Object Oriented Programming
LAB 3
PRE LAB SHEET

Read about static variables and method overloading from textbooks T1 and T2 given in the handout.

In this lab sheet,we will learn about overriding the hashCode() and equals() method.

Consider the following class definition-

```
public class Student {
   private int id;
   private String name;
   public Student(int id, String name) {
       this.name = name;
       this.id = id;
   }
   public int getId() {
       return id;
   }
   public void setId(int id) {
       this.id = id;
   }
   public String getName() {
       return name;
   }
   public void setName(String name) {
       this.name = name;
   }
}
```

We now introduce a function `equals()` which compares an object passed to the function and checks if it is equal to the given object instance. This method is implemented in the class `Object` from which every class “inherits”, more about which you will learn later. For now, we will learn how to “override” the implementation of `equals()` in `Object` class with our own implementation.

```
@Override     //Indicating that parent class method is overridden.
public boolean equals(Object obj) {

        // checking if both the object references are
        // referring to the same object.
        if(this == obj)
            return true;

        // it checks if the argument is of the
        // type Student by comparing the classes
        // of the passed argument and this object.

        if(obj == null || obj.getClass()!= this.getClass())
            return false;

        // type casting of the argument.
        Student student = (Student) obj;

        // comparing the state of argument with
        // the state of 'this' Object.
        return (this.name == student.name && this.id == student.id);

}
```

Similarly , let us look at that `hashCode()` function, again a method implemented in the `Object` class which we will override.
If two objects are equal, according to the the `equals(Object)` method, then `hashCode()` method must produce the same integer on each of the two objects.
We use the following standard algorithm while calculating the hashcode, in which we make use of `hashCode()` values of each of the fields. For example-

```
@Override
public int hashCode() {

        int result = 17;
        result = 37 * result + name.hashCode() //calls the hashCode() method of string class
        result = 37 * result + id.hashCode(); //calls the hashCode() method of int class
        return result;

}
```

Further details will be explored in lectures when inheritance will be taught. However, **Kindly look up and read these links for further reference-**

https://docs.oracle.com/javase/7/docs/api/java/lang/Object.html#equals(java.lang.Object)

https://docs.oracle.com/javase/7/docs/api/java/lang/Object.html#hashCode()

https://howtodoinjava.com/java/basics/working-with-hashcode-and-equals-methods-in-java/ (Before 3) EqualsBuilder and HashCodeBuilder from Apache Commons)

https://stackoverflow.com/questions/2265503/why-do-i-need-to-override-the-equals-and-hashcode-methods-in-java