Maharashtra Education Society's
INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC), PUNE (AUTONOMOUS)
Affiliated to Savitribai Phule Pune University

IT21 - Java Programming

Practical Assignments 3

A.Y 2024-25

1.You have developed an e-commerce website for your client. The maximum no of units of a single prodcut that one user can add to the cart is 5. If the user adds more than 5 units of a single product, then your application is expected to throw, MaximumProductsLimitExceededException. Write a custom exception class to achieve this.

→MaxProdLimitedException.java

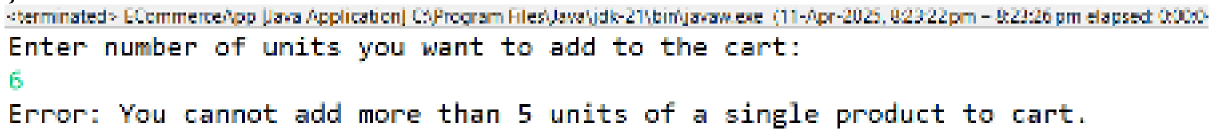
```
package pkgtablet;
public class MaxProdLimitExceededException extends Exception {
public MaxProdLimitExceededException(String message) {
super(message);
}
}
```

→ShoppingCart.java

```
package pkgProducts;
import pkgtablet.MaxProdLimitExceededException;
public class ShoppingCart {
private static final int MAX_PRODUCT_LIMIT=5;
public void addProductToCart(int quantity)throws MaxProdLimitExceededException
{
if(quantity>MAX_PRODUCT_LIMIT) {
throw new MaxProdLimitExceededException(
"You cannot add more than "+MAX_PRODUCT_LIMIT+" units of a single product to cart."
);
}
//Logic to add product to cart
System.out.println(quantity+" units added to the cart.");
}
}
```

→ECommerceApp.java

```
package pkgProducts;
import java.util.Scanner;
import pkgtablet.MaxProdLimitExceededException;
public class ECommerceApp {
public static void main(String[] args) {
Scanner scanner=new Scanner(System.in);
ShoppingCart cart=new ShoppingCart();
try {
System.out.println("Enter number of units you want to add to the cart:");
int units=scanner.nextInt();
cart.addProductToCart(units);
}catch(MaxProdLimitExceededException e) {
System.out.println("Error: "+e.getMessage());//Handles custom exception
}finally {
scanner.close();
}
}
}
```



```
<terminated> ECommerceApp [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (11-Apr-2025, 8:23:22 pm – 8:23:26 pm elapsed: 0:00:0
Enter number of units you want to add to the cart:
6
Error: You cannot add more than 5 units of a single product to cart.
```

2.The manufacturing of your medical company has very strict standards of product specifications. After each pill /tablet is ready, it is weighed. If the weight of the tablet exceeds the allowed limit, TabletWeightExceededException is raised. Using exception handling in Java, write the program to achieve the above business requirement.

→Tablet.java

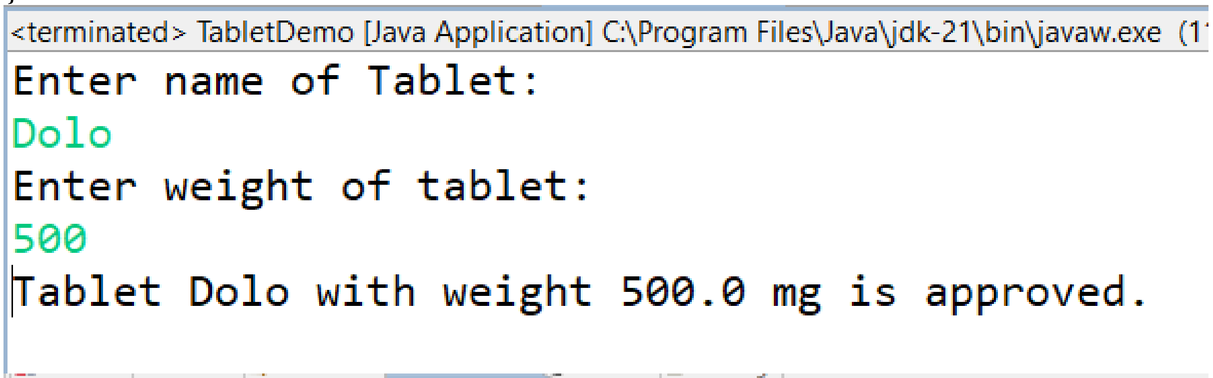
```
package pkgtablet;
public class Tablet {
private String name;
private double weight;
private static final double MAX_WEIGHT=500.0;
public Tablet(String name,double weight) throws TabletWeigthExceededException{
if(weight>MAX_WEIGHT) {
throw new TabletWeigthExceededException("Error: "+name+" exceeds the maximum allowed weight of "+MAX_WEIGHT+" mg.");
}
this.name=name;
this.weight=weight;
System.out.println("Tablet "+name+" with weight "+weight+" mg is approved.");
}
public String getName() {
return name;
}
public double getWeight() {
return weight;
}
}
```

→TabletDemo.java

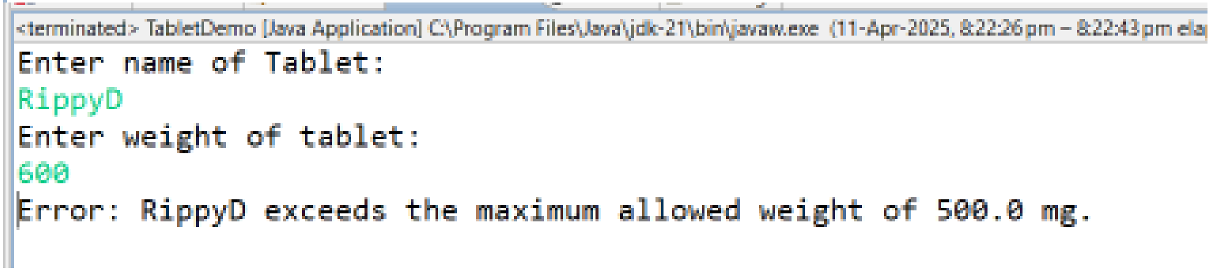
```
package pkgtablet;
import java.util.*;
public class TabletDemo {
public static void main(String[] args) {
Scanner scanner=new Scanner(System.in);
try {
System.out.println("Enter name of Tablet:");
String name=scanner.nextLine();
System.out.println("Enter weight of tablet:");
double weight=scanner.nextDouble();
Tablet tablet1=new Tablet(name,weight);
//Tablet tablet2=new Tablet("Dolo",500.0);
//Tablet tablet3=new Tablet("Aspirin",520.0);//This will throw exception
}
catch(TabletWeigthExceededException e) {
System.out.println(e.getMessage());
}
}
}
```

→TabletWeigthExceededException.java

```
package pkgtablet;
public class TabletWeigthExceededException extends Exception{
public TabletWeigthExceededException(String message) {
super(message);
}
}
```



```
<terminated> TabletDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (1
Enter name of Tablet:
Dolo
Enter weight of tablet:
500
Tablet Dolo with weight 500.0 mg is approved.
```



```
<terminated> TabletDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (11-Apr-2025, 8:22:26 pm – 8:22:43 pm ela
Enter name of Tablet:
RippyD
Enter weight of tablet:
600
Error: RippyD exceeds the maximum allowed weight of 500.0 mg.
```

3.When the battery of your mobile phone is less than 20%, the system should generate, LowBatteryException to alert the user to start charging the device. If the battery goes lower than 10 then the system should raise InsufficientChargeException and put the unit on power saver mode. Using exception handling in Java, write the program to achieve the above business requirement.

→InsufficientCharge.java

```
package pkgBattery;
public class InsufficientCharge extends Exception {
public InsufficientCharge(String message) {
super(message);
}
}
```

→LowBattery.java

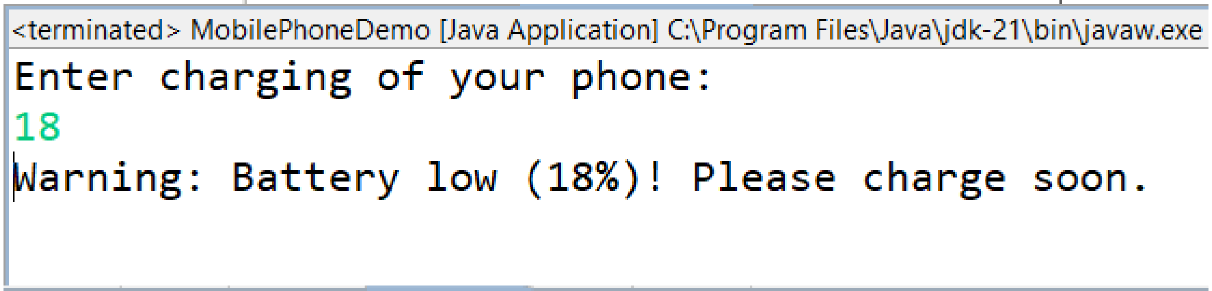
```
package pkgBattery;
public class LowBattery extends Exception {
public LowBattery(String message) {
super(message);
}
}
```

→MobilePhone.java

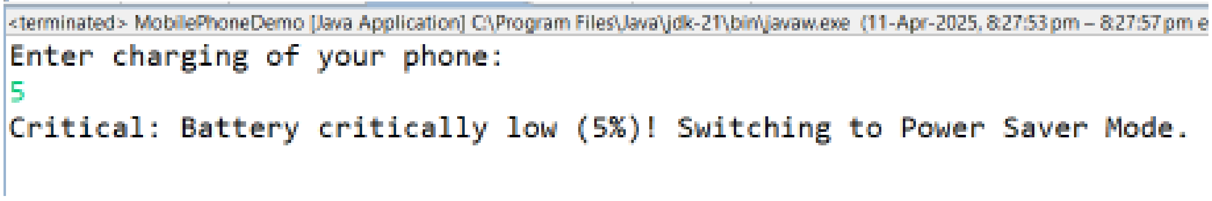
```
package pkgBattery;
public class MobilePhone {
private int batteryLevel;
public MobilePhone(int batteryLevel) throws LowBattery,InsufficientCharge {
this.batteryLevel=batteryLevel;
if(batteryLevel<10) {
throw new InsufficientCharge("Battery critically low (" + batteryLevel + "%)! Switching to
Power Saver Mode.");
}else if(batteryLevel<20) {
throw new LowBattery("Battery low (" + batteryLevel + "%)! Please charge soon.");
}else {
System.out.println("No need to charge!");
}
}
public int getBatteryLevel() {
return batteryLevel;
}
}
```

→MobilePhoneDemo.java

```
package pkgBattery;
import java.util.*;
public class MobilePhoneDemo {
public static void main(String[] args) {
Scanner scanner=new Scanner(System.in);
try {
System.out.println("Enter charging of your phone:");
Integer batteryLevel=scanner.nextInt();
MobilePhone phone=new MobilePhone(batteryLevel);
}
catch(LowBattery e) {
System.out.println("Warning: "+e.getMessage());
}catch(InsufficientCharge e) {
System.out.println("Critical: "+e.getMessage());
}
}
}
```



```
<terminated> MobilePhoneDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Enter charging of your phone:
18
Warning: Battery low (18%)! Please charge soon.
```



```
<terminated> MobilePhoneDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (11-Apr-2025, 8:27:53 pm – 8:27:57 pm e
Enter charging of your phone:
5
Critical: Battery critically low (5%)! Switching to Power Saver Mode.
```

4.You are writing an app for taking names of the volunteers for Cultural Committee of your Institute. According to the guidelines only 15 members are allowed in the committee. Using your app, take the names of the interested candidates till the number reaches 15. Once the threshold is crossed, display a message, “No more candidates allowed as volunteers. Thank you”. Use ArrayList to achieve the above given business logic.Hint : You will have to keep checking the size of the arraylist.

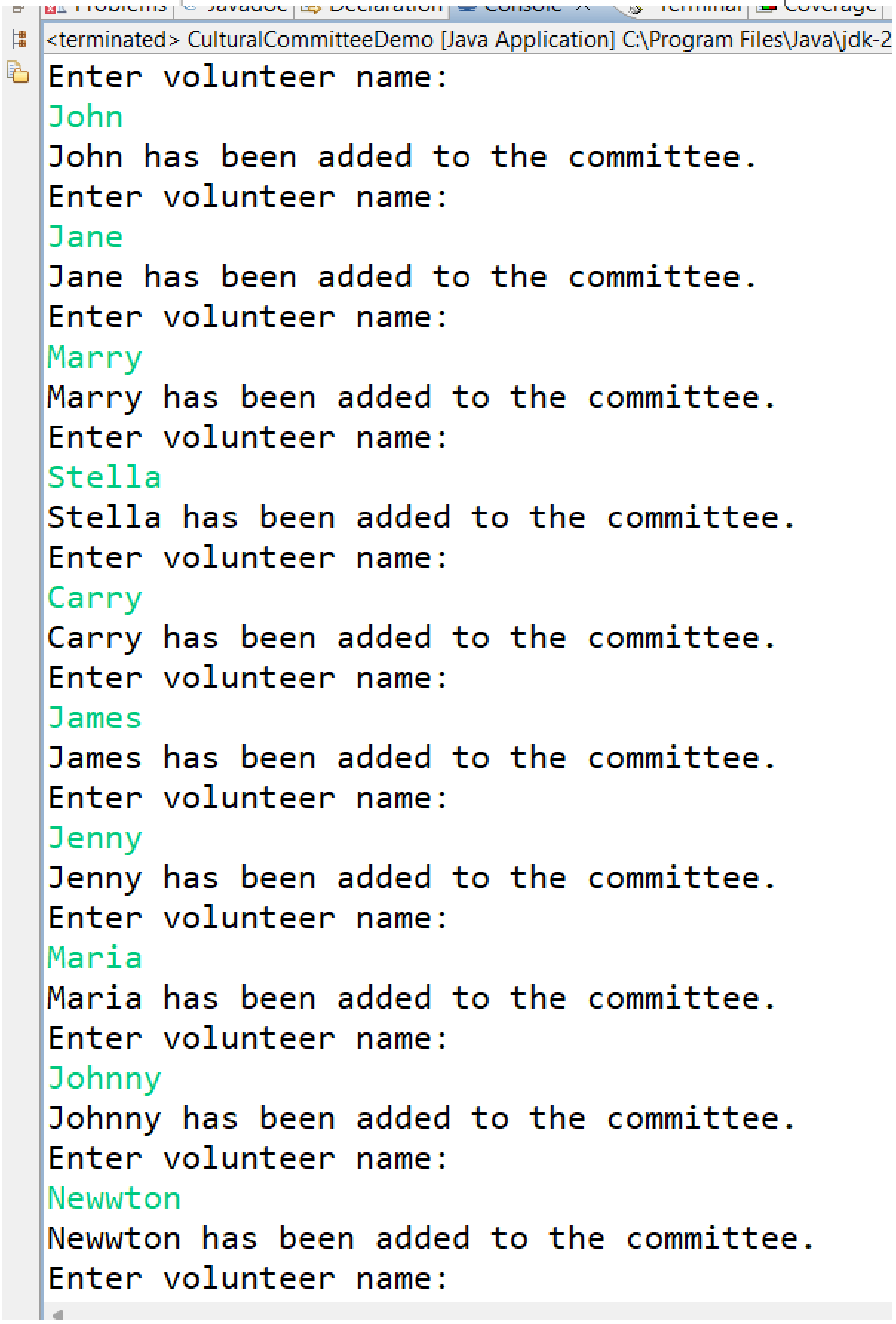
5.Once the above list of volunteers is finalized, each volunteer needs to pick a historical character as his/her badge icon. Using your app, take the name of the historical character from the volunteers and store them for future uses. Also, no two characters should be the same. In case the character is already in the list, ask the volunteer to enter some other character. Use ArrayList to achieve the above given requirement. Hint : You will have to check if the element is already contained inside the list.

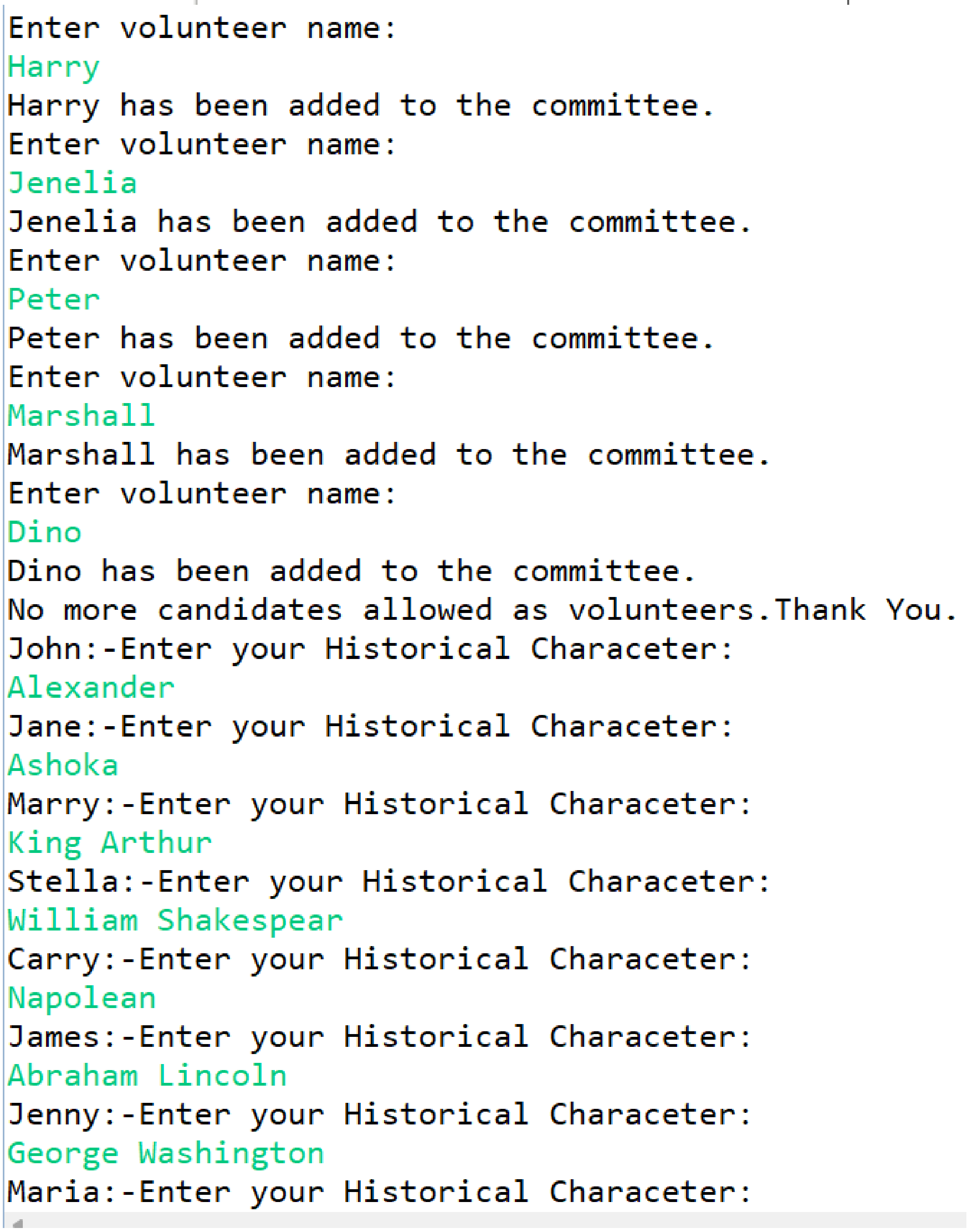
→CulturalComitee.java

```
package pkgCulturalComitee;
import java.util.*;
public class CulturalComitee
{
Scanner scanner=new Scanner(System.in);
ArrayList<String> volunteers=new ArrayList<String>();
HashMap<String,String> assignedCharacters=new HashMap<>();
//to add name of volunteers to arraylist
public ArrayList<String> addElementsToCollection() {
while(volunteers.size()<15)
{
System.out.println("Enter volunteer name:");
String name=scanner.nextLine();
volunteers.add(name);
System.out.println(name+" has been added to the committee.");
}
System.out.println("No more candidates allowed as volunteers.Thank You.");
return volunteers;
}
//to add historical character to hashmap
public HashMap<String,String> addHistoricalCharacter()
{
for(String volunteer:volunteers) {
String character;
while(true)
{
System.out.println(volunteer +":-Enter your Historical Characeter:");
character=scanner.nextLine();
//check if character is already assigned
if(!assignedCharacters.containsValue(character)) {
assignedCharacters.put(volunteer, character);
break;
}else {
System.out.println("This character is already taken.Please choose another one.");
}
}
}
return assignedCharacters;
}
//Display final list
public void displayList() {
System.out.println("\nFinal List of Volunteers and their Historical Characters:");
for(String volunteer:assignedCharacters.keySet()) {
System.out.println(volunteer+"->"+assignedCharacters.get(volunteer));
}
scanner.close();
}
}
```

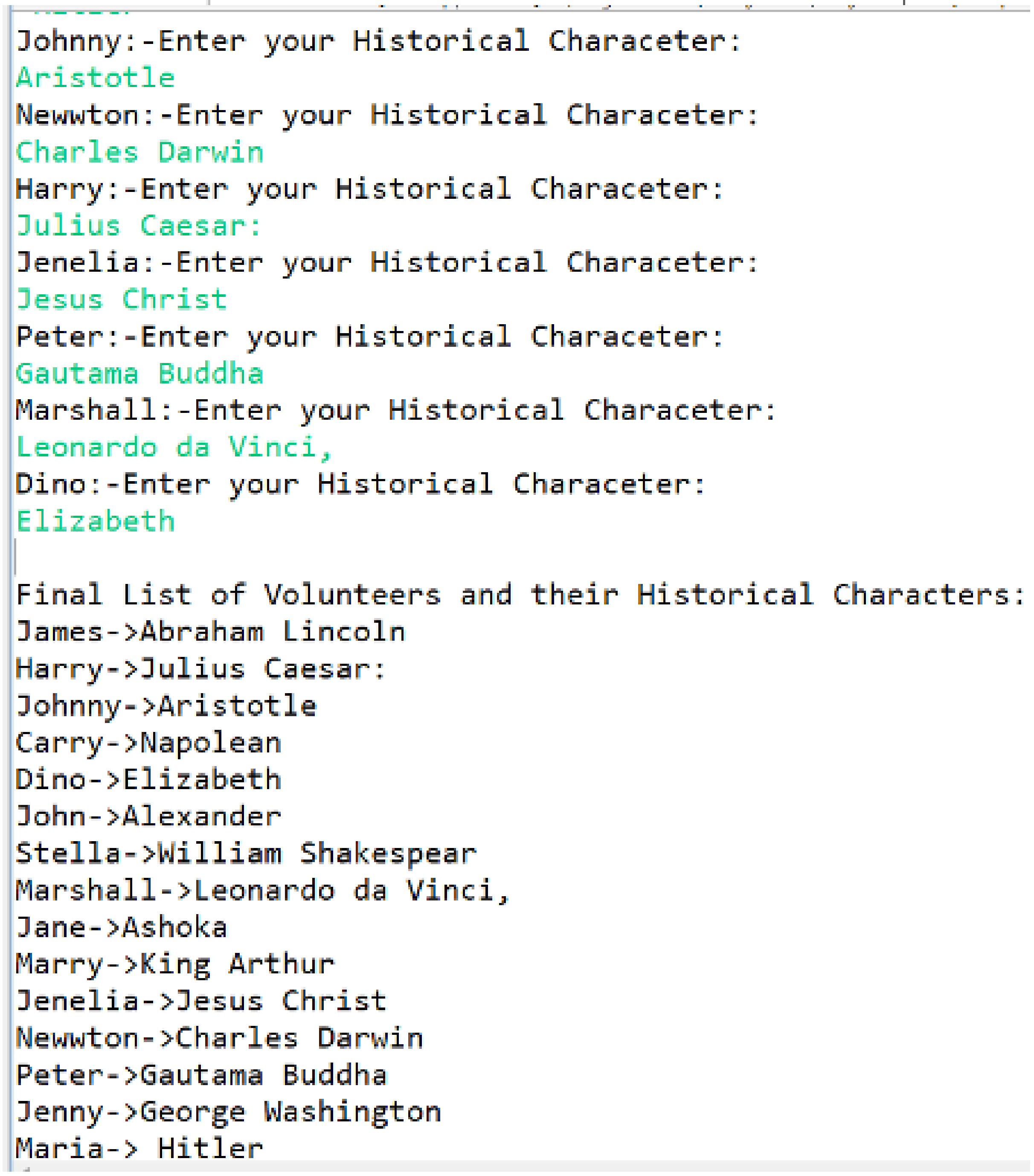
→CulturalCommitteeDemo.java

```
package pkgCulturalComittee;
import java.util.*;
public class CulturalCommitteeDemo {
public static void main(String[] args) {
CulturalComittee cc=new CulturalComittee();
cc.addElementsToCollection();
cc.addHistoricalCharacter();
cc.displayList();
}
}
```


```
<terminated> CulturalCommitteeDemo [Java Application] C:\Program Files\Java\jdk-2
Enter volunteer name:
John
John has been added to the committee.
Enter volunteer name:
Jane
Jane has been added to the committee.
Enter volunteer name:
Marry
Marry has been added to the committee.
Enter volunteer name:
Stella
Stella has been added to the committee.
Enter volunteer name:
Carry
Carry has been added to the committee.
Enter volunteer name:
James
James has been added to the committee.
Enter volunteer name:
Jenny
Jenny has been added to the committee.
Enter volunteer name:
Maria
Maria has been added to the committee.
Enter volunteer name:
Johnny
Johnny has been added to the committee.
Enter volunteer name:
Newwton
Newwton has been added to the committee.
Enter volunteer name:
```



```
Enter volunteer name:
Harry
Harry has been added to the committee.
Enter volunteer name:
Jenelia
Jenelia has been added to the committee.
Enter volunteer name:
Peter
Peter has been added to the committee.
Enter volunteer name:
Marshall
Marshall has been added to the committee.
Enter volunteer name:
Dino
Dino has been added to the committee.
No more candidates allowed as volunteers.Thank You.
John:-Enter your Historical Characeter:
Alexander
Jane:-Enter your Historical Characeter:
Ashoka
Marry:-Enter your Historical Characeter:
King Arthur
Stella:-Enter your Historical Characeter:
William Shakespear
Carry:-Enter your Historical Characeter:
Napolean
James:-Enter your Historical Characeter:
Abraham Lincoln
Jenny:-Enter your Historical Characeter:
George Washington
Maria:-Enter your Historical Characeter:
```



```
Johnny:-Enter your Historical Characeter:
Aristotle
Newwton:-Enter your Historical Characeter:
Charles Darwin
Harry:-Enter your Historical Characeter:
Julius Caesar:
Jenelia:-Enter your Historical Characeter:
Jesus Christ
Peter:-Enter your Historical Characeter:
Gautama Buddha
Marshall:-Enter your Historical Characeter:
Leonardo da Vinci,
Dino:-Enter your Historical Characeter:
Elizabeth

Final List of Volunteers and their Historical Characters:
James->Abraham Lincoln
Harry->Julius Caesar:
Johnny->Aristotle
Carry->Napolean
Dino->Elizabeth
John->Alexander
Stella->William Shakespear
Marshall->Leonardo da Vinci,
Jane->Ashoka
Marry->King Arthur
Jenelia->Jesus Christ
Newwton->Charles Darwin
Peter->Gautama Buddha
Jenny->George Washington
Maria-> Hitler
```

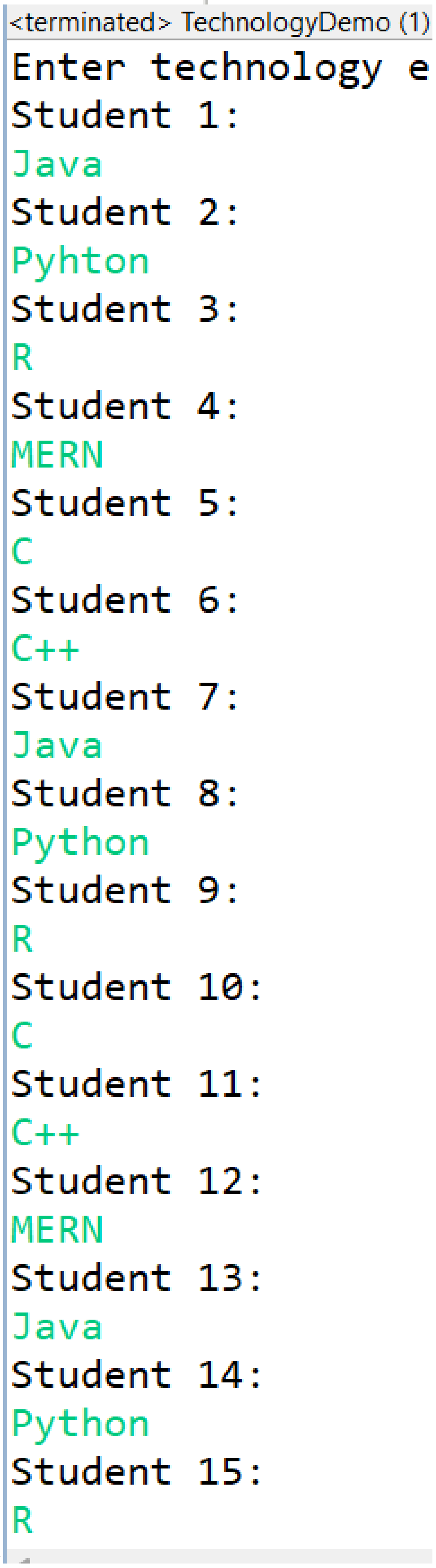
6.The placement cell of your Institute has asked you to share the name of one technology which you are expert in. Using an app, take this from 15 students. The cell then wants you to give a technology count based on the input. For ex, how many students chose Java, how many chose Python, how many entered MERN, etc. Demonstrate the use of ArrayList to achieve this .Hint : You will need to sort the arraylist and then count the individual elements pertaining to a given technology.

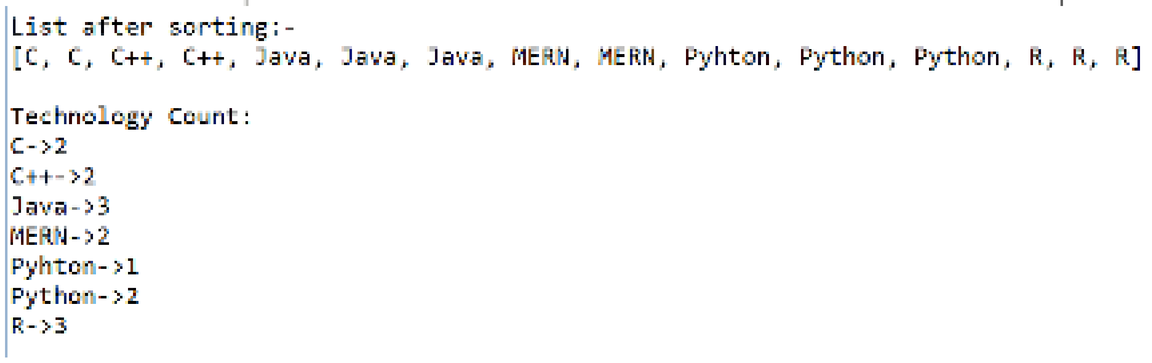
→Technology.java

```
package pkgPlacement;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Scanner;
public class Technology {
Scanner sc=new Scanner(System.in);
ArrayList<String> technologies=new ArrayList<>();
public void collectTechnology() {
//Collecting technology choices from 15 students
System.out.println("Enter technology each student is an expert in:");
for (int i=0;i<15;i++) {
System.out.println("Student "+(i+1)+": ");
String tech=sc.nextLine().trim();
technologies.add(tech);
}
}
public void sort() {
System.out.println("List after sorting:-");
Collections.sort(technologies);
System.out.println(technologies);
}
public void count() {
System.out.println("\nTechnology Count:");
int count=1;
for(int i=1;i<technologies.size();i++) {
if(technologies.get(i).equals(technologies.get(i-1))) {
count++;
}else {
System.out.println(technologies.get(i-1)+"->"+count);
count=1;
}
}
System.out.println(technologies.get(technologies.size()-1)+"->"+count);//Print last element
}
public void countAgain() {
System.out.println("\nTechnology Count:");
ArrayList<String> counted=new ArrayList<>();
for(String tech:technologies) {
if(!counted.contains(tech)){//if technology is not counted yet
int count=0;
counted.add(tech);
System.out.println(tech+": "+count);
//count occurrences of technology in list
for(String t:technologies) {
if(t.equals(tech)) {
count++;
}
}
}
}
}
}
```

➔**TechnologyDemo.java**

```
package pkgPlacement;
public class TechnologyDemo {
public static void main(String[] args) {
Technology techno=new Technology();
techno.collectTechnology();
techno.sort();
techno.count();
techno.countAgain();
}
```


```
<terminated> TechnologyDemo (1)
Enter technology e
Student 1:
Java
Student 2:
Pyhton
Student 3:
R
Student 4:
MERN
Student 5:
C
Student 6:
C++
Student 7:
Java
Student 8:
Python
Student 9:
R
Student 10:
C
Student 11:
C++
Student 12:
MERN
Student 13:
Java
Student 14:
Python
Student 15:
R
```
}



```
List after sorting:-
[C, C, C++, C++, Java, Java, Java, MERN, MERN, Pyhton, Python, Python, R, R, R]

Technology Count:
C->2
C++->2
Java->3
MERN->2
Pyhton->1
Python->2
R->3
```

7.For the recently held HR meet, the CR and LR of division A and B marked the attendance for their respective classes in separate lists. The TPO cell wants a consolidated list of FYMCA students who were present for the event. Write a program using ArrayList to mark division wise attendance first and then give the consolidated list.Hint : You will need to add the arraylists to get the final one.

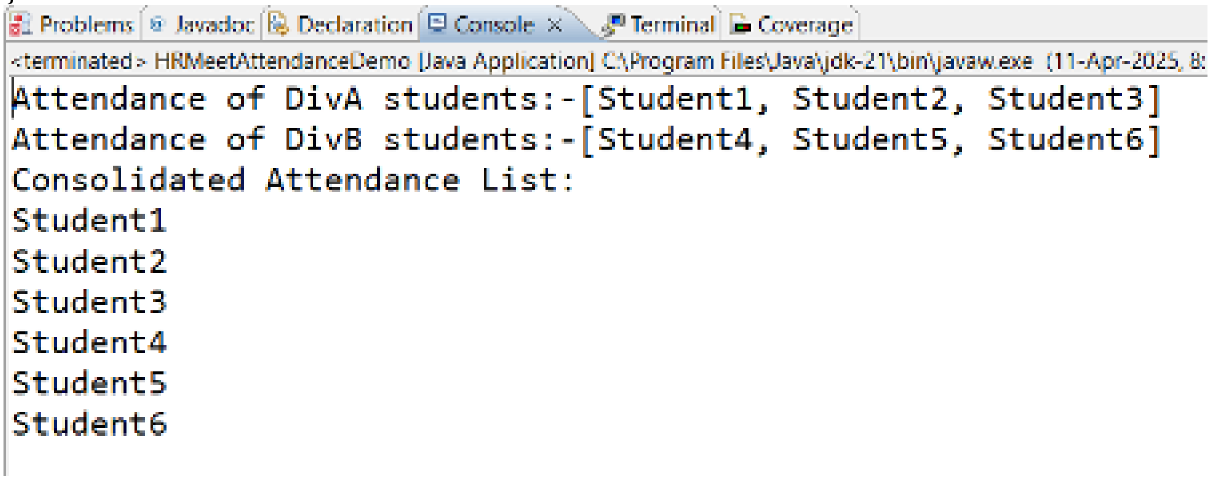
→HRMeetAttendance.java

```
package pkgPlacement;
import java.util.ArrayList;
public class HRMeetAttendance {
ArrayList<String> divA=new ArrayList<>();
ArrayList<String> divB=new ArrayList<>();
ArrayList<String> consolidatedList=new ArrayList<>();
public void markAttendance() {
//Mark attendance for Div A
divA.add("Student1");
divA.add("Student2");
divA.add("Student3");
System.out.println("Attendance of DivA students:-"+divA);
//Mark attendance for Div B
divB.add("Student4");
divB.add("Student5");
divB.add("Student6");
System.out.println("Attendance of DivB students:-"+divB);
}
public void consolidatedList() {
consolidatedList.addAll(divA);
consolidatedList.addAll(divB);
}
public void display() {
System.out.println("Consolidated Attendance List:");
for(String student:consolidatedList) {
System.out.println(student);
}
}
}
```

→HRMeetAttendanceDemo.java

```
package pkgPlacement;
```

```
public class HRMeetAttendanceDemo {
public static void main(String[] args) {
HRMeetAttendance hr=new HRMeetAttendance();
hr.markAttendance();
hr.consolidatedList();
hr.display();
}
}
```



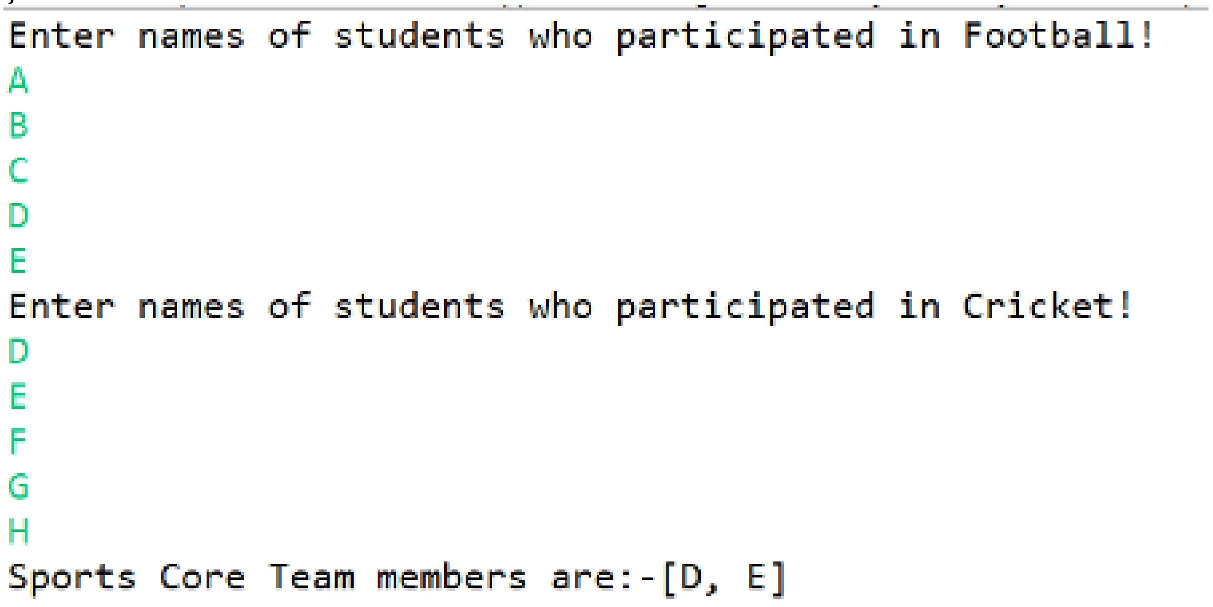
```
<terminated> HRMeetAttendanceDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (11-Apr-2025, 8:
Attendance of DivA students:-[Student1, Student2, Student3]
Attendance of DivB students:-[Student4, Student5, Student6]
Consolidated Attendance List:
Student1
Student2
Student3
Student4
Student5
Student6
```

8.Sports cell of the Institute needs to choose its core team from those students who participated in the recently held sports events. For this, the sports coordinator has decided to consider the participants of Football and Cricket. Only those players who participated in BOTH these games will be considered for the core team. Using ArrayList, write a Java program which will take the names of the students participating in Football and Cricket. Find the common names in these two events and put them into a third list, called, SportsCoreTeam.Hint : You will need to compare the two arraylists to get the third one →SportsCoreTeam.java

```
package pkgSports;
import java.util.ArrayList;
import java.util.Scanner;
public class SportsCoreTeam {
ArrayList<String> footballplayers=new ArrayList<>();
ArrayList<String> cricketplayers=new ArrayList<>();
ArrayList<String> sportscoreteam=new ArrayList<>();
Scanner sc=new Scanner(System.in);
public void collectNames() {
System.out.println("Enter names of students who participated in Football!");
for(int i=0;i<5;i++) {
String fname=sc.nextLine();
footballplayers.add(fname);
}
System.out.println("Enter names of students who participated in Cricket!");
for(int i=0;i<5;i++) {
String cname=sc.nextLine();
cricketplayers.add(cname);
}
}
public void  makeCoreTeam() {
for(String player:footballplayers) {
if(cricketplayers.contains(player)) {
sportscoreteam.add(player);
}
}
System.out.println("Sports Core Team members are:-"+sportscoreteam);
}
}
```

→SportsCoreTeamDemo.java

```
package pkgSports;
public class SportsCoreTeamDemo {
public static void main(String[] args) {
SportsCoreTeam spc=new SportsCoreTeam();
spc.collectNames();
spc.makeCoreTeam();
}
}
```



```
Enter names of students who participated in Football!
A
B
C
D
E
Enter names of students who participated in Cricket!
D
E
F
G
H
Sports Core Team members are:-[D, E]
```

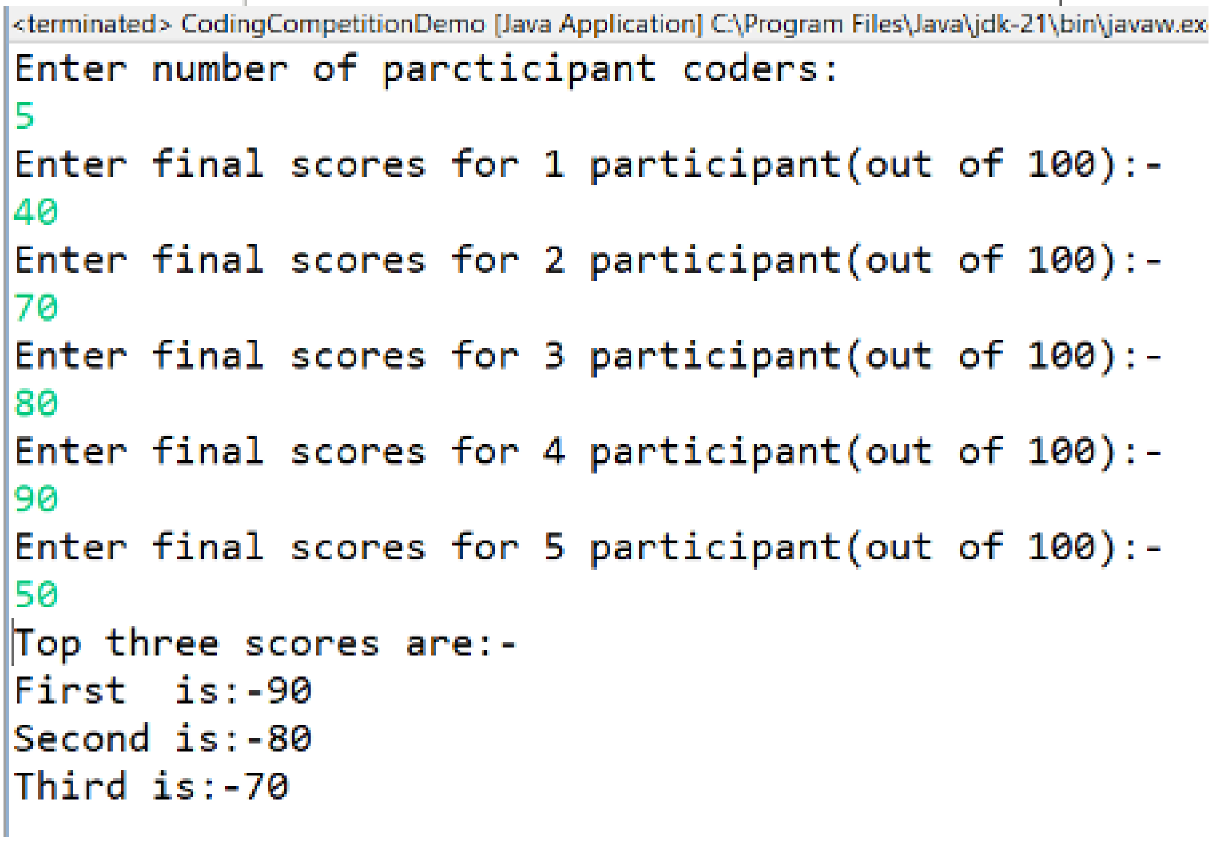
9.The top three scorers in the coding competition will be given a certificate and trophy by the Coding Club. Using Vector, take the final scores (out of 100) of the participating coders and find the top three using only the max() function available in Collections. Hint : There is no need to sort the vector.

→CodingCompetition.java

```
package pkgCodingClub;
import java.util.Collections;
import java.util.Scanner;
import java.util.Vector;
public class CodingCompetition {
Vector<Integer> scores=new Vector<>();
Scanner sc=new Scanner(System.in);
public void addDetails() {
System.out.println("Enter number of parcticipant coders:");
int totalno=sc.nextInt();
for(int i=0;i<totalno;i++) {
System.out.println("Enter final scores for "+(i+1)+" participant"+"(out of 100):-");
int finalscores=sc.nextInt();
scores.add(finalscores);
}
}
public void topthreescores() {
if(scores.size()<3) {
System.out.println("Not enough participants to determine top three");
}
else {
//firstMax
int firstMax=Collections.max(scores);
scores.remove((Integer)firstMax);//Integer is used because without it will remove element at the
given index not the value
//secondmax
int secondmax=Collections.max(scores);
scores.remove((Integer)secondmax);
//thirdmax
int thirdmax=Collections.max(scores);
System.out.println("Top three scores are:-");
System.out.println("First  is:-"+firstMax);
System.out.println("Second is:-"+secondmax);
System.out.println("Third is:-"+thirdmax);
}
}
}
```

→CodingCompetitionDemo.java

```
package pkgCodingClub;
public class CodingCompetitionDemo {
public static void main(String[] args) {
CodingCompetition cc=new CodingCompetition();
cc.addDetails();
cc.topthreescores();
}
}
```



```
<terminated> CodingCompetitionDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.ex
Enter number of parcticipant coders:
5
Enter final scores for 1 participant(out of 100):-
40
Enter final scores for 2 participant(out of 100):-
70
Enter final scores for 3 participant(out of 100):-
80
Enter final scores for 4 participant(out of 100):-
90
Enter final scores for 5 participant(out of 100):-
50
Top three scores are:-
First  is:-90
Second is:-80
Third is:-70
```

10.KKR and MumbaiIndians are going to play the kickstart match of this year's IPL season. Using Vector, you have taken the names of the players in each team and are displaying the same. But there is a last minute change in the batting line up of KKR. In place of QuintonDeKock, the team will send Anukul Roy at two down position. Make this change in their batting line up and display the new order.Hint : You need to get the original element and set it with the new one.

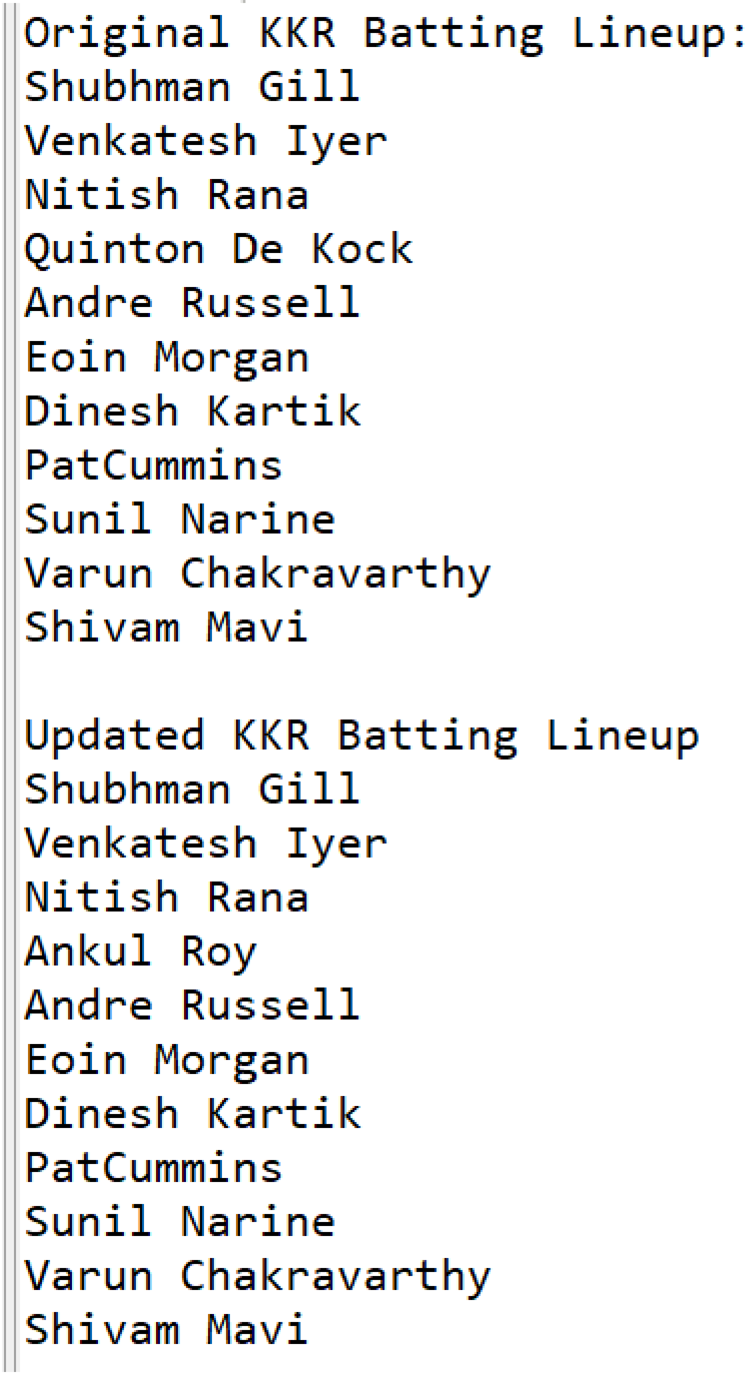
→KKR.java

```
package pkgIPL;
import java.util.Vector;
public class KKR {
Vector<String> KKRplayers=new Vector<>();
public void addnames() {
KKRplayers.add("Shubhman Gill");
KKRplayers.add("Venkatesh Iyer");
KKRplayers.add("Nitish Rana");
KKRplayers.add("Quinton De Kock");
KKRplayers.add("Andre Russell");
KKRplayers.add("Eoin Morgan");
KKRplayers.add("Dinesh Kartik");
KKRplayers.add("PatCummins");
KKRplayers.add("Sunil Narine");
KKRplayers.add("Varun Chakravarthy");
KKRplayers.add("Shivam Mavi");
}
public void displayOriginal() {
```

```
System.out.println("Original KKR Batting Lineup:");
for(String p:KKRplayers) {
System.out.println(p);
}
}
public void replace() {
int pos=3;//index for quinton de kock
KKRplayers.set(pos, "Ankul Roy");
}
public void displayUpdated() {
System.out.println("\nUpdated KKR Batting Lineup");
for(String p:KKRplayers) {
System.out.println(p);
}
}
}
```

→KKRdemo.java

```
package pkgIPL;
public class KKRdemo {
public static void main(String[] args) {
KKR kkrteam=new KKR();
kkrteam.addnames();
kkrteam.displayOriginal();
kkrteam.replace();
kkrteam.displayUpdated();
}
}
```



```
Original KKR Batting Lineup:
Shubhman Gill
Venkatesh Iyer
Nitish Rana
Quinton De Kock
Andre Russell
Eoin Morgan
Dinesh Kartik
PatCummins
Sunil Narine
Varun Chakravarthy
Shivam Mavi

Updated KKR Batting Lineup
Shubhman Gill
Venkatesh Iyer
Nitish Rana
Ankul Roy
Andre Russell
Eoin Morgan
Dinesh Kartik
PatCummins
Sunil Narine
Varun Chakravarthy
Shivam Mavi
```

11.In the e-commerce portal designed by you, the customer adds products to the shopping cart. Use a vector to hold the objects of Product class. At the time of billing, access each product object and read its price. Add the cost of all the products and display the bill total. If the cart is empty, show a message, "Can we help you in finding what you were looking for?" and end the billing process.Hint : You will need a Product class and its objects. You will also need to check if the vector holding the cart is empty or not?

→ProductClass.java

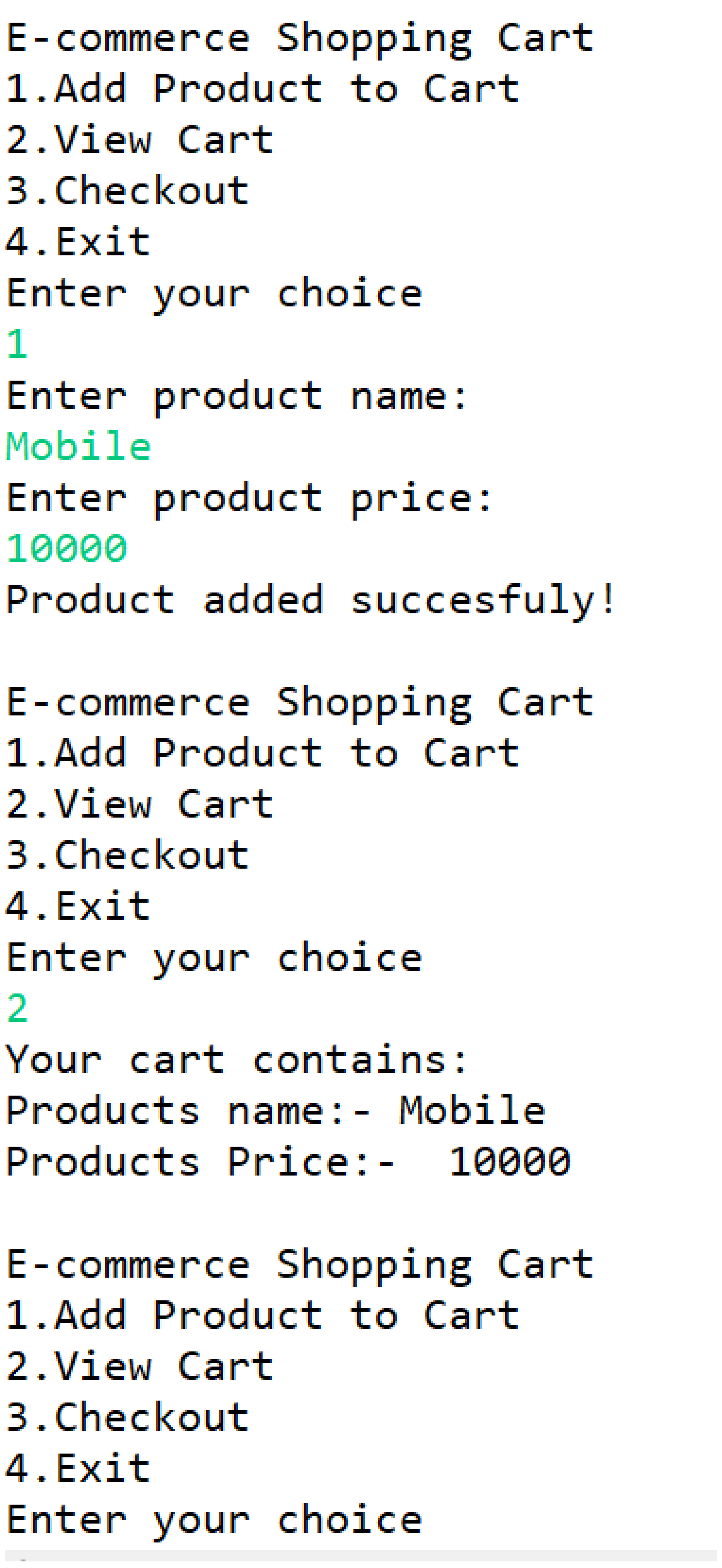
```
package pkgEcom;
public class ProductClass {
String name;
int price;
public ProductClass(String name,int price){
this.name=name;
this.price=price;
}
public void displayProduct() {
System.out.println("Product name:- "+name);
System.out.println("Price:- "+price);
}
}
```

→ShoppingCart.java

```
package pkgEcom;
import java.util.Vector;
public class ShoppingCart {
Vector<ProductClass> cart=new Vector<>();
public void addProducts(ProductClass product) {
cart.add(product);
System.out.println("Product added succesfuly!");
}
public void displayCart() {
if(cart.isEmpty()) {
System.out.println("Your cart is empty!");
}
else {
System.out.println("Your cart contains:");
for(ProductClass product:cart) {
System.out.println("Products name:- "+product.name);
System.out.println("Products Price:-  "+product.price);
}
}
}
public void checkout() {
int totalBill=0;
if(cart.isEmpty()) {
System.out.println("Can we help you in finding what you were looking for?");
return;//ends billing process
}
else {
System.out.println("Billing Details:-");
for(ProductClass product:cart) {
System.out.println("Product:- "+product.name+"| Product price:- "+product.price);
totalBill+=product.price;
}
System.out.println("Total Bill Amount :-"+totalBill);
}
}
}
```

→ProductClassDemo.java

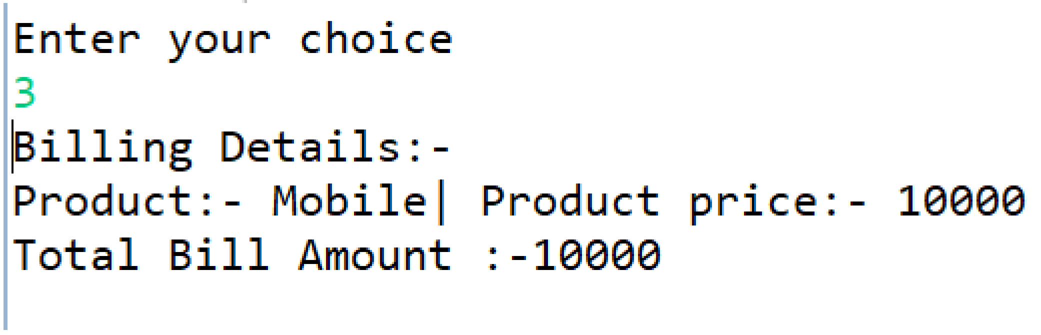
```
package pkgEcom;
import java.util.Scanner;
public class ProductClassDemo {
public static void main(String[] args) {
ShoppingCart shopcart=new ShoppingCart();
Scanner sc=new Scanner(System.in);
while(true) {
System.out.println("\nE-commerce Shopping Cart");
System.out.println("1.Add Product to Cart");
System.out.println("2.View Cart");
System.out.println("3.Checkout");
System.out.println("4.Exit");
System.out.println("Enter your choice");
int choice=sc.nextInt();
sc.nextLine();
switch(choice) {
case 1:
//Add products to cart
System.out.println("Enter product name: ");
String name=sc.nextLine();
System.out.println("Enter product price: ");
int price=sc.nextInt();
sc.nextLine();
ProductClass product=new ProductClass(name,price);
shopcart.addProducts(product);
break;
case 2:
//View cart
shopcart.displayCart();
break;
case 3:
//Checkout Process
shopcart.checkout();
sc.close();
return;//end program after checkout
case 4:
//Exit application
System.out.println("Thank you for visiting !Have a great day.");
sc.close();
return;
default:
System.out.println("Invalid choice!Please try again.");
}
}
}
}
```



```
E-commerce Shopping Cart
1.Add Product to Cart
2.View Cart
3.Checkout
4.Exit
Enter your choice
1
Enter product name:
Mobile
Enter product price:
10000
Product added succesfuly!

E-commerce Shopping Cart
1.Add Product to Cart
2.View Cart
3.Checkout
4.Exit
Enter your choice
2
Your cart contains:
Products name:- Mobile
Products Price:-  10000

E-commerce Shopping Cart
1.Add Product to Cart
2.View Cart
3.Checkout
4.Exit
Enter your choice
```



```
Enter your choice
3
Billing Details:-
Product:- Mobile| Product price:- 10000
Total Bill Amount :-10000
```

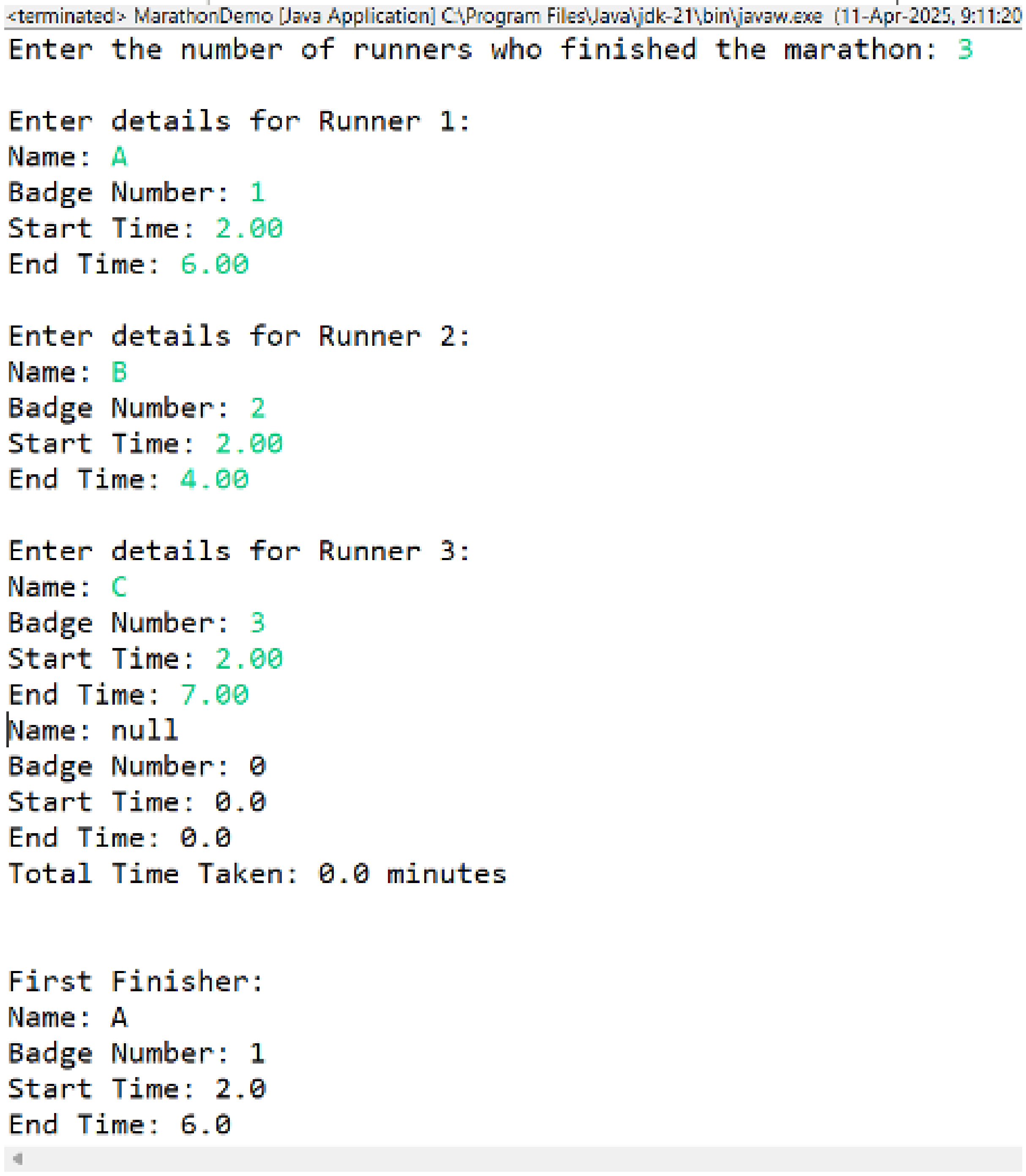
12.During the Marathon event the organisers maintained a list to hold the details of the finishers. Once the marathon got over, they displayed the details of the first runner to finish the marathon and the last one to finish the same. Write an app having the objects of MarathonRunner class in to a vector list, finishers. Display the details of the runner who comes first and of the who comes last. MarathonRunner class has the properties, name, badgeNbr, startTime and endTime. Hint : You will need to check the first and last element in the vector.

→MarathonRunner.java

```java
package pkgMarathon;
import java.util.Scanner;
import java.util.Vector;
public class MarathonRunner {
String name;
int badgeNbr;
double startTime;
double endTime;
Scanner sc = new Scanner(System.in);
Vector<MarathonRunner> finishers = new Vector<>();
// Constructor
public MarathonRunner(String name, int badgeNbr, double startTime, double endTime) {
this.name = name;
this.badgeNbr = badgeNbr;
this.startTime = startTime;
this.endTime = endTime;
}
//Default Constructor
public MarathonRunner() {
}
// Method to calculate total time taken
public double getTotalTime() {
return endTime - startTime;
}
// Accept runner details
public void acceptDetails() {
System.out.print("Enter the number of runners who finished the marathon: ");
int totalRunners = sc.nextInt();
sc.nextLine(); // Consume newline
for (int i = 0; i < totalRunners; i++) {
System.out.println("\nEnter details for Runner " + (i + 1) + ":");
System.out.print("Name: ");
String name = sc.nextLine();
System.out.print("Badge Number: ");
int badgeNbr = sc.nextInt();
sc.nextLine(); // Consume newline
System.out.print("Start Time: ");
double startTime = sc.nextDouble();
sc.nextLine(); // Consume newline
System.out.print("End Time: ");
double endTime = sc.nextDouble();
sc.nextLine(); // Consume newline
// Validate that End Time > Start Time
if (endTime < startTime) {
System.out.println("Error: End Time must be greater than Start Time. Please re-enter details.");
i--; // Retry current iteration
continue;
}
//Add runner to vector
finishers.add(new MarathonRunner(name,badgeNbr,startTime,endTime));
}
}
// Display runner details
public void displayRunnerDetails() {
System.out.println("Name: " + name);
System.out.println("Badge Number: " + badgeNbr);
System.out.println("Start Time: " + startTime);
System.out.println("End Time: " + endTime);
System.out.println("Total Time Taken: " + getTotalTime() + " minutes\n");
}
// Display first and last finisher
public void displayFirstLast() {
if (finishers.isEmpty()) {
System.out.println("No runners finished the marathon.");
} else {
System.out.println("\nFirst Finisher:");
finishers.firstElement().displayRunnerDetails();
System.out.println("Last Finisher:");
finishers.lastElement().displayRunnerDetails();
}
}
}
```

→MarathonDemo.java

```
package pkgMarathon;
public class MarathonDemo {
public static void main(String[] args) {
MarathonRunner runner=new MarathonRunner();
runner.acceptDetails();
runner.displayRunnerDetails();
runner.displayFirstLast();
}
}
```



```
<terminated> MarathonDemo [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (11-Apr-2025, 9:11:20
Enter the number of runners who finished the marathon: 3

Enter details for Runner 1:
Name: A
Badge Number: 1
Start Time: 2.00
End Time: 6.00

Enter details for Runner 2:
Name: B
Badge Number: 2
Start Time: 2.00
End Time: 4.00

Enter details for Runner 3:
Name: C
Badge Number: 3
Start Time: 2.00
End Time: 7.00
Name: null
Badge Number: 0
Start Time: 0.0
End Time: 0.0
Total Time Taken: 0.0 minutes


First Finisher:
Name: A
Badge Number: 1
Start Time: 2.0
End Time: 6.0
```

```
First Finisher:
Name: A
Badge Number: 1
Start Time: 2.0
End Time: 6.0
Total Time Taken: 4.0 minutes

Last Finisher:
Name: C
Badge Number: 3
Start Time: 2.0
End Time: 7.0
Total Time Taken: 5.0 minutes
```