Lab 2 : Inheritance

Answers :

Question 1:

1(a). MountainBike class have 2 instance variable.
1(b). The purpose of the keyword ‘super()’ in the MountainBike constructor is to call and initialize the parent class.
1(c). MoutainBike class have 3 methods. 2 methods which is the default constructor and normal constructor and another is getter method.

Question 2:

Fix the code.

```
class Animal {
   private String hairColor;

   public Animal() {
      hairColor = "unknown";
   }

   public Animal(String hc) {
      hairColor = hc;
   }

   public void setHairColor(String hc){
         this.hairColor = hc;
   }

   public String getHairColor() {
      return hairColor;
   }

   public String toString() {
      return "I am an Animal with " + hairColor + " hair color.";
   }
}
```

```
public class Cat extends Animal
{
    protected int tagNumber;

     public Cat (int tn) {
        super();
        tagNumber = tn;
        if (getHairColor().equals("unknown"))
            setHairColor(“brown”);
     }
     public int getTagNumber() {
        return tagNumber;
     }
     public String toString() {
       return "I am a Cat with " + hairColor + " hair color and tag number "
       +tagNumber;
     }
  }
```

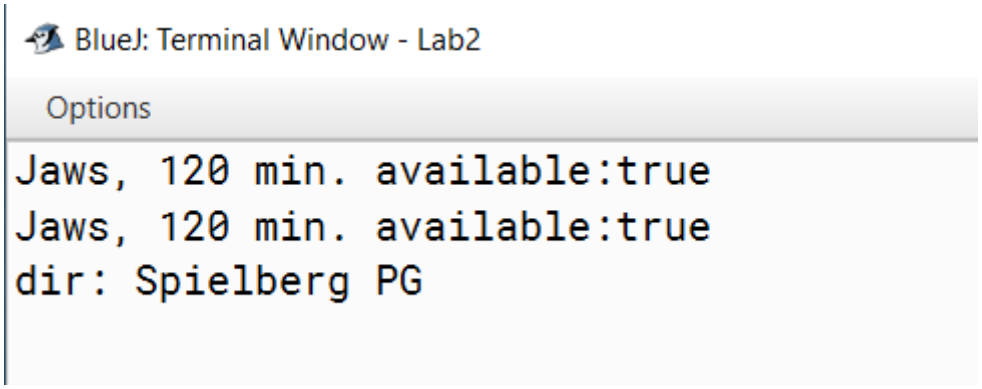
Question 3:

a.
```
class Movie extends VideoTape {
    String director;
    String rating;

    public Movie(String title, int length,String director, String   rating){
     super(title, length);
     this.director = director;
     this.rating = rating;
    }
}
```

b. Movie item2 = new Movie(“Jaws”, 120, “Spieberg”, “PG”);

c.
```
public void show()
{
   System.out.println( title + ", " + length + " min. available:" + avail );
   System.out.println( "dir: " + director + " " + rating );
}
```



Output for d

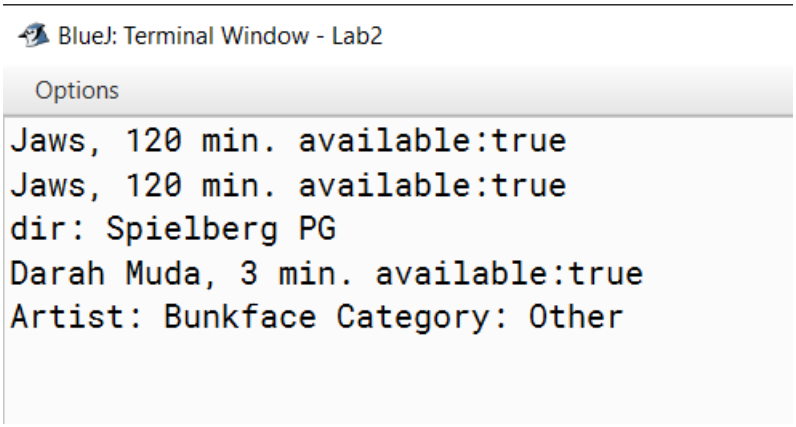
e.public void show(){

```
super.show();
System.out.println( "dir: " + director + " " + rating );
}
```

f. The output same with d

g. class MusicVideo extends VideoTape {

```
String artist; // name of the artist
String category;
 public MusicVideo(String title, int length, String artist, String category){
    super(title,length);
    this.artist = artist;
    this.category = category;
 }
 public void show(){
     super.show();
     System.out.println("Artist: " + artist + "Category: " + category);
  }
}
```



Output for h

Postlab Exercise

Answer:

a(i) normal constructor for superclass :

```
public Patient(String pn, int pnum, String address){
    patientName = pn;
    patientNumber = pnum;
    patientAddress = address;
}
```

Normal constructor for subclass:

```
public Room(String pn, int pnum, String address, String typeRoom, int noDay){
    super(pn, pnum, address);
    this.typeRoom = typeRoom;
    this.noDay = noDay;
}
```

(ii) Method calcPayment()

```
public double calcPayment(){
    double discount = 0.25;
    double disPrice = 0;
    double price = 0;
    if(typeRoom.equalsIgnoreCase("Diamond")){
        price = 200.00;
        if(noDay > 20){
            disPrice = price * discount;
```

```
            price = price - disPrice;
            return price;
        }
        return price;
    }else if(typeRoom.equalsIgnoreCase("Gold")){
        price = 100.00;
        if(noDay > 20){
            disPrice = price * discount;
            price = price - disPrice;
            return price;
        }
        return price;
    }else if (typeRoom.equalsIgnoreCase("Silver")){
        price = 80.00;
        if(noDay > 20){
            disPrice = price * discount;
            price = price - disPrice;
            return price;
        }
        return price;
    }else{
        price = 50.00;
        if(noDay > 20){
            disPrice = price * discount;
            price = price - disPrice;
            return price;
        }
        return price;
    }
}
```

b(i)

```
Scanner inputNum = new Scanner(System.in);
Scanner inputText = new Scanner(System.in);
int numData;

System.out.println("\nEnter number of patient data to be store: ");
numData = inputNum.nextInt();

//b(i)
Room[] roomList = new Room[numData];

for(int i = 0; i < roomList.length; i++){
    System.out.println("Enter patient name: ");
    String name = inputText.nextLine().toUpperCase();
    System.out.println("Enter patient number: ");
    int pnum = inputNum.nextInt();
    System.out.println("Enter patient address: ");
    String address = inputText.nextLine().toUpperCase();
    System.out.println("Enter type of room: ");
    String typeRoom = inputText.nextLine().toUpperCase();
    System.out.println("How long patient stay in days: ");
    int noDay = inputNum.nextInt();

    Room r = new Room(name,pnum,address,typeRoom,noDay);
    roomList[i] = r;
}
```

b(ii)

```
//b(ii)
int numPatient = 0;
for(int i =0; i < roomList.length; i++){
    if(roomList[i].getNoDay() > 20){
        numPatient += numPatient;
    }
}
System.out.println("Number of patient that stay more than 20 days is : " + numPatient);
```

b(iii)

```
//b(iii)
double diaRoom = 0;
double goldRoom = 0;
double silRoom = 0;
double bronRoom = 0;

for(int i =0; i < roomList.length; i++){
    if(roomList[i].getTypeRoom().equalsIgnoreCase("Diamond")){
        diaRoom = diaRoom + roomList[i].calcPayment();
    }else if(roomList[i].getTypeRoom().equalsIgnoreCase("Gold")){
        goldRoom = goldRoom + roomList[i].calcPayment();
    }else if(roomList[i].getTypeRoom().equalsIgnoreCase("Silver")){
        silRoom = silRoom + roomList[i].calcPayment();
    }else{
        bronRoom = bronRoom + roomList[i].calcPayment();
    }
}

System.out.println("Total payment for Diamond Room is RM "+ diaRoom);
System.out.println("Total payment for Gold Room is RM "+ goldRoom);
System.out.println("Total payment for Silver Room is RM "+ silRoom);
System.out.println("Total payment for Bronze Room is RM "+ bronRoom);
```

b(iv)

```
//b(iv)
double avg = 0, totalPayment = 0;
for(int i =0; i < roomList.length; i++){
    totalPayment = totalPayment + roomList[i].calcPayment();
}
avg = totalPayment / numPatient;
```