CS230: Data Structures

Spring 2018

Announcements

- Join the course group CS230-S18!
  - Post questions, answer them if you know the answer!
- Assignment 1 is available and due at 11:59pm Wednesday February 7th
  - See schedule for link to assignment description
- Reading for next class
  - Java Foundations, by Lewis, DePasquale and Chase, 2nd edition. (Denoted as LDC on the schedule)
  - See Chapters 1 and 2 for more information about todays topics
  - Read Chapters 3 and 4 (3.6 and 3.7 are optional)

Course Website

- http://cs.wellesley.edu/~cs230

Collaboration Policy
Why take CS230?

- You will learn the “big picture” of programming
  - Data abstraction
  - Modularity
  - Performance Analysis
  - Basic abstract data types (ADTs)
- You will become a more competent programmer
  - You will also become a designer, tester, analyzer, debugger, team member
- You will develop a project worth showing off
- You will have fun in the process!

Why use ADTs?

- Allows you to write complex programs more easily
  - To keep mental track of complex data interaction
  - To reuse code
  - To improve code performance
- Allows modularity of large projects
  - Easier to understand large chunks of code
  - Easier to collaborate with large teams

Basic ADTs
- Collections
- Linked List
- Stack
- Queue
- Hash Table
- Priority Queue

Less basic:
- Tree
- Set
- Graph

First Java Program

```java
/**
 * First program for CS230
 * @author Ashley DeFlumer
 */
public class Welcome {
    public static void main (String[] args) {
        System.out.println("Hello and Welcome to CS230!");
    }
}
```

A First Program

```java
/**
 * Our first CS230 program.
 * It prints out Wellesley’s motto.
 * @author Orit Shaer
 */
public class Motto {
    public static void main(String[] args) {
        System.out.println("Non ministrari");
        System.out.println("sed ministrare");
    }

    // Program execution begins with the "main" method
    public static void main(String[] args) {
        System.out.println("Non ministrari");
        System.out.println("sed ministrare");
        System.out.println("Hello and Welcome to CS230!");
    }
}
```

Multi line JavaDoc comment

Curly braces, not whitespace, denote structure

System.out.println similar to Python print function

Statements end with semicolons

Multi-line JavaDoc comment

A public class must be in a Java file with the same name

A Java “method” is similar to a Python “function”

// Program execution begins with the “main” method
public static void main(String[] args) {
    System.out.println("Non ministrari");
    System.out.println("sed ministrare");
}

Curly braces, rather than indentation, indicate the body of classes, methods, loops, and conditionals

String denoted by double quotes

Statements end with semicolons

A public class must be in a Java file with the same name

A Java “method” is similar to a Python “function”

// Program execution begins with the “main” method
public static void main(String[] args) {
    System.out.println("Non ministrari");
    System.out.println("sed ministrare");
    System.out.println("Hello and Welcome to CS230!");
}

Curly braces, rather than indentation, indicate the body of classes, methods, loops, and conditionals

String denoted by double quotes

Statements end with semicolons
Using Java and BlueJ

- You can study **data structures** using *any language*, but in this course we will use **Java**.
- You can **write and execute Java programs** in many ways, on the command line or with an IDE, but in this course we will use a simple IDE called **BlueJ**.

![Java IDE](image)

Data Types in Java

- Java is a **statically typed** language.
  - You must explicitly define the type of each variable when it is declared.
  - Unlike Python, not all variables in Java are objects.
  - Some are **primitive data types**.

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Integer</td>
</tr>
<tr>
<td>float</td>
<td>Float</td>
</tr>
<tr>
<td>double</td>
<td>Double</td>
</tr>
<tr>
<td>char</td>
<td>Char</td>
</tr>
<tr>
<td>boolean</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Variable Declaration in Java

- A variable must be declared once before it can be used.
- The type of a variable cannot be changed after declaration.
- The value of a variable can be changed any number of times.

```java
int x;
int y;
int z;
x = 7;
y = 5;
z = x + y;
System.out.println(z);
```

Variables declared and initialized in separate statements.

```java
int x = 7;
int y = 5;
int z = x + y;
System.out.println(z);
```

Variables declared and initialized in single statement.

Decimal Numbers

```java
double num = 5.2;
num = 1.4;
num = num * 2.0;
System.out.println(num);
```

```java
double fahrenheit = 98.6;
double celsius = (fahrenheit - 32) * 5 / 9;
System.out.println(celsius);
```

```java
double fahrenheit = 98.6;
double celsius = (fahrenheit - 32) * 5 / 9;
System.out.println(celsius);
```
**String Concatenation**

```java
String course = "Data Structures";
System.out.println(course);

String s1 = "CS230 is ";
String s2 = "fun";
String fact = s1 + s2;
System.out.println(fact);

double temp = 39.2;
System.out.println("It is " + temp + " degrees");
```

**Strings in Java**

- Strings in Java and Python are quite similar.
- Like Python, Java strings are immutable.
- The difference is that Java uses method calls where Python uses Operators.

<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(2,4)</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('s')</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>

**Find the Errors!**

```java
// This program has at least 5 errors. Can you find them all?
public class Errors {
    public static void main(String[] args) {
        String temperature = 80.3;
        int n = 100;
        n = "Wait, what?";
        print("This is fine.");
    }
}
```
Choose your own adventure...

- Find a group and a board

- Write your own Java program to calculate some value and print it out

- Some ideas (or make your own!):
  - Area of a circle (or other shapes) given its radius (or other necessary dimensions)
  - Volume of a box/sphere/cylinder of some given dimensions
  - Simple interest given amount, rate, time