Pencil and Paper Assignment for Lesson 7

_____ 1. What happens when the following is compiled/run?

```
class MyClass {
   public static void main(String[] args) {
      new MyClass();
   }
   MyClass() {
      recurse("Hello");
   }
   String recurse(String s){
      if(s==null) return null;
      int r = RandomNumbers.getRandomInt();
      int n = s.length();
      if(r % 2 == 0)
         return recurse(s.substring(0,n/2));
      else {
         return recurse(s.substring(n/2,n));
      }
   }
}
```

a. Compiler error
b. Returns a null value
c. NullPointerException
d. StackOverflowError

Explain your answer

_____ 2. What happens when the following is compiled/run? You may assume that the method permute is implemented correctly elsewhere, and that it has the effect of randomly rearranging the characters of a String (for instance, on different runs of permute with input “events”, the return values could be, for example, “evtsen”, “eestnv” and “evenst”).

```
class MyClass {
   public static void main(String[] args) {
      new MyClass();
   }
   MyClass() {
      recurse("Hello");
   }
   String recurse(String s){
      if(s==null || s.equals("")) return "";
      int n = s.length();
      String t = permute(s); //rearrange characters of s
      return recurse(t);
```

```
    }
}
```

a. Compiler error
b. Returns a null value
c. NullPointerException
d. StackOverflowError

Explain your answer

3. When you run the following code, you discover that the version of the `clone()` method that the class `Inner` uses in its `innerMethod()` is the one in `Object`, not the version in the enclosing class. Show (by doing) how to modify the code inside `innerMethod()` to force the code to use the `clone()` method of `Enclosing`. Verify (in Eclipse) that your solution is correct. (The code shown below is provided in the folder for this lab.)

```
public class Enclosing implements Cloneable {
   public Enclosing clone() throws CloneNotSupportedException {
         System.out.println("Inside Enclosing.clone()");
         return (Enclosing)super.clone();
   }
   class Inner implements Cloneable{
         void innerMethod() throws CloneNotSupportedException {
               Object copy = clone();
               System.out.println(copy.getClass().getName());

         }
   }
   public static void main(String[] args){
         Enclosing p1 = new Enclosing();
         Enclosing.Inner i1 = p1.new Inner();
         try {
               i1.innerMethod();
         }
         catch(CloneNotSupportedException e){
               e.printStackTrace();
         }
   }

}
```

3. When you run the following code, you discover that the version of the `clone()` method that the class `Inner` uses in its `innerMethod()` is the one in `Object`, not the version in the enclosing class. Show (by doing) how to modify the code inside `innerMethod()` to force the code to use the `clone()` method of `Enclosing`. Verify (in Eclipse) that your solution is correct. (The code shown below is provided in the folder for this lab.)

```
public class Enclosing implements Cloneable {
   public Enclosing clone() throws CloneNotSupportedException {
         System.out.println("Inside Enclosing.clone()");
         return (Enclosing)super.clone();
   }
   class Inner implements Cloneable{
         void innerMethod() throws CloneNotSupportedException {
               Object copy = clone();
               System.out.println(copy.getClass().getName());

         }
   }
   public static void main(String[] args){
         Enclosing p1 = new Enclosing();
         Enclosing.Inner i1 = p1.new Inner();
         try {
               i1.innerMethod();
         }
         catch(CloneNotSupportedException e){
               e.printStackTrace();
         }
   }

}
```