Question 1. [17 MARKS]

Below is a partially completed class modelling cars. Read it carefully. For this question, you will write code calling methods of the class, and also write additional methods for the class.

```
public class Car {
    private String maker;            // Chrysler, Ford, Toyota, etc.

    private double kmPerLitre;      // the fuel efficiency: how far
                                    // it can go on one litre of gas

    private double litresOfGas;     // gas currently in the tank
    private int tankSize;           // the maximum the tank can hold

    public Car(String m, double k, int t) {
        maker = m;
        kmPerLitre = k;
        tankSize = t;
    }
} // class Car
```

Part (a) [3 MARKS]

In a *single* java statement: construct an object representing a Toyota that has a 50 litre gas tank and gets 15.5km per litre, and declare a reference `c1` to this object.

SOLUTION:

```
Car c1 = new Car("Toyota", 15.5, 50);
```

MARKING SCHEME:

COMMENTS:

Part (b) [3 MARKS]

Write a method `fillErUp()` for the `Car` class, which sets the amount of fuel currently in the tank to the maximum amount.

SOLUTION:

```
public void fillErUp() {
    litresOfGas = tankSize;
}
```

MARKING SCHEME:

COMMENTS:

Part (c) [3 MARKS]

Write a method `fuelInTank()` that could be called as demonstrated in the line of code

```
double litres = c1.fuelInTank();
```

and that does what its name suggests.

SOLUTION:

```
public double fuelInTank() {
    return litresOfGas;
}
```

Marking Scheme:

Comments:

The next 2 parts are about a method `drive()`. The parameter for this method indicates the number of kilometers the user hopes to drive. If there is enough gas for the trip, the method removes that much gas and returns `true`. If there is not enough gas, the method uses up all the gas and returns `false`.
Here is an example of a call to the method. In this example, `c1` is a `Car` reference to an existing `Car` with some gas.

```
if (c1.drive(140.2)) {
    System.out.println("We made it to London!");
} else {
    System.out.println("We ran out of gas!");
}
```

Part (d) [2 marks]

Write the prototype for `drive()`.

Solution:

```
public boolean drive(double kms)
```

Marking Scheme:

Comments:

Part (e) [6 marks]

Write the body of `drive()`.

Solution:

```
double gasNeeded = kms / kmPerLitre;
if (litresOfGas >= gasNeeded) {
    litresOfGas -= gasNeeded;
    return true;
} else {
    litresOfGas = 0;
    return false;
}
```

Marking Scheme:

Comments:

Question 2. [12 marks]

The following is a Java program and the memory model that results after executing statement 1 in the `main` method. Update the diagram to show the state of memory after statement 1 in method `four` executes for the first time (this is initiated by the call in statement 3 of method `main`.) As in Assignment 2, you may *cheat* by representing `String`s as text inside variable boxes.

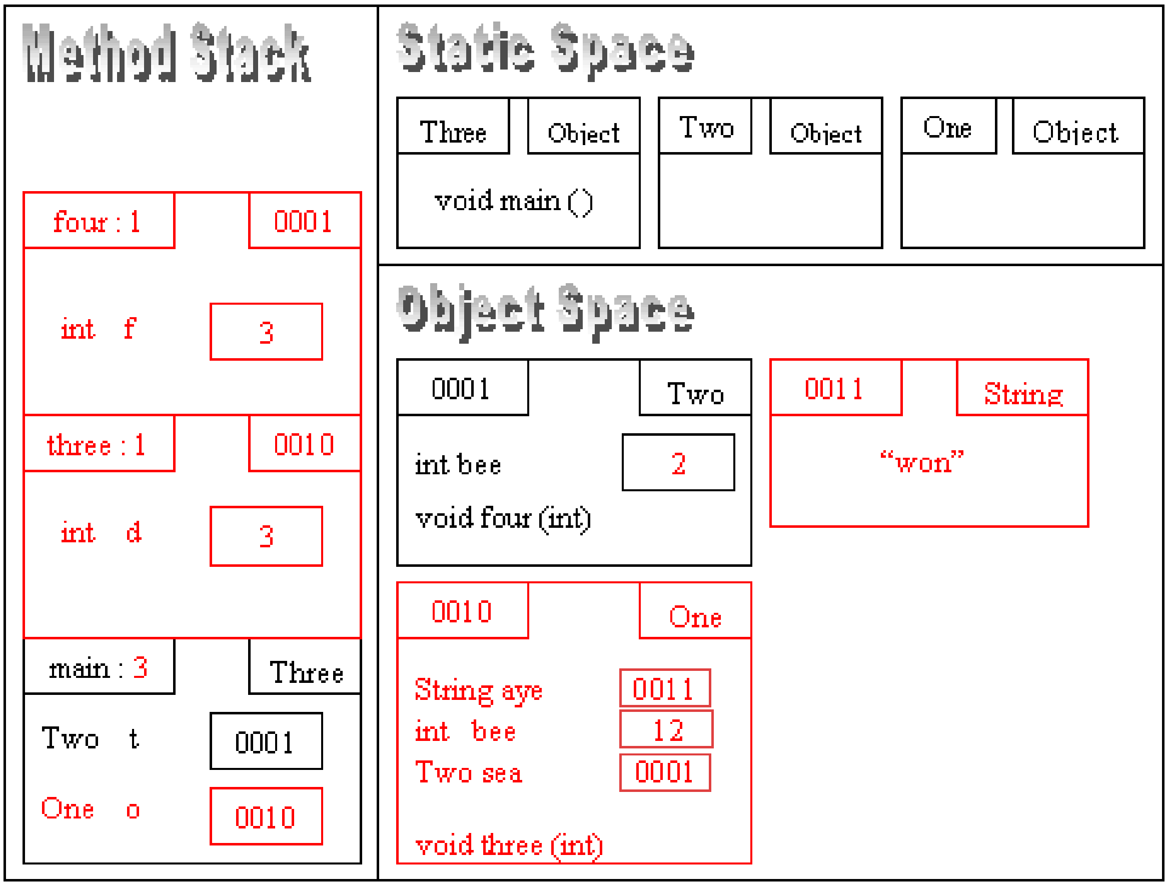
```
class One {                                  |    class Two {
   private String aye;                       |        private int bee;
   private int bee;                          |        public Two(int d) {
   private Two sea;                          |              bee = d;
   public One(String a, int b, Two c) {      |        }
       aye = a;                              |        public void four(int f) {
       bee = b;                              |            bee -= f;         // stmt 1
```

```
        sea = c;
    }
    public void three(int d) {
        sea.four(d);        // stmt 1
    }
} // class One
```

```
            int dee = 5;        // stmt 2
        }
    } // class Two

    public class Three {
        public static void main(String[] args) {
            Two t = new Two(5);             // stmt 1
            One o = new One("won",12,t);    // stmt 2
            o.three(3);                     // stmt 3
        }
    } // class Three
```

Solution:



Marking Scheme:

Comments:

Question 3. [14 marks]

Here is a Java program, which compiles and runs without error messages. Write the output from the program in the space below it.

```
public class Q3 {
    public static void main(String[] args) {
        String s = "Alan Turing";
        String t = s;
```

```
        System.out.println(s.substring(1, 3));
        s = s.toUpperCase() + " " + t.toLowerCase();
        System.out.println(t);
        mister(s);
        System.out.println(s);
        Person p1 = new Person("Alan Turing", "Student");
        Person p2 = new Person("Alan Turing", "Computer Scientist");
        Person p3 = p1;
        p3.changeOccupation("Secret Agent");
        p1.printOccupation();
        p2.printOccupation();
        p3.printOccupation();
    }
    public static void mister(String s) {
        s = "Mr. " + s;
        System.out.println(s);
    }
}

public class Person {
    private String name;
    private String occupation;
    public Person(String n, String o) {
        name = n;
        occupation = o;
    }
    public void changeOccupation(String o) {
        occupation = o;
    }
    public void printOccupation() {
        System.out.println(occupation);
    }
}
```

SOLUTION:

```
la
Alan Turing
Mr. ALAN TURING alan turing
ALAN TURING alan turing
Secret Agent
Computer Scientist
Secret Agent
```

MARKING SCHEME: 2 marks per line of output. Output that is close but not exact will usually result in 0 marks because it means they don't know what caused the output. If they are off by one with the `substring` call, give them 0.5 marks. If they think `s` changed due to the `substring` but are consistent in using this afterwards, only penalize them once.

COMMENTS: