER BREAK'
s.lower()
'almost winter break'
array = s.split()
print(array)
['Almost', 'winter', 'break']
```
Data Type: ArrayLists
Data Type: ArrayLists

- Java **ArrayLists** are roughly equivalent to Python **lists**
- Both are **dynamic arrays** (that grow and shrink in size automatically)
- Unlike Python where a list can contain anything, in Java we declare what type of objects our **ArrayList** is going to contain
- We **cannot use** [] operator in **ArrayLists**
  - Rely on **methods** (like `get()`, `set()`, `add()`) to manipulate the list
- Let’s compare **ArrayLists** to **lists** in Python
- We will also discuss Java **Arrays** which are also similar to Python **lists** but are statically-sized, more commonly used, and support [] operator
ArrayLists vs. Lists

Java:

```
ArrayList<String> alist = new ArrayList<String>();
alist.add("Jeannie");
alist.add("Rohit");
alist.add("Lida");
alist.add("Steve");
alist.add("Dan");
alist.add("Sam");
true
System.out.println(alist); // print the list
[Jeannie, Rohit, Lida, Steve, Dan, Sam]
```

Python:

```
alist = []
alist.append("Jeannie")
alist.append("Rohit")
alist.append("Lida")
alist.append("Steve")
alist.append("Dan")
alist.append("Sam")
print(alist)
['Jeannie', 'Rohit', 'Lida', 'Steve', 'Dan', 'Sam']
```
ArrayLists vs. Lists

Java:

```java
ArrayList<String> alist = new ArrayList<String>();
alist.add("Jeannie");
alist.add("Rohit");
alist.add("Lida");
alist.add("Steve");
alist.add("Dan");
alist.add("Sam");
true
System.out.println(alist); // print the list
[Jeannie, Rohit, Lida, Steve, Dan, Sam]
alist.add(3, "Iris"); // add Iris to index 3
System.out.println(alist);
[Jeannie, Rohit, Lida, Iris, Steve, Dan, Sam]
alist.get(2); // get the element at index 2
Lida
// set index 5 to Steve (returns old value)
alist.set(5, "Steve");
Dan
System.out.println(alist);
[Jeannie, Rohit, Lida, Iris, Steve, Steve, Sam]
```

Python:

```python
alist = []
alist.append("Jeannie")
alist.append("Rohit")
alist.append("Lida")
alist.append("Steve")
alist.append("Dan")
alist.append("Sam")
print(alist)
['Jeannie', 'Rohit', 'Lida', 'Steve', 'Dan', 'Sam']
alist.insert(3, "Iris")
print(alist)
['Jeannie', 'Rohit', 'Lida', 'Iris', 'Steve', 'Dan', 'Sam']
alist[2]
'Lida'
alist[5] = "Steve"
print(alist)
['Jeannie', 'Rohit', 'Lida', 'Iris', 'Steve', 'Steve', 'Sam']
```
Data Type:
Arrays
Data Type: Arrays

- An **array** is a primitive data structure in Java (with corresponding **Arrays** objectified class), and are also similar to **Lists**

- They do **support [ ] list notation**

- They **cannot dynamically shrink and grow**

- To declare a new array object in Java, we need to specify the **type** of its values and the **size** it will have
  
  - Size must be **specified directly**, or
  
  - Can just **assign values** at declaration

- Unlike lists in Python we cannot store heterogeneous types in an array!
Data Type: Arrays

- An **array** is a primitive data structure in Java.
- Can use list notation and assign values directly (but cannot grow or shrink).

```java
import java.util.Arrays;

public class Test {

    public static void main(String args[]) {
        int size = 10;

        // option 1: create an array directly
        int [] array1 = new int[] {2, 3, 5};

        // option 2: declare an with size then assign values
        int [] array2 = new int[3];
        array2[0] = 2;
        array2[1] = 3;
        array2[2] = 5;

        System.out.println(Arrays.toString(array1));
        System.out.println(Arrays.toString(array2));
    }
}
```
Java Arrays: More Examples

```java
import java.util.Arrays;

String [] myList = new String[6];

String [] myList = {"Jeannie", "Rohit", "Lida", "Steve", "Dan", "Sam"};

System.out.println(Arrays.toString(myList));
[Jeannie, Rohit, Lida, Steve, Dan, Sam]

System.out.println(myList[2]);
Lida

myList[4] = "Aaron";
Aaron

System.out.println(Arrays.toString(myList));
[Jeannie, Rohit, Lida, Steve, Aaron, Sam]
```

When declaring, either define size or give specific values. (Not necessary to do both!)

Java provides an array wrapper class called **Arrays** that provides convenient static methods for working with primitive arrays.

Can use list notation.

Can replace values, but can’t easily insert.

Print values of array.
Data Type: HashMaps
Other Data Types: Dictionaries

- **HashMaps** in Java are roughly equivalent to **Dictionaries** in Python
- Provide easy (O(1)) access to key, value pairs
- Provide convenient methods for interacting with the keys, values efficiently
- Require keys to be unique
- Java **HashMaps** do not support **[]** operator
  - Must use methods (like `put()`, `get()`, `containsKey()`) to manipulate **HashMap**
- Python **Dictionaries** support **[]** operator **and** methods
HashMaps vs. Dictionaries

Java:

```java
HashMap<Integer, String> csCourses;
csCourses = new HashMap<Integer, String>();
csCourses.put(237, "Computer Organization");
csCourses.put(134, "Intro to Computer Science");
csCourses.put(136, "Data Structures");
csCourses.put(256, "Algorithms");

csCourses.get(237);
Computer Organization

csCourses.get(134);
Intro to Computer Science

csCourses.containsKey(134);
true

false

csCourses.containsKey(361);
true

Python:

csCourses = dict()
csCourses[237] = "Computer Organization"
csCourses[134] = "Intro to Computer Science"
csCourses[136] = "Data Structures"
csCourses[256] = "Algorithms"

csCourses[237]
'Computer Organization'

csCourses.get(134)
'Intro to Computer Science'

134 in csCourses
True

361 in csCourses.keys()
False

"Data Structures" in csCourses.values()
True
Programming Language Features: Conditionals
Programming Language Features

- **Basic features:**
  - Data Types
  - Reading user input
  - Loops
  - **Conditionals**

- **Advanced topics:**
  - Classes
  - Interfaces
  - Collections
  - Graphical User Interface Programming
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
- Let’s consider three basic patterns

  1. Simple if in Python: 
     ```python
     if condition:
     statement1
     statement2
     ...
     ```

  2. Simple if in Java: 
     ```java
     if (condition) {
     statement1;
     statement2;
     ...
     }
     ```
Conditional Statements

- **Conditional** (if-else) statements in Python and Java are very similar
- Let’s consider three basic patterns

2. if else in Python:

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

if else in Java:

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

Note the use of () and {}
Conditional Statements

- **Conditional** (if-else) statements in Python and Java are very similar.
- Let’s consider three basic patterns.

3. if elif else in Python:

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

Nested if else if in Java:

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

Java does not have an elif equivalent.
Conditional Statements

Java:

```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

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
```

Python:

```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
```
The end!
Recall one of the first examples we looked at involving conditionals in Python (slightly modified to accept user input):

```python
def main():
    temp = float(input("Enter temp: "))
    if temp > 80:
        print("It is a hot one out there."")
    elif temp >= 60:
        print("Nice day out, enjoy!"")
    elif temp >= 40:
        print("Chilly day, wear a sweater."")
    else:
        print("Its freezing out, bring a winter jacket!"")

if __name__ == "__main__":
    main()
```
Lecture 5 Revisited

• Let’s write it in Java!
Lecture 5 Revisited

```java
import java.util.Scanner;

public class WeatherFinal {
    public static void main(String args[]) {
        double temp;
        Scanner in;

        in = new Scanner(System.in);
        System.out.print("Enter temp: ");
        temp = in.nextDouble();

        if (temp > 80) {
            System.out.println("It is a hot one out there.");
        } else if (temp >= 60) {
            System.out.println("Nice day out, enjoy!");
        } else if (temp >= 40) {
            System.out.println("Chilly day, wear a sweater.");
        } else {
            System.out.println("Its freezing out, bring a winter jacket!");
        }
    }
}
```

Could use `Double` here as well.