# Lab Assignment 09

BRAC UNIVERSITY

Inspiring Excellence

| | |
|---|---|
| **Course Code:** | **CSE111** |
| **Course Title:** | **Programming Language II** |
| **Topic:** | **Inheritance, Polymorphism** |
| **Number of Tasks:** | **10 (Classwork: 04, Homework: 06)** |

***[Submit all the Coding Tasks (Homework: Task 1 to 3) in the Google Form shared on buX before the next lab.]***

**[You are not allowed to change the driver codes of any of the tasks]**

## CLASSWORK

### Task 1

```java
public class Bank {
 public static int rs = 3;
 public static int cd = -8;
 public int bl = 0;
 public int br = 0;
 public Bank() {
   br = rs - 2;
   bl = rs + 1;
   rs -= 2;
 }
 public void deposit(int a, int b) {
   int cd = 0;
   br = br + a + (rs++);
   cd = cd + 1 + b;
   bl = bl + cd + br;
   System.out.println(cd + " " + br + " " + bl);
 }
}
public class Account extends Bank {
 public static int cd = 1;
 public int bl = -4;
 public Account() {
   bl = 0;
   br = rs + 3;
   super.bl = 2 + rs + 3;
   rs -= 2;
 }
 public Account(Account acc) {
   bl = acc.bl + super.bl;
   cd = acc.cd;
   acc.withdraw(2, 3);
 }
 public void withdraw(int a, int b) {
   int br = 0;
   br = br + this.br;
   cd = br + 2 + (++rs);
   deposit(cd, br);
   bl = cd + br + bl;
   System.out.println(cd + " " + br + " " + bl);
 }
}
```

Write the output of the following code:

```
public class Tester {
   public static void main(String[] args) {
       Bank b1 = new Bank();
       Account a1 = new Account();
       Account a2 = new Account(a1);
       a1.deposit(3, 4);
       a2.withdraw(1, 6);
   }
}
```

| Output:- | | |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

## Task 2

Design the **CinemexTicket** class derived from the MovieTicket Class so that the given output is produced:

- ❖ The seatTypes and seatPrices arrays contain the type of the seat and its corresponding price
- ❖ Night show charge (15% of ticket price) will be applicable if the time is between 6:00 PM - 11:00 PM
- ❖ Unique id for a ticket is generated by: MovieName-FirstLetterOfSeatType-TicketCount
- ❖ You may need to use .split() and Integer.parseInt() built-in methods

**Parent Class**

```
public class MovieTicket {
  public static String [] seatTypes = {"Regular", "Premium", "IMAX 3D"};
  public static double [] seatPrices = {300.0, 450.0, 600.0};
  public static int nightShowCharge = 15;
  private String movie;
  public String showtime;
  public String date;
  private double price;
  public String seat;
  public MovieTicket(String movie, String date, String showtime, double price) {
    this.movie = movie;
    this.showtime = showtime;
    this.date = date;
    this.price = price;
    this.seat = "Not Selected";
  }
  public void setPrice(double price) {
    this.price = price;
  }
  public double getPrice() {
    return price;
  }
  public String getMovie() {
    return movie;
  }
  public String toString() {
    return "Movie: " + movie + "\nShowtime: " + showtime + "\nDate: " + date;
  }
}
```

**Driver Code**

```java
public class Tester {
 public static void main(String[] args) {
  CinemexTicket ticket1 = new CinemexTicket("Deadpool
and Wolverine", "18:30", "Action-Comedy", "July 24,
2024");
  System.out.println("Total movie ticket(s): " +
CinemexTicket.getTotalTickets());
  System.out.println("1============================");
  ticket1.calculateTicketPrice();
  System.out.println("2============================");
  System.out.println(ticket1);
  System.out.println("3============================");
  System.out.println(ticket1.confirmPayment());
  System.out.println("4============================");
  System.out.println(ticket1);
  System.out.println("5============================");
  CinemexTicket ticket2 = new CinemexTicket("Twisters",
"10:00", "Sci-Fi", "August 10, 2024", "Premium");
  System.out.println("Total movie ticket(s): " +
CinemexTicket.getTotalTickets());
  System.out.println("6============================");
  ticket2.calculateTicketPrice();
  System.out.println("7============================");
  System.out.println(ticket2.confirmPayment());
  System.out.println("8============================");
  System.out.println(ticket2);
  System.out.println("9============================");
  System.out.println(ticket2.confirmPayment());
 }
}
```

**Output**

```
Total movie ticket(s): 1
1============================
Ticket price is calculated
successfully.
2============================
Ticket ID: Deadpool and
Wolverine-R-1
Movie: Deadpool and Wolverine
Showtime: 18:30
Date: July 24, 2024
Genre: Action-Comedy
Seat Type: Regular
Price(tk): 345.0
Status: Not Paid
3============================
Payment Successful.
4============================
Ticket ID: Deadpool and
Wolverine-R-1
Movie: Deadpool and Wolverine
Showtime: 18:30
Date: July 24, 2024
Genre: Action-Comedy
Seat Type: Regular
Price(tk): 345.0
Status: Paid
5============================
Total movie ticket(s): 2
6============================
Ticket price is calculated
successfully.
7============================
Payment Successful.
8============================
Ticket ID: Twisters-P-2
Movie: Twisters
Showtime: 10:00
Date: August 10, 2024
Genre: Sci-Fi
Seat Type: Premium
Price(tk): 450.0
Status: Paid
9============================
Ticket price is already paid!
```

## Task 3

Write the **Mango** and the **Jackfruit** classes derived from Fruit class so that the following code generates the output below:

**Parent Class**

```java
public class Fruit{
  private boolean formalin = false;
  private String name = "";
  public Fruit(boolean formalin, String name){
    this.formalin = formalin;
    this.name = name;
  }
  public String getName(){
    return name;
  }
  public boolean hasFormalin(){
    return formalin;
  }
}
```

**Driver Code**

```java
public class FruitTester{
  public static void testFruit(Fruit f){
    System.out.println("----Printing Detail-----");
    if(f.hasFormalin()){
      System.out.println("Do not eat the "+f.getName()+".");
      System.out.println(f);
    }else{
      System.out.println("Eat the "+f.getName()+".");
      System.out.println(f);
    }
  }
  public static void main(String [] args){
    Mango m = new Mango();
    testFruit(m);
    Jackfruit j = new Jackfruit();
    testFruit(j);
  }
}
```

**Output**

```
----Printing Detail-----
Do not eat the Mango.
Mangos are bad for you
----Printing Detail-----
Eat the Jackfruit.
Jackfruits are good for you
```

## Task 4

```
public class Caramel extends SilkOreo{
  String texture = "Softy";
  public void method1() {
    System.out.println("Caramel m1");
  }
  public void method4() {
  System.out.println("Caramel m4");
  }
  public String toString(){
    method2();
    return "Caramel is "+ texture;
  }
}
public class Chocolate{
  String texture = "Chocolaty";
  public void method1() {
    method2();
    System.out.println("Chocolate m1");
  }
  public void method2() {
    System.out.println("Chocolate m2");
  }
  public String toString(){
    method2();

    return "Chocolate is "+ texture;
  }
}
public class DairyMilk extends Chocolate{
  String texture = "Yummy";
  public void method2() {
    System.out.println(this.texture);
    System.out.println("DairyMilk m2");
  }
  public void method3() {
    System.out.println("DairyMilk m3");
  }
}
public class KitKat extends Chocolate{
  String texture = "Crunchy";
  public void method1() {
    System.out.println("KitKat m1");
  }
  public void method4() {
    System.out.println("KitKat m4");
  }
  public String toString(){
    method2();
    return "KitKat is "+ texture;
  }
}
public class SilkOreo extends DairyMilk{
  String texture = "Silky";
  public void method1() {
    super.method1();
    System.out.println("SilkOreo m1");
  }
  public void method3() {
    System.out.println("SilkOreo m3");
    System.out.println(this);
  }
}
```

**Assuming the following variables have been defined:**

```
Chocolate choco1 = new Chocolate();
KitKat kit = new KitKat();
DairyMilk dairyMilk1 = new DairyMilk();
DairyMilk dairyMilk2 = new SilkOreo();
Object obj1 = new DairyMilk();
Object obj2 = new KitKat();
Chocolate caramel1 = new Caramel();
```

In the table below,

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

| | Statement | Output |
|---|---|---|
| 1 | choco1.method1(); | |
| 2 | dairyMilk1.method1(); | |
| 3 | dairyMilk2.method4(); | |
| 4 | caramel1.method1(); | |
| 5 | System.out.println(caramel1); | |
| 6 | System.out.println(caramel1.texture); | |
| 7 | ((Chocolate)kit).method2(); | |
| 8 | ((SilkOreo)dairyMilk2).method3(); | |
| 9 | ((DairyMilk)kit).method2(); | |
| 10 | ((Chocolate)kit).method3(); | |
| 11 | ((Chocolate)dairyMilk2).method1(); | |
| 12 | ((Chocolate)obj1).method2(); | |
| 13 | ((Caramel)obj1).method2(); | |
| 14 | ((SilkOreo)obj2).method3(); | |
| 15 | System.out.println(((Object)choco1).toString()); | |
| 16 | System.out.println(((Chocolate)kit).texture); | |

# HOMEWORK

## Task 1

Design the **Manager** and **Developer** class derived from the **Employee** class with appropriate attributes and properties so that the driver code can generate the output given below. [**Hint**:
Manager:
1. Adds a bonus to the base salary if the manager works more than 40 hours.
2. If the manager works more than 100 hours, the full amount is approved; if they work more than 80 hours, half the amount is approved. Otherwise, the increment is denied.

Developer:
1. Adds $700 to the base salary if the developer works with Java programming language.]

**Driver Code and Parent Class**

```java
public class Employee {
    public String name;
    private double baseSalary;
    private int hoursWorked;

    public Employee(String name, double baseSalary, int hoursWorked){
        this.name = name;
        this.baseSalary = baseSalary;
        this.hoursWorked = hoursWorked;
    }
    public double getBaseSalary() {
        return baseSalary;
    }
    public void setBaseSalary(double baseSalary) {
        this.baseSalary = baseSalary;
    }
    public int getHoursWorked() {
        return hoursWorked;
    }
    public void setHoursWorked(int hoursWorked) {
        this.hoursWorked = hoursWorked;
    }
    public void displayInfo() {
        System.out.println("Name: " + name);
        System.out.println("Base Salary: $" + baseSalary);
        System.out.println("Work Hours: " + hoursWorked);
    }
}
public class EmployeeTester {
 public static void main(String[] args) {
  Manager neymar = new Manager("Neymar",1000, 45, 10);
  Developer messi = new Developer("Messi", 1000, 50, "Java");
  Developer chiesa = new Developer("Chiesa", 1000, 50, "Javascript");
  neymar.calculateSalary();
  System.out.println("1.==========");
  neymar.displayInfo();
```

**Output**

```
1.==========
Name: Neymar
Base Salary: $1000.0
Work Hours: 45
Bonus: 10.0 %
Final Salary: $1100.0
2.==========
Increment denied.
3.==========
$50 Increment approved.
4.==========
5.==========
Name: Neymar
Base Salary: $1050.0
Work Hours: 85
Bonus: 10.0 %
Final Salary: $1155.0
6.==========
7.==========
Name: Messi
Base Salary: $1000.0
Work Hours: 50
Language: Java
Final Salary: $1700.0
8.==========
9.==========
Name: Chiesa
Base Salary: $1000.0
Work Hours: 50
Language: Javascript
Final Salary: $1000.0
```

```java
  System.out.println("2.==========");
  neymar.requestIncrement(100);
  System.out.println("3.==========");
  neymar.setHoursWorked(85);
  neymar.requestIncrement(100);
  System.out.println("4.==========");
  neymar.calculateSalary();
  System.out.println("5.==========");
  neymar.displayInfo();
  System.out.println("6.==========");
  messi.calculateSalary();
  System.out.println("7.==========");
  messi.displayInfo();
  System.out.println("8.==========");
  chiesa.calculateSalary();
  System.out.println("9.==========");
  chiesa.displayInfo();
 }
}
```

## Task 2

Design the **KKTea** (parent) and **KKFlavouredTea** (child) classes so that the following output is produced. **The KKFlavouredTea class should inherit KKTea and KKTea should inherit the Tea class.** Note that:

- An object of either class represents a single box of teabags.
- Each tea bag weighs 2 grams.
- The status of an object refers to whether it is sold or not

**Driver Code and Parent Class**

```java
public class Tea {
  public String name;
  protected int price;
  protected boolean status;

  public Tea(String name, int price) {
    this.name = name;
    this.price = price;
    this.status = false;
  }

  public void productDetail() {
    System.out.println("Name: " + name + ", Price: " + price);
    System.out.println("Status: " + status);
  }
}
//Driver Code
public class TeaTester{
 public static void main(String[] args) {
   KKTea t1 = new KKTea(250, 50);
   System.out.println("--------1---------");
   t1.productDetail();
   System.out.println("--------2---------");
   KKTea.totalSales();
   System.out.println("--------3---------");
   KKTea t2 = new KKTea(470, 100);
   KKTea t3 = new KKTea(360, 75);
   KKTea.updateSoldStatusRegular(t1);
   KKTea.updateSoldStatusRegular(t2);
   System.out.println("--------4---------");
   t2.productDetail();
   System.out.println("--------5---------");
   KKTea.totalSales();
   System.out.println("--------6---------");
   KKFlavouredTea t4 = new KKFlavouredTea("Jasmine", 260, 50);
   KKFlavouredTea t5 = new KKFlavouredTea("Honey Lemon", 270, 45);
   KKFlavouredTea t6 = new KKFlavouredTea("Honey Lemon", 270, 45);
   System.out.println("--------7---------");
   t4.productDetail();
   System.out.println("--------8---------");
   t6.productDetail();
   System.out.println("--------9---------");
   KKFlavouredTea.updateSoldStatusFlavoured(t4);
   KKFlavouredTea.updateSoldStatusFlavoured(t5);
   KKFlavouredTea.updateSoldStatusFlavoured(t6);
   System.out.println("--------10---------");
   KKTea.totalSales();
 }
}
```

**Output**

```
--------1---------
Name: KK Regular Tea, Price: 250
Status: false
Weight: 100, Tea Bags: 50
--------2---------
Total Sales: 0
KK Regular Tea: 0
--------3---------
--------4---------
Name: KK Regular Tea, Price: 470
Status: true
Weight: 200, Tea Bags: 100
--------5---------
Total Sales: 2
KK Regular Tea: 2
--------6---------
--------7---------
Name: KK Jasmine Tea, Price: 260
Status: false
Weight: 100, Tea Bags: 50
--------8---------
Name: KK Honey Lemon Tea, Price: 270
Status: false
Weight: 90, Tea Bags: 45
--------9---------
--------10---------
Total Sales: 5
KK Regular Tea: 2
KK Flavoured Tea: 3
```

## Task 3

Write the **CSEStudent** and **CSE111Student** classes derived from **Student** class so that the following code generates the output below:

**Parent Class**

```java
public class Student{
  public String msg = "I love BU";
  public String shout(){
    return msg;
  }
}
```

**Driver Code**

```java
public class StudentTester{
  public static void printShout(Student s){
    System.out.println("------------------");
    System.out.println(s.msg);
    System.out.println(s.shout());
  }
  public static void main(String [] args){
    Student s = new Student();
    CSEStudent cs = new CSEStudent();
    CSE111Student cs111 = new CSE111Student();
    System.out.println(s.msg);
    System.out.println(cs.msg);
    System.out.println(cs111.msg);
    printShout(s);
    printShout(cs);
    printShout(cs111);
  }
}
```

**Output**

```
I love BU
I want to transfer to CSE
I love Java Programming
------------------
I love BU
I love BU
------------------
I love BU
I want to transfer to CSE
------------------
I love BU
I love Java Programming
```

## Task 4

```java
public class Gandalf {
  public void method1(){
    System.out.println("Gandalf 1");
  }

  public void method2(){
    System.out.println("Gandalf 2");
    method1();
  }
}
public class Bilbo extends Gandalf{
  public void method1(){
    System.out.println("Bilbo 1");
  }
}
public class Gollum extends Gandalf{
  public void method3(){
    System.out.println("Gollum 3");
  }
}
public class Frodo extends Bilbo{
  public void method1(){
    System.out.println("Frodo 1");
    super.method1();
  }

  public void method3(){
    System.out.println("Frodo 3");
  }
}
```

**Assuming the following variables have been defined:**

```java
Gandalf var1 = new Frodo();
Gandalf var2 = new Bilbo();
Gandalf var3 = new Gandalf();
Object var4 = new Bilbo();
Bilbo var5 = new Frodo();
Object var6 = new Gollum();
```

In the table below,

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be Detected.

| | **Statement** | **Output** |
|---|---|---|
| 1 | var1.method1(); | |
| 2 | var2.method1(); | |
| 3 | var4.method1(); | |
| 4 | var6.method1(); | |
| 5 | var1.method2(); | |
| 6 | var3.method2(); | |
| 7 | var4.method2(); | |
| 8 | var5.method2(); | |
| 9 | var6.method2(); | |
| 10 | ((Frodo)var4).method3(); | |
| 11 | ((Frodo)var6).method2(); | |
| 12 | ((Gollum)var1).method3(); | |
| 13 | ((Gollum)var4).method1(); | |
| 14 | ((Gandalf)var1).method2(); | |
| 15 | ((Frodo)var4).method1(); | |
| 16 | ((Gollum)var6).method2(); | |
| 17 | ((Gandalf)var2).method1(); | |
| 18 | ((Bilbo)var6).method2(); | |
| 19 | ((Frodo)var1).method3(); | |
| 20 | ((Gandalf)var5).method3(); | |

## Task 5

```java
public class Sue {
 void method1() {
   System.out.println("sue 1");
 }
 void method3() {
   System.out.println("sue 3");
 }
}

public class Blue {
 void method1() {
   System.out.println("blue 1");
   method3();
 }
 void method3() {
   System.out.println("blue 3");
 }
}

public class Moo extends Blue {
 void method2() {
   super.method3();
   System.out.println("moo 2");
   this.method3();
 }
 void method3() {
   System.out.println("moo 3");
 }
}

public class Crew extends Moo {
 void method1() {
   System.out.println("crew 1");
 }
 void method3() {
   System.out.println("crew 3");
 }
}
```

**Assuming the following variables have been defined:**

```java
Moo var1 = new Crew();
```

```
Blue var2 = new Moo();
Object var3 = new Sue();
Sue var4 = new Sue();
Blue var5 = new Crew();
Blue var6 = new Blue();
```

In the table below,

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

| | **Statement** | **Output** |
|---|---|---|
| 1 | var1.method1(); | |
| 2 | var2.method1(); | |
| 3 | var3.method1(); | |
| 4 | var4.method1(); | |
| 5 | var5.method1(); | |
| 6 | var6.method1(); | |
| 7 | var1.method3(); | |
| 8 | var2.method3(); | |
| 9 | var3.method3(); | |
| 10 | ((Blue)var1).method1(); | |
| 11 | ((Crew)var1).method2(); | |
| 12 | ((Sue)var1).method3(); | |
| 13 | ((Blue)var3).method1(); | |
| 14 | ((Crew)var3).method1(); | |
| 15 | ((Sue)var3).method3(); | |
| 16 | ((Moo)var2).method2(); | |
| 17 | ((Crew)var3).method2(); | |
| 18 | ((Moo)var5).method2(); | |
| 19 | ((Moo)var6).method2(); | |
| 20 | ((Moo)var2).method1(); | |

## Task 6

```java
public class Foo {
    String name = "foo";
    public void call1() {
        System.out.println("Foo 1");
    }
    public void call2() {
        call1();
        System.out.println("Foo 2");
    }
}

public class Bar extends Foo {
    public void call2() {
        System.out.println("Bar 2");
    }
    public void call3() {
        System.out.println("Bar 3");
    }
}

public class Buzz extends Bar {
    String name = "Buzz";
    public void call1() {
        System.out.println("Buzz 1");
    }
    public void call4() {
        call3();
        System.out.println("Buzz 4");
    }
}
public class Bux extends Foo {
    String name = "Bux";
    public void call1() {
        System.out.println("Bux 1");
    }
    public void call3() {
        System.out.println("Bux 3");
    }
}
```

**Assuming the following variables have been defined:**

```java
Foo foo1 = new Foo();
Bar bar1 = new Bar();
```

```
Bux bux1 = new Bux();
Foo foo2 = new Buzz();
Bar bar2 = new Buzz();
Object obj1 = new Foo();
```

In the table below,

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

| | **Statement** | **Output** |
|---|---|---|
| 1 | bar1.call1(); | |
| 2 | foo2.call1(); | |
| 3 | foo2.call2(); | |
| 4 | bar2.call3(); | |
| 5 | System.out.println(bar1.name); | |
| 6 | System.out.println(bar2.name); | |
| 7 | System.out.println(((Buzz)bar2).name); | |
| 8 | ((Buzz)bar1).call4(); | |
| 9 | ((Bar)foo1).call3(); | |
| 10 | ((Foo)bux1).call1(); | |
| 11 | ((Bux)foo1).call1(); | |
| 12 | bux1.call1(); | |
| 13 | bux1.call2(); | |
| 14 | ((Foo)foo2).call2(); | |
| 15 | ((Buzz)obj1).call3(); | |
| 16 | ((Buzz)obj1).call2(); | |
| 17 | ((Bux)foo2).call2(); | |
| 18 | ((Buzz)obj1).call1(); | |
| 19 | System.out.println(foo2.name); | |
| 20 | System.out.println(((Bux)foo2).name); | |

## Ungraded Tasks (Optional)

(You don’t have to submit the ungraded tasks)

## Task1

Write the **PlatinumCard** and **SignatureCard** classes derived from **CreditCard** class so that the following code generates the output below.

**Note**: Platinum card users initially have 100 reward points and will get 2 reward points for spending 100 taka each. Signature card users initially have 200 reward points and will get 4 reward points for spending 100 taka each. Signature card users are allowed to bring upto 5 companions at lounges.

**Parent Class**

```java
public class CreditCard {
   public String cardHolder;
   public String accountNo;
   public int rewardPoints;
   public CreditCard(String cardHolder, String accountNo, int rewardPoints){
       this.cardHolder = cardHolder;
       this.accountNo = accountNo;
       this.rewardPoints = rewardPoints;
   }
   public void cardDetails(){
       System.out.println("Card Holder Name: " + cardHolder);
       System.out.println("Account Number: " + accountNo);
       System.out.println("Reward point gained: " + rewardPoints);
   }
}
```

**Driver Code**

```java
public class CardTester {
  public static void main(String[] args) {
    CreditCard card1 = new PlatinumCard("Ali", "345 127");
    CreditCard card2 = new SignatureCard("Rahul", "514 123");
    CreditCard card3 = new SignatureCard("Rohan", "147 965");
    CreditCard [] cards = {card1, card2, card3};
    for (int i = 0; i<cards.length; i++)
    {
      System.out.println("================");
      if (cards[i] instanceof SignatureCard)
      {
        SignatureCard new_card = (SignatureCard) cards[i];
        new_card.spendCash(500);
      }
      else if (cards[i] instanceof PlatinumCard)
      {
        PlatinumCard new_card = (PlatinumCard) cards[i];
        new_card.spendCash(200);
      }
      System.out.println("================");
      cards[i].cardDetails();
    }
  }
}
```

**Output**

```
================
Previous Reward Points: 100
Reward points after spending 200 taka: 104
================
Card Holder Name: Ali
Account Number: 345 127
Reward point gained: 104
================
Previous Reward Points: 200
Reward points after spending 500 taka: 220
================
Card Holder Name: Rahul
Account Number: 514 123
Reward point gained: 220
Possible Number of Companions for Lounge: 5
================
Previous Reward Points: 200
Reward points after spending 500 taka: 220
================
Card Holder Name: Rohan
Account Number: 147 965
Reward point gained: 220
Possible Number of Companions for Lounge: 5
```