CSE 310 – Java Workshop

Example Classroom Code

- Starting Code: https://replit.com/@cmacbeth/CSE310JavaWorkshop
- Solution Code: https://replit.com/@cmacbeth/CSE310JavaWorkshopSolution

Useful Reference Links

- https://www.w3schools.com/java/
- https://docs.oracle.com/en/java/javase/16/docs/api/index.html

Development Environment

There are two ways to install Java:

- Oracle JDK
  - Requires license for commerical use
  - https://www.oracle.com/java/technologies/javase-downloads.html
- OpenJDK
  - Open Source - GNU Public License
  - https://openjdk.java.net/
  - Does not have an installation. Uncompress the zip file and put the folder where you want it.

Java code is compiled using the `javac` tool. Look in the `bin` directory of your java installation for this tool. If you are running from the command line, you will want to add this to your environment path. This example compiles three `.java` files into `.class` files.

```
javac hello.java goodbye.java start.java
```

The `.class` files are object code which is interpreted using the java virtual machine. After compiling into `.class` files, the software can now be executed on any operating system. When you download Java, you have to select the download that matches your operating system so that the `java` tool will interpret the `.class` files correctly. To run your code you use the `java` tool.

```
java Start.class
```

In your Java code, one of the classes will have a `main` function that defines where the code begins. In our example above, the `Start` class has the `main` function. When running the `java` tool, we specify the class that has the `main` function to run.

Java Syntax

Classes

In Java, everything is a class. Code does not exist without a class.

```java
public class Hello {

}
```

At least one of your classes must have a static `main` function to describe how the program should begin.

```java
public class Hello {

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

A class can have attributes and functions. All parts of a class can be declared with scope access of either:

- public - Accessible by any class
- private - Accessible only in the class
- protected - Accessible only in the class or derived classes (inheritance)
- If no access is given, then this is called "Package Scope" which is accessible by all classes in the package (or folder)

A class should also have a constructor to initialize attributes in the class.

```java
public class Box {
    private float length;
    private float width;

    public Box(float length, float width) {
        this.length = length;
        this.width = width;
    }

    public float getArea() {
        return length * width;
    }
}
```

A useful function to override in Java is the `toString` function. This function is used to represent the object as a string.

```java
public class Box {
    ...
    public String toString() {
        String result = "Box[w=" + width + ",l=" + length + "]";
        return result;
    }
}
```

Objects and Memory

All objects of classes are dynamically allocated on the heap. Java uses garbage collection to free unused memory. The `new` operator is used to create objects.

```
Box box1 = new Box(3.2f, 1.8f);
Box box2 = new Box(1.2f, 4.2f);
float totalArea = box1.getArea() + box2.getArea();
```

When you pass an object to a function, it passes a copy of the memory addresses. This means that the receiving function can affect changes on the object (e.g. by calling a funciton on it).

Data Structures

The Java API is very extensive including support for things such as data structures, networking, files, and graphics. Common data structures include:

- ArrayList - Implements a dynamic array with support of indicies
- LinkedList - Implements a doubly linked list useful for queues
- HashMap - Implements a key/value pair lookup table

When creating a data structure object, Java requires that you specify that data type that it will hold. This uses a syntax technique called templates.

```
ArrayList<Box> boxes = new ArrayList<>();
boxes.add(new Box(9.9f, 1.1f));
boxes.add(new Box(7.2f, 8.9f));
boxes.add(new Box(2.4f, 3.0f));
```

Loops

Java supports counter based loops using the `for` statement.

```
int sum = 0;
for (int i=1; i<=10; i++) {
    sum += i;
}
```

Java also supports iterators for data structures.

```java
float totalArea = 0.0f;
for (Box b : boxes) {
    totalArea += b.getArea();
}
```