**Problem 1 : Accelerate the Car (20 Marks)**

**a)** Create a new Java class named **Car** that has the following **private** fields

⚫ **year** - The year field is an **int** that holds a car's year model (e.g. 2010)

⚫ **make** - The make field is a **String** object that holds the make of the car (e.g. "Porsche")

⚫ **speed** - The speed field is an **double** that holds a car's current speed (e.g. 25.0)

**b)** In addition, the Car class should have the following methods.

⚫ **Constructor** - The constructor should accept the car's year, make, and beginning speed as

arguments

◆ These values should be used to initialize the Car's year, make, and speed fields

⚫ **Getter Methods** - Write three accessor (getter) methods to get the values stored in an object's fields

⚫ **accelerate** - Write an **accelerate** method that has no arguments (parameters) passed to it and adds

**1** to the **speed** field each time it is called

◆ For example: if the car was going 3 mph, accelerate would set the speed to 4 mph

**c)** Write a separate java class **RaceTrack** in a separate file with a **main()** method that

⚫ Create a new **Car** object (using the Car constructor method), passing in the year, make, and speed

⚫ Display the current status of the car object using the getter methods **getYear**(), **getMake**(), and

**getSpeed**()

⚫ Call the car's accelerate method and then re-display the car's speed using **getSpeed**()

class Car{

private int year;

private String make;

private double speed;

Car( int year,String make, double speed){

this.year=year;

this.make=make;

this.speed=speed;

}

int getYear(){

return year;

}

String getMake(){

return this.make;

}

double getSpedd(){

return speed;

}

void accelerate(){

speed=speed+1;

}

}

class RaceTrack{

public static void main(String []args){

Car c1=new Car(2010,"Porsche",25.00);

//Currnt stutus of object

System.out.println("Year of manufacturing is =="+c1.getYear());

System.out.println("Name of car is =="+c1.getMake());

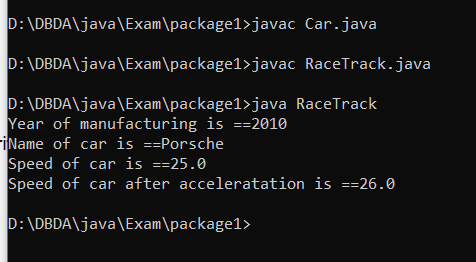
System.out.println("Speed of car is =="+c1.getSpedd());

c1.accelerate();

System.out.println("Speed of car after acceleratation is =="+c1.getSpedd());

}

}



**Problem 2 : Inventory Management (20 Marks)**

Write a program to create an inventory of items which will allow basic inventory management such as below

==============================================================

**1) Add Item (Prevent duplication)**

**2) Display complete inventory in sorted order of item names as well as itemId.**

**3) Remove Item.**

**4) Exit**

Please enter your choice (1-3)

1. Define a class **Item** with two attributes **itemId** and **itemName**.

2. The implementation should use **ArrayList** collection where every element of ArrayList collection holds the

object of class Item.

3. Put constraint on ArrayList of Item objects that, it should not allow insertion when the values of attributes

itemID and itemName previously exist **together** in the ArrayList.

Ex : If itemId and itemName with values (1, “Item1”) respectively, already exists in the **ArrayList** Collection, then

same entry should not exist in your collection.

--------------------------------------------------------------------------------------------------------------------------------------

import java.util.\*;

class Item{

int itemId;

String itemName;

Item(int itemId, String itemName){

this.itemId = itemId;

this.itemName = itemName;

}

Item(){}

void setitemId(int itemId){

this.itemId = itemId;

}

void setitemName(String itemName){

this.itemName = itemName;

}

public String toString(){

return this.itemId+ " " +this.itemName;

}

}

class namesort implements Comparator<Item>{

public int compare(Item I1, Item I2){

return I1.itemName.compareTo(I2.itemName);

}

}

class idsort implements Comparator<Item>{

public int compare(Item I1, Item I2){

return I1.itemId-(I2.itemId);

}

}

class inventory{

static Item I = new Item();

static ArrayList<Item> list = new ArrayList<> ();

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int choice;

Item I1 = new Item(1, "A");

Item I2 = new Item(3, "B");

Item I3 = new Item(2, "C");

Item I4 = new Item(4, "D");

Item I5 = new Item(10, "E");

list.add(I1);

list.add(I2);

list.add(I3);

list.add(I4);

list.add(I5);

do{

System.out.println("Enter your choice ==");

System.out.println("1) Add Item. \n2) Display complete inventory insorted order of itemnames as well as itemId. \n3) Remove Item. \n4) exit");

choice = sc.nextInt();

switch(choice){

case 1:

System.out.println("Enter your details as follows");

System.out.println("Enter Item you want add");

for (int i =1; i<=1; i++){

System.out.println("Enter id of item ");

int d = sc.nextInt();

I.setitemId(d);

System.out.println("Enter name of item ");

sc.nextLine();

String ss = sc.nextLine();

I.setitemName(ss);

list.add(I);

}

System.out.println("Addes item as follows");

System.out.println(list);

break;

case 2:

System.out.println("before sorting");

System.out.println(list);

System.out.println("Sorting by id");

idsort n2 = new idsort();

Collections.sort(list,n2);

System.out.println(list);

System.out.println("Sorting by name");

namesort n1 = new namesort();

Collections.sort(list, n1);

System.out.println(list);

break;

case 3:

System.out.println("List as follows");

System.out.println(list);

System.out.println("Enter index of item which you want to remove index start from 0");

int re = sc.nextInt();

list.remove(re);

System.out.println("list after removal");

System.out.println(list);

break;

case 4:

System.out.println("Thank you");

break;

}

} while(choice!=4);

}

}

Output

