5COSC019W - Solutions to Tutorial 3 Exercises

1 Implementing Constructors

```
class VendingMachine {
    private int numCans;
    private int tokens;


    public VendingMachine() {
        numCans = 10;
        tokens = 0;
    }

    public VendingMachine(int cans) {
        numCans = cans;
        tokens = 0;
    }

    public void fillUp(int cans) {
        numCans = numCans + cans;
    }

    public void insertToken() {
       numCans = numCans - 1;
       tokens = tokens + 1;
    }

    public int getCanCount() {
        return numCans;
    }

    public int getTokenCount() {
        return tokens;
    }
}
```

2 Implementing a Class

```
class Employee {
    String name;
```

```
    double salary;

    public Employee() {
    }

    public Employee(String n, double sal) {
        name = n;
        salary = sal;
    }

    public String getName() {
        return name;
    }

    public double getSalary() {
        return salary;
    }
}

public class EmployeeTest {
    public static void main(String[] args) {
        Employee e1 = new Employee();
        Employee e2 = new Employee("John Jones", 35000);
        Employee e3 = new Employee("Ian Smith", 20000);

        System.out.println("e1: " + e1.getName() + ", salary: " + e1.getSalary());
        System.out.println("e2: " + e2.getName() + ", salary: " + e2.getSalary());
        System.out.println("e3: " + e3.getName() + ", salary: " + e3.getSalary());
    }
}
```

When the program is run, it displays:

```
e1: null, salary: 0.0
e2: John Jones, salary: 35000.0
e3: Ian Smith, salary: 20000.0
```

3 Designing and Implementing a Class

1. ```
   public class VotingMachine {
       private int labourVotes;  // number of votes for Labour party
       private int conservativeVotes;  // number of votes for Conservatives party

       public void voteLabour() {
           ++labourVotes;
       }

       public void voteConservatives() {
   ```
   ```

```
        ++conservativeVotes;
    }

    public int getLabourVotes() {
        return labourVotes;
    }

    public int getConservativeVotes() {
        return conservativeVotes;
    }

    // clear the state of the voting machine
    public void clear() {
        labourVotes = 0;
        conservativeVotes = 0;
    }
}
```

The implemented class can be tested using the following test class:

```
public class VotingMachineTest {
    public static void main(String[] args) {
        VotingMachine vMachine = new VotingMachine();

        for (int i=1; i <= 1000; i++) {
            /* flip a coin what to vote for - vote for each party
               with 50% probability */
            double flip = Math.random();
            if (flip <= 0.5)
                vMachine.voteLabour();
            else
                vMachine.voteConservatives();
        }

        // print results
        System.out.println("Election results");
        System.out.println("----------------");

        System.out.println("Labour received: " + vMachine.getLabourVotes());
        System.out.println("Conservatives received: " +
                                            vMachine.getConservativeVotes());
    }
}
```

2. The biased version of the `VotingMachine` (biased towards Labour) is:

```
// Biased (unfair) version of voting machine
public class VotingMachine {
    private int labourVotes;  // number of votes for Labour party
    private int conservativeVotes;  // number of votes for Conservatives party
```

```
    public void voteLabour() {
        ++labourVotes;
        labourVotes += (int) 3*Math.random();
    }

    public void voteConservatives() {
        ++conservativeVotes;
    }

    public int getLabourVotes() {
        return labourVotes;
    }

    public int getConservativeVotes() {
        return conservativeVotes;
    }

    // clear the state of the voting machine
    public void clear() {
        labourVotes = 0;
        conservativeVotes = 0;
    }
}
```

4 Objects are Copied by Reference

The output of the program is:

```
After mutate() is called, colour of cat c1 is: Pink
After creation of cat c2
Memory address of object c1 is: Cat@923e30
Memory address of object c2 is: Cat@130c19b


After assignment c2 = c1;
Memory address of object c1 is: Cat@923e30
Memory address of object c2 is: Cat@923e30


After c2.setColour("Yellow")
Colour of c1 is: Yellow
Colour of c2 is: Yellow

Inside Cat.create(), address of created cat object is: Cat@1f6a7b9
Address of c3 is: Cat@1f6a7b9
```

5 Object Comparison

When the `new` keyword is not used to create a `String` object then Java is using a pool of constant strings to optimise space. This means a new object will not be re-created but it will be reused from the pool of Java strings.

6 The null keyword

During execution an exception occurs:

```
Exception in thread "main" java.lang.NullPointerException
        at NullObjectsTest.main(NullObjectsTest.java:4)
```

A reference variable must point to an object before attempting to access fields or call methods on the object. Since `s` contains the value `null`, method `toUpperCase()` cannot be called on `null` and an exception will be generated during running the program.

7 Overloading Methods

1.
```
public Car(String licensePlate1, double maxSpeed1, double speed1) {
          this.licensePlate = licensePlate1;
          this.speed  = 0.0;
          if (maxSpeed1 >= 0.0) {
              maxSpeed = maxSpeed1;
          }
          else {
              maxSpeed = 0.0;
          }

          // set speed according to the value of passed argument
          if (speed1 < 0)
              speed = 0;
          else if (speed1 > maxSpeed)
              speed = maxSpeed;
          else
              speed = speed1;
  }
```

2.
```
      // print the details of the car
      public void print() {
          System.out.println("Current speed: " + speed + "\nMax speed: " +
                             maxSpeed + "\nLicence: " + licensePlate);
      }
```

3.
```
public class CarTest {
      public static void main(String[] args) {
```

```
        Car car1 = new Car("K123WMI", 150.0);
        car1.print();

        Car car2 = new Car("L777ALA", 120.0, 80.0);
        car2.print();

        Car car3 = new Car("F180AST", 180.0, 40.0);
        car3.print();
    }
}
```

8 Challenge: Designing and Implementing Classes - The US Postal Service

This is an optional challenge exercise. If you attempt this and if you have any doubts about your solution, you could show this to your tutor.