IB Diploma Program Computer Science

Semester 2 Internal Test

Name _____

Part A - Option D Questions
Part B - Topic 3

37

PART A - Option D : Recommended Time 50 Mins

A Bus has up to 10 Passengers. To simulate this in a computer program, an array big enough for 10 Passenger objects will be used. Indexes in the array will be used as seat numbers (ie 0-9).

The UML diagrams for the Bus and Passenger classes are shown below:

Passenger
- String name - String nationality - int seatNum
+ Passenger(String, String, int) + getName():String + getNationality():String + getSeatNum():int + setSeatNum(int):void

Bus
- String regNum - Passenger[] passengers - boolean lowFuel
+ Bus(String) + setFuel():void + addPassenger(Passenger):void + getPassengerName(int):String + getPassengers(): Passenger[ ] + getLowFuel(): boolean + getPassengerCount():int

a) Complete the UML diagram by showing the relationship between the 2 classes. [1]

b) The `getNationality` method of the `Passenger` class returns the nationality of the `Passenger`. Write code to implement the `getNationality` method of the`Passenger` class.

[2]

c) The `getNationality` method of the `Passenger` class is an accessor method. Explain the purpose of an accessor method.

[1]

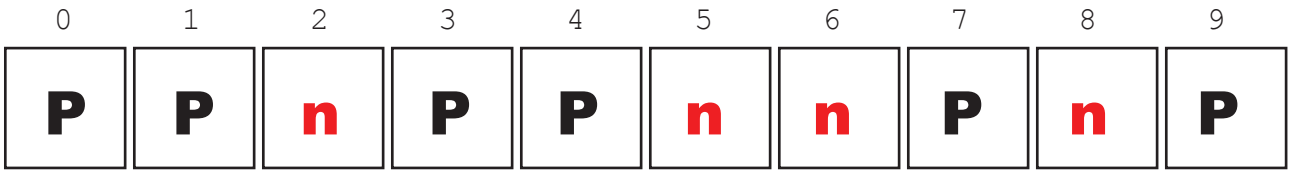
d) The `Bus` class constructor method is designed as follows:

```
public Bus(String regNum) {

        this.regNum = regNum;
        this.passengers = new Passenger[10];
        this.lowFuel = false;
}
```

The `passengers` property of the `Bus` class is initialised as an array of 10 nulls.

The `addPassenger` method of the `Bus` class (see UML diagram) will attempt to add a new `Passenger` at the given seat number. For example, a Bus instance, `bus1`, exists and this is the current state of `passengers`:



P = instance of Passenger class, n = null

then if a program attempted to do this:

```
bus1.addPassenger(new Passenger("Jack Li", "USA", 5));
```

then `bus1`'s `passengers` property would be successfully updated as there is no `Passenger` currently at index/seat 5.

However, if the program attempted to do this:

```
bus1.addPassenger(new Passenger("Jill Jones", "Bahamas", 0));
```

then no change to `bus1`'s `passengers` property would be made.

Write code to implement the `addPassenger` method of the `Bus` class.

pre-condition: the parameter, `p`, has a valid `seatNum` value.

```
public void addPassenger(Passenger p){
```

[4]

e) Another implementation of the `addPassenger` method of the `Bus` class is being designed. This time, if the `addPassenger` method attempts to add the parameter `p` to `passengers` and the seat/index is available (ie null), then `p` is added as before. However, if the seat/index is unavailable, then the method will search for an available seat/index, add `p`, and update the `p`'s `seatNum` property. If there are no seats available, no changes are made to `passengers`.

Write code to implement this version of the `addPassenger` method of the `Bus` class.
pre-condition: the parameter, `p`, has a valid `seatNum` property.
pre-condition: if `p` is added to `passengers` at a seat index other than its original `seatNum`, its `seatNum` property will be updated

```
public void addPassenger(Passenger p){
```

[5]

f) A `SchoolBus` IS-A `Bus`! However, a SchoolBus has an additional property, a safety switch, which be either on or off. When an instance of the SchoolBus class is created, the status of the switch is passed as a parameter.
Write the entire `SchoolBus` class, including any necessary accessor/mutator methods.

[4]

g) A Trip has 5 Bus instances. Here is the entire Trip class:

```
public class Trip {

     private SchoolBus[] buses;

     public Trip(SchoolBus b1, SchoolBus b2, SchoolBus b3, SchoolBus b4,
                                                          SchoolBus b5) {
          buses= new SchoolBus[5];
          buses[0] = b1;
          buses[1] = b2;
          buses[2] = b3;
          buses[3] = b4;
          buses[4] = b5;
     }

     public SchoolBus[] getBuses(){
          return buses;
     }
     public void sortBuses() {
          //yet to be implemented
     }
     public int countNationality(String nationality){
          //yet to be implemented
     }
}
```

The `sortBuses` method of the `Trip` class sorts the buses in the trip into ascending order of number of passengers.
Write code to implement the `sortBuses` method of the `Trip` class.
Use the `getPassengerCount` method of the `Bus` class in your solution. it returns a count of the number of passengers on a bus. Assume it works as expected.

Your sorting algorithm should be either bubble sort or selection sort.

pre-condition: every trip has 5 buses even if one or more are empty
post-condition: the `buses` property is sorted in ascending order of number of passengers.

[5]

g) The `countNationality` method of the `Trip` class counts how many passengers on the `Trip` have the same nationality as the parameter `nationality`.
You may assume that there are 5 buses but any number of them may not be full of passengers ie your code should process nulls appropriately.

Write code to implement the `countNationality` method of the `Trip` class.

[5]

PART B - Topic 3 REVIEW

1 Outline one reason for the use of standards in the construction of networks.

[2]

2 Identify two differences between a wide area network (WAN) and a local area network (LAN).

[2]

3 Outline the reason for compression when transmitting data over a network.

[2]

4 A school currently has a cabled network but wants to add wireless networking across the whole campus.
Identify two advantages to the students of the new wireless network.

[2]

5 Identify two reasons why fibre optic cable would be preferred over wireless connectivity.

[2]