Codethon Solutions

Codethon-2 Solutions:-

- **Problem-1 Employee –**

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;


    public class Employee
    {
    //Write your code here
    private string empCode,empName;
    private double empSal;
    private char deptCode;
    private static int empCounter = 1000;

    public Employee(string empName,double empSal,char deptCode)
    {
        this.empName = empName;
        this.empSal = empSal;
        this.deptCode = deptCode;
        empCounter++;
        this.empCode = generateEmployeeCode(deptCode);
    }

    public Employee(string empName,double empSal)
    {
        this.empName = empName;
        this.empSal = empSal;
        this.deptCode = 'A';
        empCounter++;
        this.empCode = generateEmployeeCode(deptCode);
    }
```

```
    public string EmpName{
        get{return empName;}
        set{empName=value;}
    }
    public double EmpSal{
        get{return empSal;}
        set{empSal=value;}
    }
    public string EmpCode{
        get{return empCode;}
    }
    public char DeptCode{
        get{return deptCode;}
        set{deptCode=value;}
    }

    private string generateEmployeeCode(char deptCode)
    {
        return empCounter+deptCode.ToString();
    }

    public string getEmployeeDetails()
    {
        return "Code-"+empCode+",Name-"+empName+",Salary-
"+empSal.ToString()+",Department-"+deptCode.ToString();
    }

  }
```

- **CD 2- Problem-2 Product :-**

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;

public class Product {
    //Write your code here
    private String productCode, productName;
    private double productPrice;
    private char categoryCode;
    private static int prodCounter = 100;

    private String generateProductCode()
   {
        prodCounter++;
        return String.Concat(this.categoryCode,Convert.ToString(prodCounter));
    }

    public String getProductDetails()
   {
        return String.Format("Code-{0},Name-{1},Price-{2},Category-{3}",this.productCode,
this.productName, this.productPrice, this.categoryCode);
    }

    public Product() {}

    public Product(String productName, double productPrice, char categoryCode) {
        this.productName = productName;
        this.productPrice = productPrice;
        this.categoryCode = categoryCode;
        this.productCode = generateProductCode();
    }

    public Product(String productName, double productPrice) {
        this.productName = productName;
        this.productPrice = productPrice;
        this.categoryCode = 'E';
        this.productCode = generateProductCode();
    }
```

```
    public String ProductCode {
        get {return productCode; }
        set {productCode = value; }
    }

    public String ProductName {
        get {return productName; }
        set {productName = value; }
    }

    public double ProductPrice {
        get {return productPrice; }
        set {productPrice = value; }
    }

    public char CategoryCode {
        get {return categoryCode; }
        set {categoryCode = value; }
    }
}

class Example {
    public static void Main() {
        Product p=new Product("Laptop",45000.00, 'E');
        Console.WriteLine(p.getProductDetails());
    }
}
```

- **CD 2 Problem-3 Player:-**

```
class Player
    {
        string name;
        string country;
        int age;
        DateTime doj;

        public string Name
        { get { return name; } set { name = value; } }
```

```
        public string Country
        { get { return country; } set { country = value; } }

        public int Age
        {
           get { return age; }
           set { age = value; }
        }

        public DateTime Doj { get { return doj; }set { doj = value; } }

        public Player() { }

        public Player(string name, string country, int age, DateTime doj)
        {
            this.name = name;
            this.country = country;
            this.age = age;
            this.doj = doj;
        }

  public override string ToString()
        {
            return "Name-" + Name + ",Country-" + Country + ",Age-" + Age;
        }

        public bool Equals(Player p)
        {
            if (this.name.ToLower() == p.name.ToLower() & this.country.ToLower() ==
p.country.ToLower())
                return true;
            else
                return false;
        }
    }
```

Codethon-3 Solutions:-

- **CD3 Problem-1 Cab –**

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;

class Cab
{
    //Write your code here
    private string cid,regNo,type;
    private int capacity;
    private double costPerKm;
    public string Cid
    {
        get{return cid;}
        set{cid=value;}
    }
    public string Type
    {
        get{return type;}
        set{type=value;}
    }
    public string RegNo
    {
        get{return regNo;}
        set{regNo=value;}
    }
    public int Capacity
    {
        get{return capacity;}
        set{capacity=value;}
    }
    public double CostPerKm
    {
        get{return costPerKm;}
        set{costPerKm=value;}
    }
    public Cab()
```

```
    {
    }
    public Cab(string cid,string regno,string type,int capacity,double costPerKm)
    {
        this.regNo=regno;
        this.cid=cid;
        this.type=type;
        this.capacity=capacity;
    this.costPerKm=costPerKm;

    }

    public override string ToString()
    {
      //Write your code here
      return
"RegistrationNumber:"+RegNo+"VehicleType:"+Type+"Capacity:"+Capacity+"CostPerKm:"+
CostPerKm+"";
    }

    public bool Equals(Cab c)
    {
      //Write your code here

      if(this.RegNo.ToLower()==c.RegNo.ToLower() &&
this.Type.ToLower()==c.Type.ToLower())
      {
          return true;
      }
      else
      return false;

    }

}
```

- **CD3 Problem-2 Movie –**

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
```

```
class Movie
{
    private string name;
    private string movieId;
    private string director;
    private int rating;
    private DateTime releaseDate=new DateTime();

    public string Name{
        get{return name;}
        set{name=value;}
    }
    public string MovieId{
        get{return movieId;}
        set{movieId=value;}
    }
    public string Director{
        get{return director;}
        set{director=value;}
    }
    public int Rating{
        get{return rating;}
        set{rating=value;}
    }
    public DateTime ReleaseDate{
        get{return releaseDate;}
        set{releaseDate=value;}
    }
    public Movie(){}
    public Movie(string name,string movieId,string director,int rating,
            DateTime releaseDate){
        this.name=name;
        this.movieId=movieId;
        this.director=director;
        this.rating=rating;
        this.releaseDate=releaseDate;
    }
    //Write your code here
    public override string ToString()
    {
        //Write your code here.
        return "Name:"+name+"\n"+
                "MovieId:"+movieId+"\n"+
```

```
            "Director:"+director+"\n"+
            "Rating:"+rating+"\n"+
            "ReleaseDate:"+releaseDate;
    }

    public  bool Equals(Movie m)
    {
        //Write your code here
        if((this.name.ToLower()==m.name.ToLower())
            &&(this.movieId.ToLower()==m.movieId.ToLower()))
            return true;
        else
            return false;
    }

}
```

- **CD3 Problem-3 Product :-**

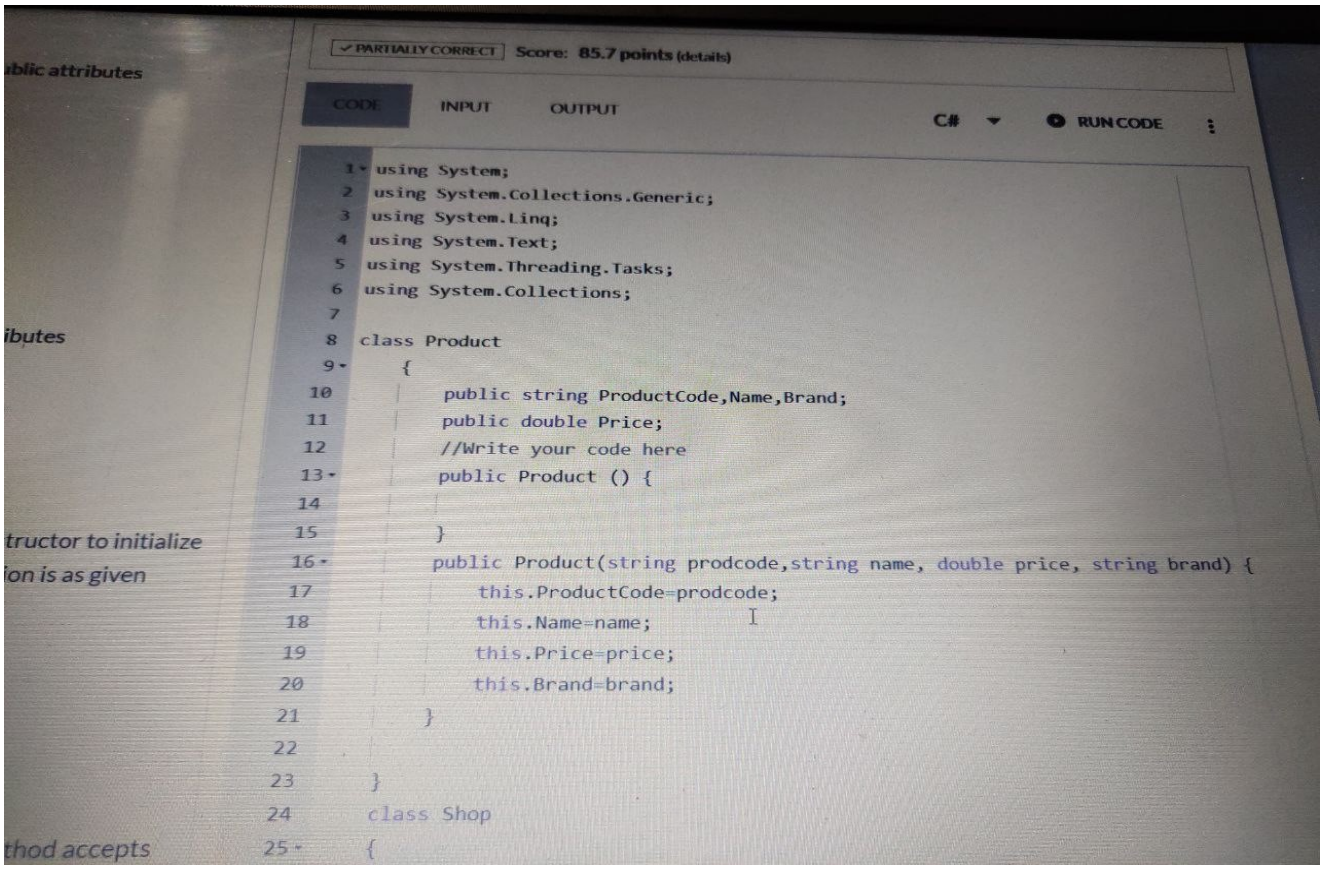
```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;

class Product
{
  //Write your code here
  private string name,productCode,brandName;
  private int stockLeft;
  private double price;
  private DateTime expiryDate;

  public string Name{
     get{return name;}
     set{name=value;}
  }
  public string ProductCode{
     get{return productCode;}
     set{productCode=value;}
  }
  public string BrandName{
```

```
    get{return brandName;}
    set{brandName=value;}
  }
  public int StockLeft{
    get{return stockLeft;}
    set{stockLeft=value;}
  }
  public double Price{
    get{return price;}
    set{price=value;}
  }
  public DateTime ExpiryDate{
    get{return expiryDate;}
    set{expiryDate=value;}
  }
  public Product(){}

  public Product(string name,string productCode,string brandName,int stockLeft,double
price, DateTime expiryDate){
    this.name=name;
    this.productCode=productCode;
    this.brandName=brandName;
    this.stockLeft=stockLeft;
    this.price=price;
    this.expiryDate=expiryDate;
  }



  public override string ToString()
      {
        //Write your code here
        return
"Name:"+Name+"\n"+"ProductCode:"+ProductCode+"\n"+"BrandName:"+BrandName+"\n"
+"StockLeft:"+StockLeft.ToString()+"\n"+"Price:"+Price.ToString()+"\n"+"ExpiryDate:"+Expiry
Date.ToString();

      }

      public  bool Equals(Product p)
      {
```

```
            //Write your code here
            if(this.Name.ToLower()==p.Name.ToLower() &&
this.ProductCode.ToLower()==p.ProductCode.ToLower()){
                return true;
            }
            else{
                return false;
            }
        }

    }
```

Codethon-4 Solutions:-

- **CD4 Problem-1 Product and Shop Class –**



the list.

Code) :This
es the
duct with a
rue.If

st,
d name

e given

```
public List<Product> ProdList = new List<Product>();

public Shop() {

}
public Shop(string name, List<Product> productList) {
    this.Name=name;
    this.ProdList=productList;
}

public void AddProductToShop(Product p)
{
    //Write your code here
    int flag=0;
    foreach(var str in ProdList) {
        if(str.Name==p.Name && str.ProductCode==p.ProductCode) {
            flag=1;
        }
    }
    if(flag==0) {
        ProdList.Add(p);
    }

}
public bool RemoveProductFromShop(string productCode)
{
    //Write your code here
```


that

```
    }
    return false;
}

}
class ProductBO
{
    public List<Product> ProdList2 = new List<Product>();
    public List<Product> ProdList3 = new List<Product>();
    public List<Product> FindProduct(List<Product> productList, string brand)
    {
//Write your code here

        foreach( var str in productList) {
            if(str.Brand==brand) {
                ProdList2.Add(str);
            }
        }
        return ProdList2;

    }

    public List<Product> FindProduct(List<Product> productList, double price)
```

```
foreach( var str in productList) {
    if(str.Price==price) {
        ProdList3.Add(str);
    }
}
return ProdList3;
}

public Hashtable BrandWiseCount(List<Product> productList )
{

    Hashtable ht1 = new Hashtable();
//Write your code here
    return ht1;


}
}
```

10 revisions found for this solution.

SHOW REVISIONS

manipal-adapt.in.capgemini.com is sharing your screen. Stop sharing Hide

- **CD4 Problem-2 Passenger and Cab Class –**

```
class Passenger
    {
        //Write your code here
        public string Pid,Name,Email;
        public int ContactNo;

    }

    class Cab
    {
     //Write your code here
     public string Cabid;
     public string RegNo;
     public string Type;
     public int Capacity;
     public double CostPerKm;
     public List<Passenger> PassengerList= new List<Passenger>();

     public Cab(){

     }
     public Cab(string cabid,string regno,string type,int capacity,double cost,List<Passe
         this.Cabid=cabid;
```

```
     public double CostPerKm;
     public List<Passenger> PassengerList= new List<Passenger>();

     public Cab(){

     }
     public Cab(string cabid,string regno,string type,int capacity,double cost,List<Passe
         this.Cabid=cabid;
         this.RegNo=regno;
         this.Type=type;
         this.Capacity=capacity;
         this.CostPerKm=cost;
         this.PassengerList=paslist;

     }
     public void AddPassengerToCab(Passenger p)
     {
         //Write your code here
         if(!PassengerList.Contains(p))
         PassengerList.Add(p);
     }
     public bool RemovePassengerFromCab(string id)
     {
         //Write your code here
```

ab class

nger p) : This method
assenger to the passenger
h same id and name already
not be added to the list.

ring id) :This method will
ssenger with specified id
en id is found , then
enger with id is not

hods:

ing type) : This
parameter..This
list of cabs that

```
    }
    public void AddPassengerToCab(Passenger p)
    {
        //Write your code here
        if(!PassengerList.Contains(p))
        PassengerList.Add(p);
    }
    public bool RemovePassengerFromCab(string id)
    {
        //Write your code here
        foreach(var p in PassengerList){
            var code=p.Pid;
            if(code==id){
                PassengerList.Remove(p);
                return true;
            }
        }
        return false;
    }



}

class CabBO
{

    public List<Cab> FindCab(List<Cab> cabList, string type)
    {
```

as input parameter..This
returns a list of cabs that

bList , int capacity) : This
s input parameter.This
turns a list of cabs that

Cab> cabList) : This
bject that contains

```
{

    public List<Cab> FindCab(List<Cab> cabList, string type)
    {
        //Write your code here
        var list=new List<Cab>();
        foreach(var cab in cabList){
            if(cab.Type==type){
                list.Add(cab);
            }
        }
        return list;
    }
    public List<Cab> FindCab(List<Cab> cabList, int capacity)
    {
        //Write your code here
        var list=new List<Cab>();
        foreach(var cab in cabList){
            if(cab.Capacity==capacity){
                list.Add(cab);
            }
        }
        return list;
    }

    public Hashtable CapacityWiseCount(List<Cab> cabList)
    {
        Hashtable ht = new Hashtable();
        //Write your code here
        return ht;

    }
```

Final Assesment Webinar Codes:-

Question-1 (OOPs):-

```
class Player
  {
      string name;
      string country;
      int age;
      DateTime doj;


      public string Name
      { get { return name; } set { name = value; } }
      public string Country
      { get { return country; } set { country = value; } }
      public int Age
      {
        get { return age; }
        set { age = value; }
      }
      public DateTime Doj { get { return doj; }set { doj = value; } }
      public Player() { }
      public Player(string name, string country, int age, DateTime doj)
      {
         this.name = name;
         this.country = country;
         this.age = age;
```

```
        this.doj = doj;
    }

public override string ToString()
    {
        return "Name-" + Name + ",Country-" + Country + ",Age-" + Age;
    }
    public bool Equals(Player p)
    {
        if (this.name.ToLower() == p.name.ToLower() & this.country.ToLower() == p.country.ToLower())
            return true;
        else
            return false;
    }
  }
public override string ToString()
    {
        return "Name-" + Name + ",Country-" + Country + ",Age-" + Age;
    }
    public bool Equals(Player p)
    {
        if (this.name.ToLower() == p.name.ToLower() & this.country.ToLower() == p.country.ToLower())
            return true;
        else
            return false;
   } } }
```

- **Question-2 (Collections) :-**

```
class Player
  {
     public string Fname;
     public string Lname;
     public string Country;
     public int Rating;
  }

  class Match
  {
     public List<Player> team1 = new List<Player>();
     public List<Player> team2 = new List<Player>();
     public Match(List<Player> l1, List<Player> l2)
     {
        this.team1 = l1;
       this.team2 = l2;
     }
     public Match() { }

public void AddPlayer(Player p, string team)
     {
        if (team == "Team1")
           team1.Add(p);
        if (team == "Team2")
           team2.Add(p);
     }


 public bool RemovePlayer(string fname)
     {
        bool flag;
        Player p1 = team1.Find(x => x.Fname == fname);
        Player p2= team2.Find(x => x.Fname == fname);
        if( p1== null & p2==null)
        {
           flag = false;
        }
        else
```

```
        {
            if (p1 != null)
                team1.Remove(p1);
            if (p2 != null)
                team2.Remove(p2);
            flag = true;
        }
        return flag;
    }
}

class MatchBO
{
    public List<Player> FindPlayer(List<Player> playerList, string country)
    {
        List<Player> l1 = playerList.FindAll(x => x.Country == country);
        return l1;
    }
    public List<Player> FindPlayer(List<Player> playerList, int rating)
    {
        List<Player> l1 = playerList.FindAll(x => x.Rating == rating);
        return l1;
    }
    public Hashtable CountryWiseCount(List<Player> playerList)
    {
        Hashtable ht = new Hashtable();
        var res = from p in playerList
                  group p by p.Country into grp
                  select new { Country = grp.Key, Cnt = grp.Count() };
        foreach( var x in res)
        {
            ht.Add(x.Country, x.Cnt);
        }
        return ht;

    }
}
```

---END---

Collections- Product

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Collections;
namespace Practice_2
{
    class Product
    {
        public string ProductCode, Name, Brand;
        public double Price;

        public Product() { }
        public Product(string prodcode, string name, double price,string brand)
        {
            ProductCode = prodcode;
            Name = name;
            Price = price;
            Brand = brand;

        }
    }

    class Shop
    {
        string Name;
        List<Product> Prodlist = new List<Product>();


        public Shop() { }
        public Shop(string name, List<Product> productlist)
        {
            Name = name;
            Prodlist = productlist;
        }

        public void AddproductToShop(Product p)
        {
            int flag = 0;
            foreach (var str in Prodlist)
            {
                if (str.ProductCode == p.ProductCode && str.Name == p.Name)
                {
                    flag = 1;
                }
            }
            if (flag == 0)
                Prodlist.Add(p);
        }

        public bool RemoveProductFromShop(string productCode)
        {
            foreach (var str in Prodlist)
            {
                if (str.ProductCode == productCode)
                {
                    Prodlist.Remove(str);
```

```
                return true;
            }
        }

        return false;
    }
}


class ProductBO
{
    public List<Product> FindProduct(List<Product> prodlist, string brand)
    {
        List<Product> prod4 = new List<Product>();
        foreach (var str in prodlist)
        {
            if (str.Brand == brand)
                prod4.Add(str);
        }

        return prod4;
    }
    public List<Product> FindProduct(List<Product> prodlist, double price)
    {
        List<Product> prod2 = new List<Product>();
        foreach (var str in prodlist)
        {
            if (str.Price == price)
                prod2.Add(str);
        }

        return prod2;
    }

    public Hashtable BrandWiseCount(List<Product> productList)
    {
        Hashtable ht = new Hashtable();
        var res = from p in productList
                  group p by p.Brand into brand
                  select new {Brand=brand.Key,Cnt=brand.Count() };

        foreach(var x in res)
        {
            ht.Add(x.Brand, x.Cnt);
        }

        return ht;

    }

}
}
```

Collections- Movie/Theatre

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Collections;

namespace Practice_2
{
    class Movie
    {
        public string MovieId;
        public string Name;
        public int Rating;
        public string Language;
        public Movie() { }

        public Movie(string MovieId, string Name, int Rating, string Language)
        {
            this.MovieId = MovieId;
            this.Name = Name;
            this.Rating = Rating;
            this.Language = Language;
        }
    }

    class Theatre
    {
        public string Name;
        public List<Movie> MovieList = new List<Movie>();

        public Theatre() { }
        public Theatre(string Name, List<Movie> MovieList)
        {
            this.Name = Name;
            this.MovieList = MovieList;
        }

        public void AddMovieToTheatre(Movie m)
        {
            int flag = 0;
            foreach(var mov in MovieList)
            {
                if(mov.Name==m.Name && mov.Language==m.Language)
                {
                    flag = 1;
                }
            }

            if(flag==0)
            {
                MovieList.Add(m);
            }
        }

        public bool UpdateMovie(string movieid, int newrating)
        {
            foreach(var mov in MovieList)
```

```
            {
                if(mov.MovieId==movieid)
                {
                    mov.Rating = newrating;
                    return true;
                }
            }
            return false;
        }
    }

    class MovieBo
    {
        public List<Movie> FindMovie(List<Movie> MovieList,string Language)
        {
            List<Movie> List = new List<Movie>();
            foreach(var mov in MovieList)
            {
                if(mov.Language==Language)
                {
                    List.Add(mov);
                }
            }
            return List;
        }

        public List<Movie> FindMovie(List<Movie> MovieList, int Rating)
        {
            List<Movie> List = new List<Movie>();
            foreach (var mov in MovieList)
            {
                if (mov.Rating == Rating)
                {
                    List.Add(mov);
                }
            }
            return List;
        }

        public Hashtable LanguageWiseCount(List<Movie> MovieList)
        {
            Hashtable ht = new Hashtable();
            var res = from m in MovieList
                      group m by m.Language into lang
                      select new {Movie=lang.Key,Cnt=lang.Count()};
            foreach(var x in res)
            {
                ht.Add(x.Movie, x.Cnt);
            }

            return ht;
        }
    }
}
```

Collections- Student

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Practice_2
{
    class Student
    {
        public string Name { get; set; }
        public int Score { get; set; }
    }

    class StudentImplementation
    {
        public StudentImplementation() { }

        public string NameofAllStudents(IList<Student> students)
        {
            string ans = "";
            foreach(Student s in students)
            {
                Console.WriteLine(s.Name);
                ans = ans + s.Name + " ";
            }

            return ans;
        }

        public int TotalScoreOfAllStudents(IList<Student> students)
        {
            int res = 0;
            foreach(Student s in students)
            {
                res = res + s.Score;
            }
            Console.WriteLine(res);
            return res;
        }

        public double AverageScore(IList<Student> students)
        {
            double res = 0;
            double i = 0;
            double average = 0;
            foreach(Student s in students)
            {
                i++;
                res = res + s.Score;
            }
            if(i!=0)
            average = res / i;
            Console.WriteLine(res);
            return average;
        }
    }
}
```

Collections- Player

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Collections;

namespace Practice1
{
    class Player
    {
        public string Fname, Lname, Country;
        public int Rating;
    }

    class Match
    {
        public List<Player> team1 = new List<Player>();
        public List<Player> team2 = new List<Player>();

        public Match() { }

        public Match(List<Player>team1,List<Player>team2)
        {
            this.team1 = team1;
            this.team2 = team2;
        }
        public void AddPlayer(Player p, string team)
        {
            if(team == "Team1")
            {
                team1.Add(p);
            }
            if (team == "Team2")
            {
                team2.Add(p);
            }
        }

        public bool RemovePlayer(string fname)
        {
            bool flag;
            Player p1 = team1.Find(x => x.Fname == fname);
            Player p2 = team2.Find(x => x.Fname == fname);
            if(p1==null && p2==null)
            {
                flag = false;
            }
            else
            {
                if(p1!=null)
                {
                    team1.Remove(p1);
                }
                if (p2 != null)
                {
                    team1.Remove(p2);
                }
```

```
                flag = true;
            }
            return flag;
        }
    }

    class MatchBo
    {
        public List<Player>FindPlayer(List<Player>playerList,string country)
        {
            List<Player> playerList1 = new List<Player>();
            foreach(var player in playerList)
            {
                if(player.Country==country)
                {
                    playerList1.Add(player);
                }
            }
            return playerList1;
        }

        public List<Player>FindPlayer(List<Player>playerList,int rating)
        {
            List<Player> playerList2 = new List<Player>();
            foreach(var player in playerList)
            {
                if(player.Rating==rating)
                {
                    playerList2.Add(player);
                }
            }
            return playerList2;
        }

        public Hashtable CountryWiseCount(List<Player> playerlist)
        {
            Hashtable ht = new Hashtable();

            var res = from p in playerlist
                      group p by p.Country into country
                      select new { Country=country.Key,Cnt=country.Count()};

            foreach(var x in res)
            {
                ht.Add(x.Country, x.Cnt);
            }
            return ht;
        }

    }
}
```