Prelab Inner Classes

Sarthak Moorjani, Shubhi Jain, Aviral Agrawal

August 2018

1 General Instructions

- **READ THE QUESTION PROPERLY BEFORE ATTEMPTING OR CALLING TA's FOR UNNECESSARY DOUBTS.**
- **DO NOT COMPILE THE CODE ON AUTOLAB. ALL THE LOGS OF SUBMISSION WILL BE THOROUGHLY CHECKED AND ANY COMPILATION ERROR ON AUTOLAB MAY RESULT IN UNDESIRABLE SITUATIONS.**
- In the lab, you will have to make 2 outer classes which may or may not have inner classes. The documentation will contain all the information that is required to attempt the lab.
- It is advised that you should have a thorough knowledge about inner classes before you come to the lab. The concepts should be crystal clear. We will not help you with the fundamentals of inner classes.
- The lab is more about clearing your concepts and not fetching marks. Attempt the lab accordingly.

2 About Inner Classes

- An object of an inner class has a link to the enclosing object that made it, and so it can access the members of that enclosing object—without any special qualifications. In addition, inner classes have access rights to all the elements in the enclosing class.
- It's not possible to create an object of the inner class unless you already have an object of the outer class. This is because the object of the inner class is quietly connected to the object of the outer class that it was made from. However, if you make a nested class (a static inner class), then it doesn't need a reference to the outer-class object.

The following example illustrates some basic concepts about inner classes.

```
public class Parcel2 {
  class Contents {
    private int i = 11;
    public int value() { return i; }
  }
  class Destination {
    private String label;
    Destination(String whereTo) {
      label = whereTo;
    }
    String readLabel() { return label; }
  }
  public Destination to(String s) {
    return new Destination(s);
  }
  public Contents contents() {
    return new Contents();
  }
  public void ship(String dest) {
    Contents c = contents();
    Destination d = to(dest);
    System.out.println(d.readLabel());
  }
  public static void main(String[] args) {
    Parcel2 p = new Parcel2();
    p.ship("Tasmania");
    Parcel2 q = new Parcel2();
    // Defining references to inner classes:
    Parcel2.Contents c = q.contents();
    Parcel2.Destination d = q.to("Borneo");
  }
}
 Output:
Tasmania
```

- Use of **.new** keyword

```
    public class DotNew {
    public class Inner {}
    public static void main(String[] args) {
    DotNew dn = new DotNew();
    DotNew.Inner dni = dn.new Inner();
  }
}
```

- **Multiple Inner Classes:** It doesn't matter how deeply an inner class may be nested—it can transparently access all of the members of all the classes it is nested within, as seen here:

```
    class MNA {
    private void f() {}
    class A {
    private void g() {}
    public class B {
      void h() {
        g();
        f();
      }
    }
  }
}
public class MultiNestingAccess {
  public static void main(String[] args) {
    MNA mna = new MNA();
    MNA.A mnaa = mna.new A();
    MNA.A.B mnaab = mnaa.new B();
    mnaab.h();
  }
}
```

- **Method Overloading** Methods of outer class can be overloaded in inner classes. The following example shall clear this concept further.

```
    public class Outer {
    public void foo() {
        System.out.println("In Outer");
    }
    class Inner {
        public void foo() {
            System.out.println("In Inner");
        }
    }
    public static void main(String [] args) {
        Outer outer = new Outer();
        Outer.Inner inner = outer.new Inner();
        inner.foo();
        outer.foo();
    }
}
Output:
In Inner
In Outer
```

3 Some Information About the Lab

- We will be continuing our design based on shapes only. Our campus placements entered another dimension this year, so we also introduce another dimension :P Let's go 3D!
- A house is a 3 dimensional object which is made of shapes :D We want you to design a small little house which has only 1 floor made up of few rooms (cuboid) (which indeed are made up of walls(Rectangles)) and a roof. Since we are familiar with the magnificent rains in Goa, we decided to make the roof of the house pyramidal with a rectangular base and triangular walls.
- The house can be likened to a big cuboid filled with smaller cuboids(rooms).
- Further, we want you to design a small colony of houses too.
- More on this on Tuesday :)

4 References

Bruce Eckel, Chapter on Inner Classes, Thinking in Java, PHI, 4th Edition, 2006.

******ALL THE BEST******