RESORT MANAGEMENT SYSTEM

ARSHDEEP SINGH
(2020A7PS0144U)

ACKNOWLEDGEMENT

I would like to express my sincere and heartfelt gratitude to ***Prof. Srinivasan Madapusi***, Director, BITS Pilani, Dubai Campus for giving us the opportunity of thriving to learn, comprehend, and apply the numerous different concepts and ideas of engineering in the real world and being able to do such project reports under a soothing and motivating environment.

I am very thankful to my Object Oriented Programming Lab Faculty, ***Prof. Sujala Shetty*** for providing me with the opportunity and guidance to complete the assigned project report.

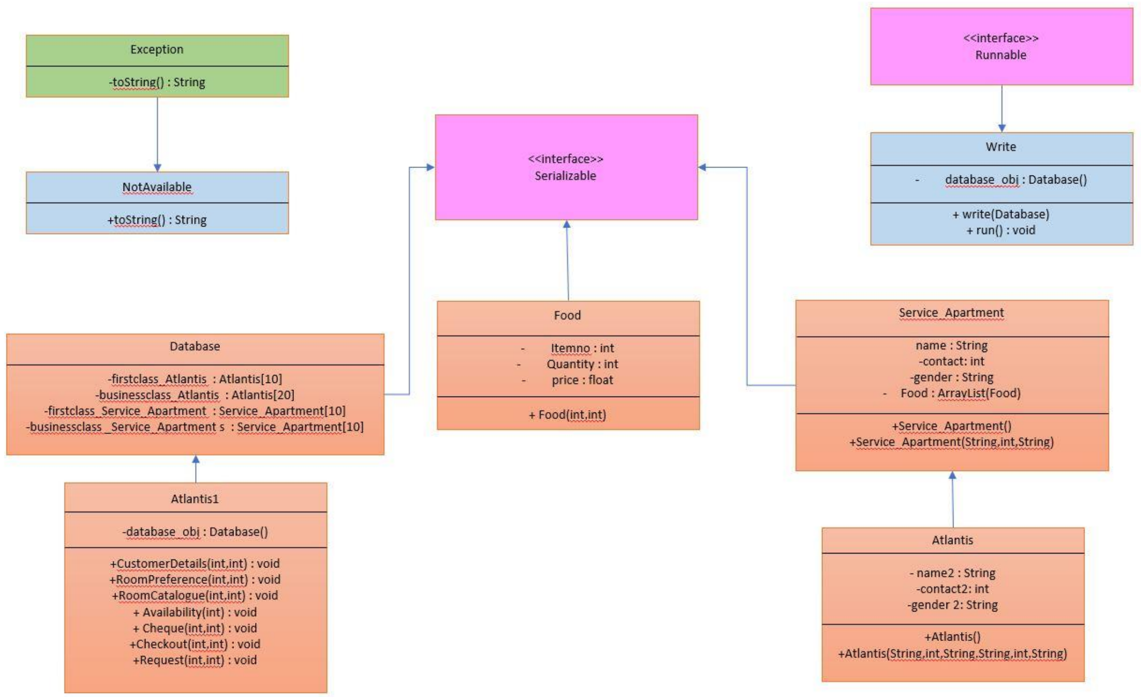
INTRODUCTION:

I have built a Resort Management System using the concepts ArrayList, Classes and Objects, implementing Interface, Inheritance, File Handling with Objects, User Defined Exception and Exception Handling.
The Atlantis provides a very efficient service to the guests and it's system is well planned and organized for the authorities of the hotel to run it smoothly. The system can be used to store customer details, booking rooms of four different varieties as per customers choice, the number of guests and ordering food from menu of the Palace for particular rooms.
The Atlantis offers Exotic Suites and Service Apartments. There are two categories of rooms in the Palace, first class and Business class rooms. The Service apartments consist of a single bed whereas the Exotic Suites include two beds and some extra luxuries. Depending on the guest's choice, they can choose whether they want an AC or a non-AC room, a smoking or a non-smoking room.
The food menu of NOBU DUBAI contains delicious items that the guests can order such as Tiger Prawns, Salmon with Caviar, Wagyu Beef Sliders. Complementary activities and services have been provided by the The Atlantis under all categories of rooms. If the customers change their mind, they can unBook Roomthe room as well.

The system views different types of features of the rooms and checks for the room availability as well. During the time of checkout, a bill will be generated which incorporates the room charges and the food invoice, and produces the final total to be paid. This is a menu-driven program and it runs until the user exits. File Handling has been used to store the current status of the Palace, which includes the customer details, booked rooms, and the food ordered from the menu into a file, as once the program terminates, and when restarted later, the previous details are not lost. The program reads the file when it restarts to know the previous status of the Palace. If the user tries to Book Rooman already allotted room, a User defined exception is thrown. The use of Exception handling has been made to properly deal with any kind of unexpected issues, making the program definitive.

OOPS ASSIGNMENT
ARSHDEEP SINGH
2020A7PS0144U

UML DIAGRAM:


Exception
-toString() : String
NotAvailable
+toString() : String
<<interface>> Serializable
<<interface>> Runnable
Write
- database_obj : Database()
+ write(Database)
+ run() : void
Database
-firstclass_Atlantis : Atlantis[10]
-businessclass_Atlantis : Atlantis[20]
-firstclass_Service_Apartment : Service_Apartment[10]
-businessclass _Service_Apartment s : Service_Apartment[10]
Food
- Itemno : int
- Quantity : int
- price : float
+ Food(int,int)
Service_Apartment
name : String
-contact: int
-gender : String
- Food : ArrayList(Food)
+Service_Apartment()
+Service_Apartment(String,int,String)
Atlantis1
-database_obj : Database()
+CustomerDetails(int,int) : void
+RoomPreference(int,int) : void
+RoomCatalogue(int,int) : void
+ Availability(int) : void
+ Cheque(int,int) : void
+Checkout(int,int) : void
+Request(int,int) : void
Atlantis
- name2 : String
-contact2: int
-gender 2: String
+Atlantis()
+Atlantis(String,int,String,String,int,String)

PACKAGES USED:

1	**import java.io. ***	**File** **FileInputStream** **FileOutputStream** **ObjectInputStream** **ObjectOutputStream** **Serializable**
2	**import java.util. ***	**ArrayList** **Scanner**

CONCEPTS USED:

ArrayList	ArrayList supports dynamic arrays that can grow as needed. In Java, standard arrays are of a fixed length. After arrays are created, they cannot grow or shrink, which means that you must know in advance how many elements an array will hold. But, sometimes, you may not know until run time precisely how large of an array you need. To handle this situation, the collections framework defines ArrayList. In essence, an ArrayList is a variable-length array of object references. That is, an ArrayList can dynamically increase or decrease in size. Array lists are created with an initial size. When this size is exceeded, the collection is automatically enlarged. When objects are removed, the array may be shrunk.
Classes and Objects	A class is a user defined blueprint or prototype from which objects are created. Objects are the Basic unit of Object Oriented Programming. They represent the real life entities and are instances of a class
Interfaces	Interfaces are syntactically similar to classes, but they lack instance variables, and their methods are declared without any body. In practice, this means that you can define interfaces which don’t make assumptions about how they are implemented. Once it is defined, any number of classes can implement an interface
Inheritance	A class inherits from another if all objects of its class are special cases of objects of the other class, capable of exhibiting the same behavior but possibly with additional responsibilities and a richer state.
User Defined Exception	Exceptions can be generated by the Java run-time system, or they can be manually generated by your code. Exceptions thrown by Java relate to fundamental errors that violate the rules of the Java language or the constraints of the Java execution environment. Manually generated exceptions are typically used to report some error condition to the caller of a method
Exception Handling	escribes an exceptional (that is, error) condition that has occurred in a piece of code. When an exceptional condition arises, an object representing that exception is created and thrown in the method that caused the error. That method may choose to handle the exception itself, or pass it on

CODE:

```
import java.io.*;
import java.util.*;


class Food implements Serializable
{
   int itemno;
   int quantity;
   float price;

   Food(int itemno,int quantity)
   {
      this.itemno=itemno;
      this.quantity=quantity;
      switch(itemno)
      {
         case 1:price=quantity*220;
            break;
         case 2:price=quantity*259;
            break;
         case 3:price=quantity*125;
            break;
         case 4:price=quantity*150;
            break;
      }
   }

}


class Service_Apartment implements Serializable
{
         String name;
   double contact;
   String gender;
   ArrayList<Food> food =new ArrayList<>();

   Service_Apartment()
   {
      this.name="";
   }
   Service_Apartment(String name,double contact,String gender)
   {
      this.name=name;
      this.contact=contact;
      this.gender=gender;
   }
}

class Atlantis extends Service_Apartment implements Serializable
{
   String name2;
   double contact2;
   String gender2;

   Atlantis()
   {
      this.name="";
      this.name2="";
   }
   Atlantis(String name,double contact,String gender,String name2,double contact2,String
gender2)
   {
      this.name=name;
      this.contact=contact;
      this.gender=gender;
      this.name2=name2;
      this.contact2=contact2;
      this.gender2=gender2;
   }
}

class NotAvailable extends Exception
{
  // @Override
   public String toString()
   {
      return "Not Available !";
   }
}

class Database implements Serializable
{
         Atlantis firstclass_Atlantis[]=new Atlantis[10];
         Atlantis businessclass_Atlantis[]=new Atlantis[20];
         Service_Apartment firstclass_Service_Apartment[]=new Service_Apartment[10];
         Service_Apartment businessclass_Service_Apartment[]=new Service_Apartment[20];
}

class Atlantis1
{
   static Database database_obj=new Database();
   static Scanner sc = new Scanner(System.in);
   static void CustomerDetails(int i,int rn)
   {
      String name, gender;
      double contact;
      String name2 = null;
      double contact2 = 0;
      String gender2="";
      System.out.print("\nEnter customer name: ");
      name = sc.next();
      System.out.print("Enter contact number: ");
      contact=sc.nextDouble();
      System.out.print("Enter gender: ");
      gender = sc.next();
      if(i<3)
      {
      System.out.print("Enter second customer name: ");
      name2 = sc.next();
      System.out.print("Enter contact number: ");
      contact2=sc.nextDouble();
      System.out.print("Enter gender: ");
      gender2 = sc.next();
      }

        switch (i) {
         case 1:database_obj.firstclass_Atlantis[rn]=new
Atlantis(name,contact,gender,name2,contact2,gender2);
                break;
         case 2:database_obj.businessclass_Atlantis[rn]=new
Atlantis(name,contact,gender,name2,contact2,gender2);
                break;
         case 3:database_obj.firstclass_Service_Apartment[rn]=new
Service_Apartment(name,contact,gender);
                break;
         case 4:database_obj.businessclass_Service_Apartment[rn]=new
Service_Apartment(name,contact,gender);
                break;
         default:System.out.println("Wrong option");
                break;
      }
   }


static void RoomPreferance(int i)
   {
      int j;
      int rn;
      System.out.println("\nChoose room number from : ");
      switch (i) {
         case 1:
                for(j=0;j<database_obj.firstclass_Atlantis.length;j++)
                {
                    if(database_obj.firstclass_Atlantis[j]==null)
                    {
                        System.out.print(j+1+",");
                    }
                }
                System.out.print("\nEnter room number: ");
                try{
                rn=sc.nextInt();
                rn--;
                if(database_obj.firstclass_Atlantis[rn]!=null)
                    throw new NotAvailable();
                CustomerDetails(i,rn);
                }
                catch(Exception e)
                {
                    System.out.println("Invalid Option");
                    return;
                }
                break;
         case 2:
              for(j=0;j<database_obj.businessclass_Atlantis.length;j++)
                {
                    if(database_obj.businessclass_Atlantis[j]==null)
                    {
                        System.out.print(j+11+",");
                    }
                }
                System.out.print("\nEnter room number: ");
                try{
                rn=sc.nextInt();
                rn=rn-11;
                if(database_obj.businessclass_Atlantis[rn]!=null)
                    throw new NotAvailable();
                CustomerDetails(i,rn);
                }
                catch(Exception e)
                {
                    System.out.println("Invalid Option");
                    return;
                }
                break;
         case 3:
               for(j=0;j<database_obj.firstclass_Service_Apartment.length;j++)
                {
                    if(database_obj.firstclass_Service_Apartment[j]==null)
                    {
                        System.out.print(j+31+",");
                    }
                }
                System.out.print("\nEnter room number: ");
                try{
```

```
            rn=sc.nextInt();
            rn=rn-31;
            if(database_obj.firstclass_Service_Apartment[rn]!=null)
               throw new NotAvailable();
            CustomerDetails(i,rn);
            }
            catch(Exception e)
            {
               System.out.println("Invalid Option");
               return;
            }
            break;
         case 4:
              for(j=0;j<database_obj.businessclass_Service_Apartment.length;j++)
            {
               if(database_obj.businessclass_Service_Apartment[j]==null)
               {
                  System.out.print(j+41+",");
               }
            }
            System.out.print("\nEnter room number: ");
            try{
            rn=sc.nextInt();
            rn=rn-41;
            if(database_obj.businessclass_Service_Apartment[rn]!=null)
               throw new NotAvailable();
            CustomerDetails(i,rn);
            }
            catch(Exception e)
            {
              System.out.println("Invalid Option");
               return;
            }
            break;
         default:
            System.out.println("Enter valid option");
            break;
      }
      System.out.println("Room Booked");
   }



static void RoomCatalogue(int i)
{
   switch (i) {
      case 1:System.out.println("Number of King size beds : 1\nSmoking room\nAC : Yes\nFree
breakfast, lunch & dinner : Yes\nCharge per day:4000 aed, Free access to spa ");
         break;
```

```
        case 2:System.out.println("Number of King size beds : 1\nSmoking room\nAC : No\nFree breakfast : Yes\nCharge per day:3000 aed, Drinks on the house ");
            break;
        case 3:System.out.println("Number of Queen size beds : 1\nNon-Smoking room\nAC : Yes\nFree breakfast, lunch & dinner : Yes\nCharge per day:2200 aed, Only Soft Drinks on the house ");
            break;
        case 4:System.out.println("Number of Queen size beds : 1\nNon-Smoking room\nAC : No\nFree breakfast : Yes\nCharge per day:1200 aed, complementary JetSkiing and Parasailing ");
            break;
        default:
            System.out.println("Incorrect Option! Please enter a valid option");
            break;
    }
 }

static void availability(int i)
{
  int j,count=0;
    switch (i) {
        case 1:
            for(j=0;j<10;j++)
            {
                if(database_obj.firstclass_Atlantis[j]==null)
                    count++;
            }
            break;
        case 2:
            for(j=0;j<database_obj.businessclass_Atlantis.length;j++)
            {
                if(database_obj.businessclass_Atlantis[j]==null)
                    count++;
            }
            break;
        case 3:
            for(j=0;j<database_obj.firstclass_Service_Apartment.length;j++)
            {
                if(database_obj.firstclass_Service_Apartment[j]==null)
                    count++;
            }
            break;
        case 4:
            for(j=0;j<database_obj.businessclass_Service_Apartment.length;j++)
            {
                if(database_obj.businessclass_Service_Apartment[j]==null)
                    count++;
            }
            break;
        default:
          System.out.println("Enter valid option");
          break;
    }
    System.out.println("Number of rooms available : "+count);
}

    static void Cheque(int rn,int rtype)
    {
       double amount=0;
       String list[]={"Tiger Prawns               ","Salmon with Caviar Biryani            ","Rosemary
Lamb             ","Wagyu Beef Sliders                        "};
       System.out.println("\n*******");
       System.out.println(" Cheque:-");
       System.out.println("*******");

       switch(rtype)
       {
          case 1:
             amount+=4000;
                System.out.println("\nPalace Charges - "+4000);
                System.out.println("\n==================================================
=====================================================");
                System.out.println("Eatery Charges:- ");
                System.out.println("==================================================
====================================================");
                 System.out.println("Commodity             Quantity         Cost");
                System.out.println("--------------------------------------------------------------------------------
-----");
                for(Food obb:database_obj.firstclass_Atlantis[rn].food)
                {
                   amount+=obb.price;
                   String format = "%-10s%-10s%-10s%n";
                   System.out.printf(format,list[obb.itemno-1],obb.quantity,obb.price );
                }

             break;
          case 2:amount+=3000;
                System.out.println("Palace Charge - "+3000);
                System.out.println("\nEatery Charges:- ");
                System.out.println("==================================================
==============================================");
                 System.out.println("Commodity             Quantity         Cost");
                System.out.println("--------------------------------------------------------------------------------
-----");
                for(Food obb:database_obj.businessclass_Atlantis[rn].food)
                {
                   amount+=obb.price;
                   String format = "%-10s%-10s%-10s%n";
                   System.out.printf(format,list[obb.itemno-1],obb.quantity,obb.price );
                }
            break;
        case 3:amount+=2200;
                System.out.println("Palace Charge - "+2200);
                System.out.println("\nEatery Charges:- ");
                System.out.println("==================================================
==============================================");
                System.out.println("Commodity           Quantity        Cost");
                System.out.println("------------------------------------------------------------------------------
---");
                for(Food obb:database_obj.firstclass_Service_Apartment[rn].food)
                {
                    amount+=obb.price;
                    String format = "%-10s%-10s%-10s%n";
                    System.out.printf(format,list[obb.itemno-1],obb.quantity,obb.price );
                }
            break;
        case 4:amount+=1200;
                System.out.println("Palace Charge - "+1200);
                System.out.println("\nEatery Charges:- ");
                System.out.println("==================================================
========================================");
                System.out.println("Commodity           Quantity        Cost");
                System.out.println("------------------------------------------------------------------------------
---");
                for(Food obb:database_obj.businessclass_Service_Apartment[rn].food)
                {
                    amount+=obb.price;
                    String format = "%-10s%-10s%-10s%n";
                    System.out.printf(format,list[obb.itemno-1],obb.quantity,obb.price );
                }
            break;
        default:
            System.out.println("Not valid");
    }
    System.out.println("\nTotal Amount- "+amount);
  }

static void Checkout(int rn,int rtype)
  {
    int j;
    char w;
    switch (rtype) {
        case 1:
            if(database_obj.firstclass_Atlantis[rn]!=null)
                System.out.println("Room used by
"+database_obj.firstclass_Atlantis[rn].name);
            else
            {
                System.out.println("Empty Already");
```

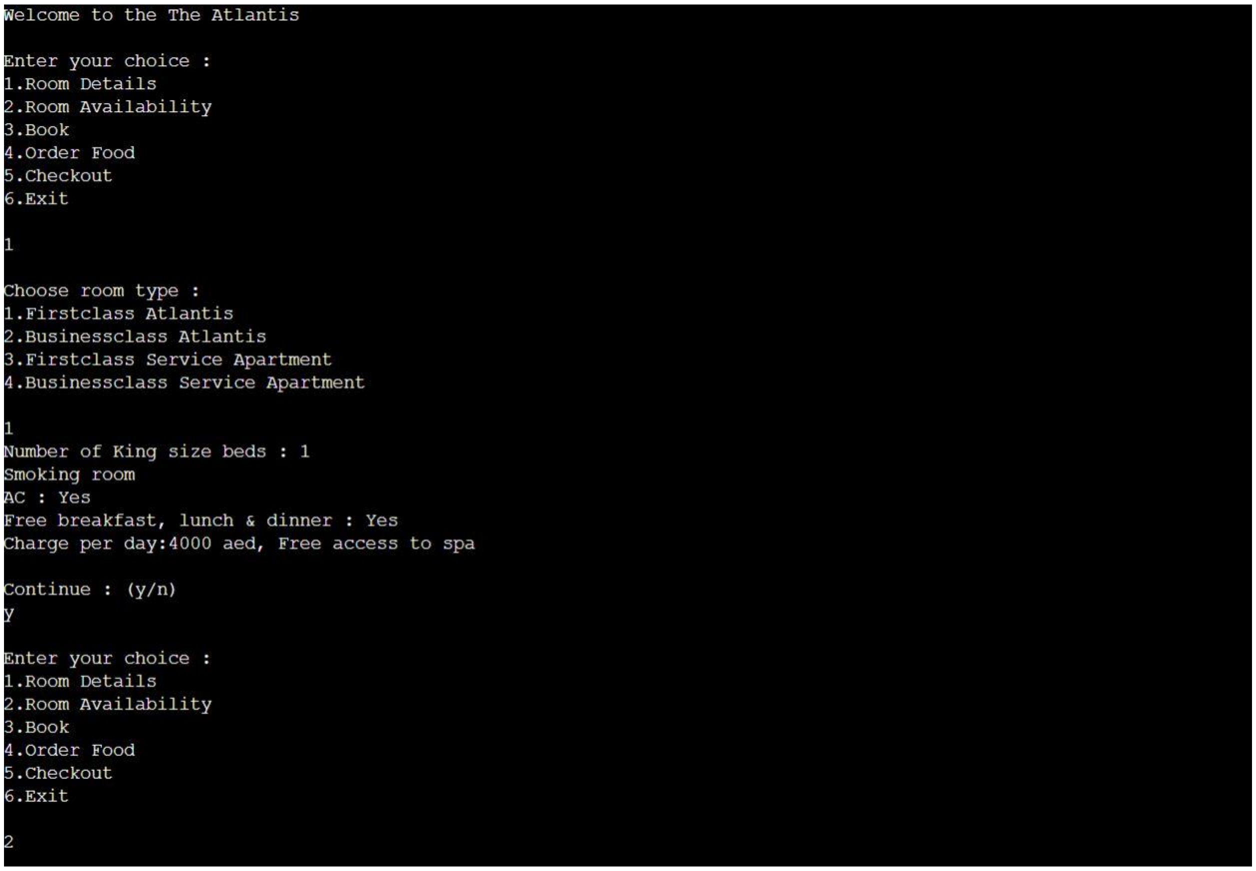
```
                        return;
                }
                System.out.println("Do you want to checkout ?(y/n)");
                 w=sc.next().charAt(0);
                if(w=='y'||w=='Y')
                {
                    Cheque(rn,rtype);
                    database_obj.firstclass_Atlantis[rn]=null;
                    System.out.println("Checkout succesfully done");
                }

                break;
            case 2:
                if(database_obj.businessclass_Atlantis[rn]!=null)
                    System.out.println("Room used by
"+database_obj.businessclass_Atlantis[rn].name);
                else
                {
                    System.out.println("Empty Already");
                        return;
                }
                System.out.println(" Do you want to checkout ?(y/n)");
                 w=sc.next().charAt(0);
                if(w=='y'||w=='Y')
                {
                    Cheque(rn,rtype);
                    database_obj.businessclass_Atlantis[rn]=null;
                    System.out.println("Checkout succesfully done");
                }

                break;
            case 3:
                if(database_obj.firstclass_Service_Apartment[rn]!=null)
                    System.out.println("Room used by
"+database_obj.firstclass_Service_Apartment[rn].name);
                else
                 {
                    System.out.println("Empty Already");
                        return;
                }
                System.out.println(" Do you want to checkout ? (y/n)");
                w=sc.next().charAt(0);
                if(w=='y'||w=='Y')
                {
                    Cheque(rn,rtype);
                    database_obj.firstclass_Service_Apartment[rn]=null;
                    System.out.println("Checkout succesfully done");
                }

                break;
```

```
        case 4:
            if(database_obj.businessclass_Service_Apartment[rn]!=null)
                System.out.println("Room used by 
"+database_obj.businessclass_Service_Apartment[rn].name);
            else
             {
                System.out.println("Empty Already");
                    return;
            }
            System.out.println(" Do you want to checkout ? (y/n)");
             w=sc.next().charAt(0);
            if(w=='y'||w=='Y')
            {
                Cheque(rn,rtype);
                database_obj.businessclass_Service_Apartment[rn]=null;
                System.out.println("Checkout succesfully done");
            }
            break;
        default:
            System.out.println("\nEnter valid option : ");
            break;
        }
    }

    static void Request(int rn,int rtype)
    {
        int i,q;
        char wish;
         try{
            System.out.println("\n==================================================\
n   WELCOME TO NOBU 
DUBAI  \n==================================================\n\n 1.Tiger 
Prawns                  AED 220\n 2.Salmon with Caviar            AED 259\n 3.Rosemary 
Lamb                AED 125\n 4.Wagyu Beef Sliders            AED 150\n");
        do
        {
            i = sc.nextInt();
            System.out.print("Quantity- ");
            q=sc.nextInt();

              switch(rtype){
            case 1: database_obj.firstclass_Atlantis[rn].food.add(new Food(i,q));
                break;
            case 2: database_obj.businessclass_Atlantis[rn].food.add(new Food(i,q));
                break;
            case 3: database_obj.firstclass_Service_Apartment[rn].food.add(new Food(i,q));
                break;
            case 4: database_obj.businessclass_Service_Apartment[rn].food.add(new Food(i,q));
                break;
        }
              System.out.println("Do you want to order anything else ? (y/n)");
              wish=sc.next().charAt(0);
        }while(wish=='y'||wish=='Y');
        }
         catch(NullPointerException e)
            {
               System.out.println("\nRoom not booked");
            }
         catch(Exception e)
         {
            System.out.println("Cannot be done");
         }
    }
}

class write implements Runnable
{
    Database database_obj;
    write(Database database_obj)
    {
        this.database_obj=database_obj;
    }
    @Override
    public void run() {
          try{
        FileOutputStream fout=new FileOutputStream("backup");
        ObjectOutputStream oos=new ObjectOutputStream(fout);
        oos.writeObject(database_obj);
        }
        catch(Exception e)
        {
            System.out.println("Error in writing "+e);
        }

    }

}

class Main
{

        public static void main(String[] args){

        System.out.println("Welcome to the The Atlantis");

        try
        {
        File f = new File("backup");
        if(f.exists())
        {
            FileInputStream fin=new FileInputStream(f);
            ObjectInputStream ois=new ObjectInputStream(fin);
            Atlantis1.database_obj=(Database)ois.readObject();
        }
        Scanner sc = new Scanner(System.in);
        int ch,ch2;
        char wish;
        x:
        do{

        System.out.println("\nEnter your choice :\n1.Room Details\n2.Room Availability
\n3.Book\n4.Order Food\n5.Checkout\n6.Exit\n");
        ch = sc.nextInt();
        switch(ch){
            case 1: System.out.println("\nChoose room type :\n1.Firstclass Atlantis
\n2.Businessclass Atlantis \n3.Firstclass Service Apartment \n4.Businessclass Service
Apartment\n");
                    ch2 = sc.nextInt();
                    Atlantis1.RoomCatalogue(ch2);
                break;
            case 2:System.out.println("\nChoose room type :\n1.Firstclass Atlantis \n2.Businessclass
Atlantis \n3.Firstclass Service Apartment \n4.Businessclass Service Apartment\n");
                     ch2 = sc.nextInt();
                     Atlantis1.availability(ch2);
                break;
            case 3:System.out.println("\nChoose room type :\n1.Firstclass Atlantis \n2.Businessclass
Atlantis \n3.Firstclass Service Apartment \n4.Businessclass Service Apartment\n");
                     ch2 = sc.nextInt();
                     Atlantis1.RoomPreferance(ch2);
                break;
            case 4:
                 System.out.print("Room Number -");
                     ch2 = sc.nextInt();
                     if(ch2>60)
                         System.out.println("Room doesn't exist");
                     else if(ch2>40)
                         Atlantis1.Request(ch2-41,4);
                     else if(ch2>30)
                         Atlantis1.Request(ch2-31,3);
                     else if(ch2>10)
                         Atlantis1.Request(ch2-11,2);
                     else if(ch2>0)
                         Atlantis1.Request(ch2-1,1);
                     else
                         System.out.println("Room doesn't exist");
                     break;
            case 5:
```

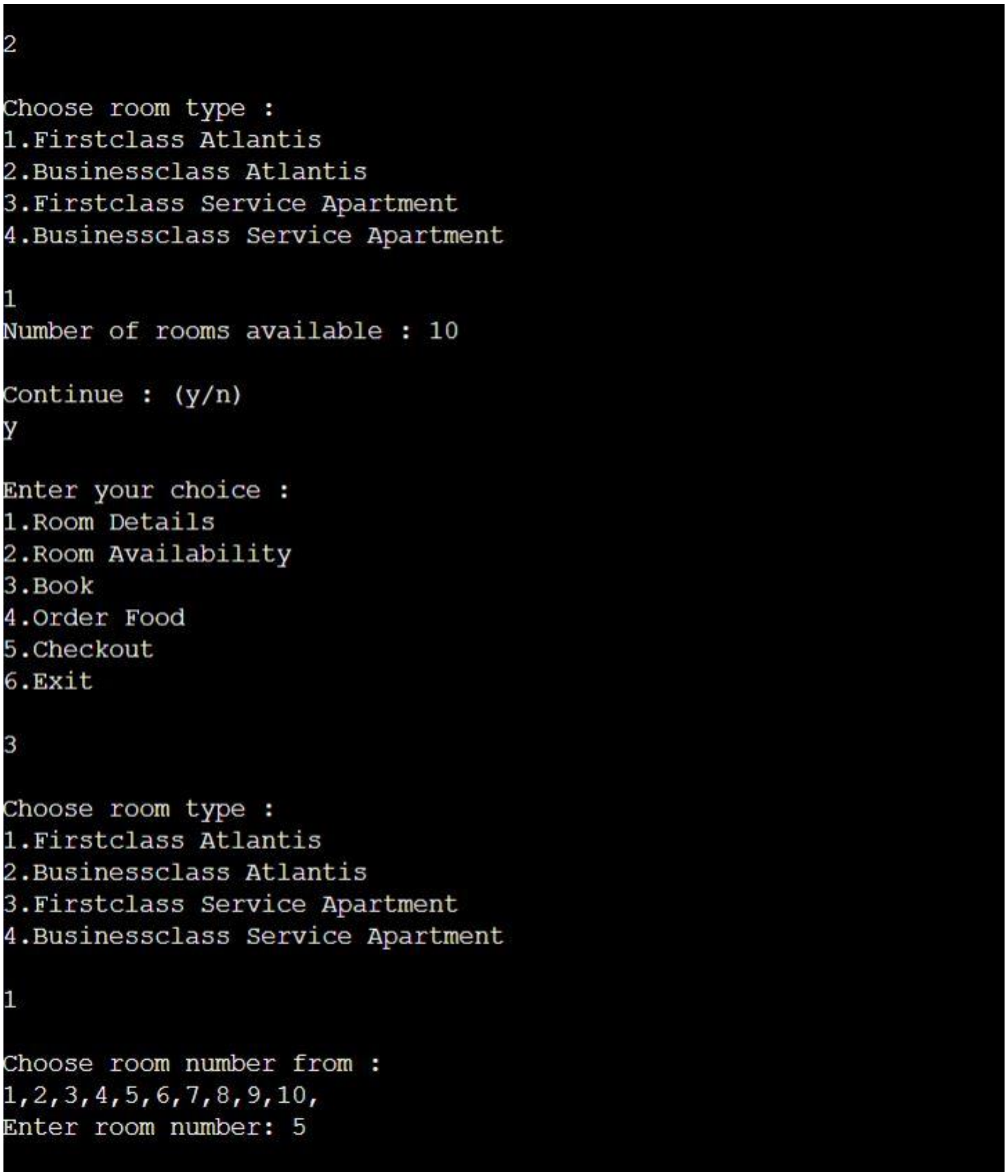
```
            System.out.print("Room Number -");
                ch2 = sc.nextInt();
                if(ch2>60)
                   System.out.println("Room doesn't exist");
                else if(ch2>40)
                   Atlantis1.Checkout(ch2-41,4);
                else if(ch2>30)
                  Atlantis1.Checkout(ch2-31,3);
                else if(ch2>10)
                   Atlantis1.Checkout(ch2-11,2);
                else if(ch2>0)
                  Atlantis1.Checkout(ch2-1,1);
                else
                   System.out.println("Room doesn't exist");
                break;
         case 6:break x;

      }

         System.out.println("\nContinue : (y/n)");
         wish=sc.next().charAt(0);
         if(!(wish=='y'||wish=='Y'||wish=='n'||wish=='N'))
         {
            System.out.println("Invalid Option");
            System.out.println("\nContinue : (y/n)");
            wish=sc.next().charAt(0);
         }

      }while(wish=='y'||wish=='Y');

      Thread t=new Thread(new write(Atlantis1.database_obj));
      t.start();
      }
         catch(Exception e)
         {
            System.out.println("Not a valid input");
         }
   }
}
```

******************************END****************************

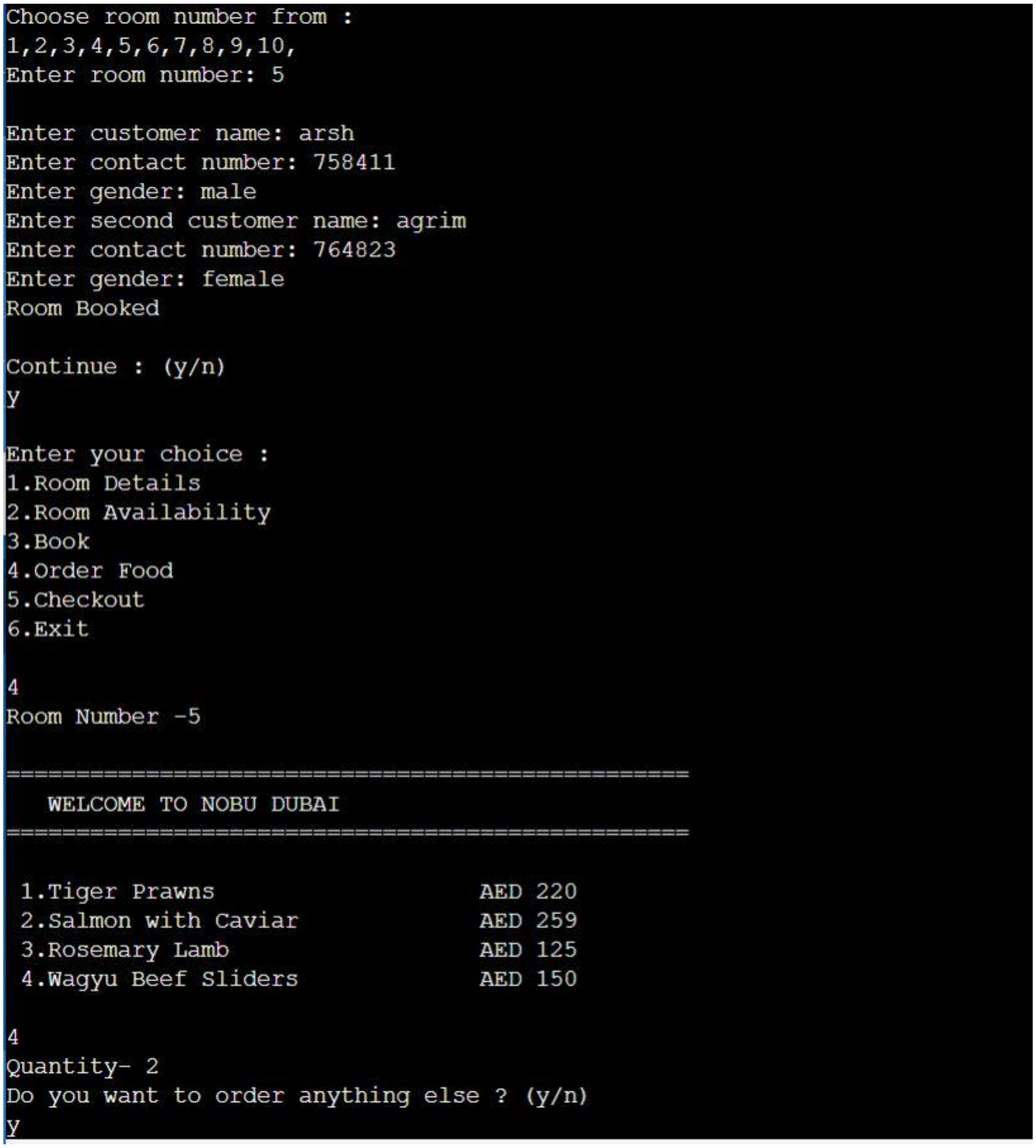
SAMPLE OUTPUT:



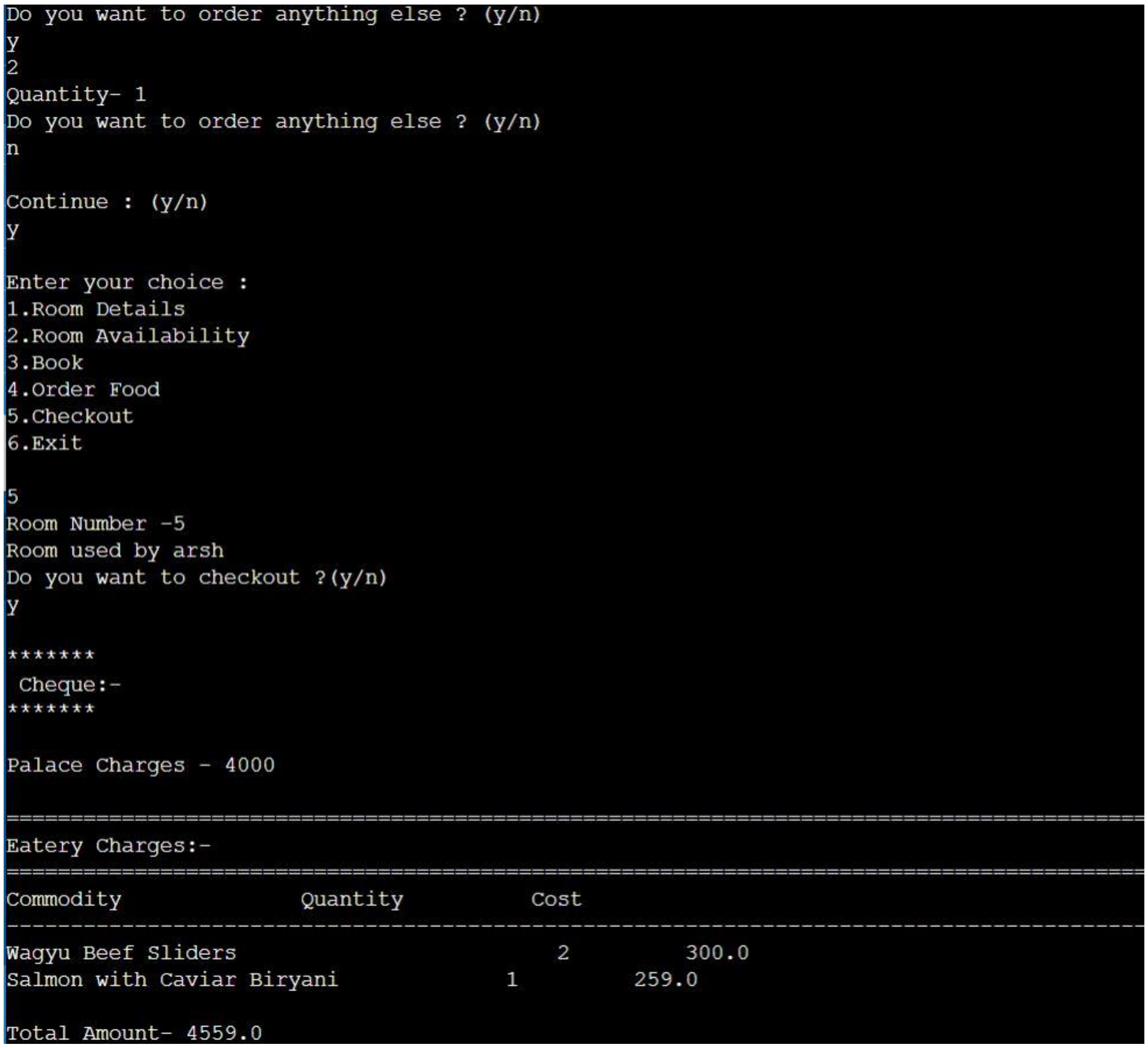
```
Welcome to the The Atlantis

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

1

Choose room type :
1.Firstclass Atlantis
2.Businessclass Atlantis
3.Firstclass Service Apartment
4.Businessclass Service Apartment

1
Number of King size beds : 1
Smoking room
AC : Yes
Free breakfast, lunch & dinner : Yes
Charge per day:4000 aed, Free access to spa

Continue : (y/n)
y

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

2
```



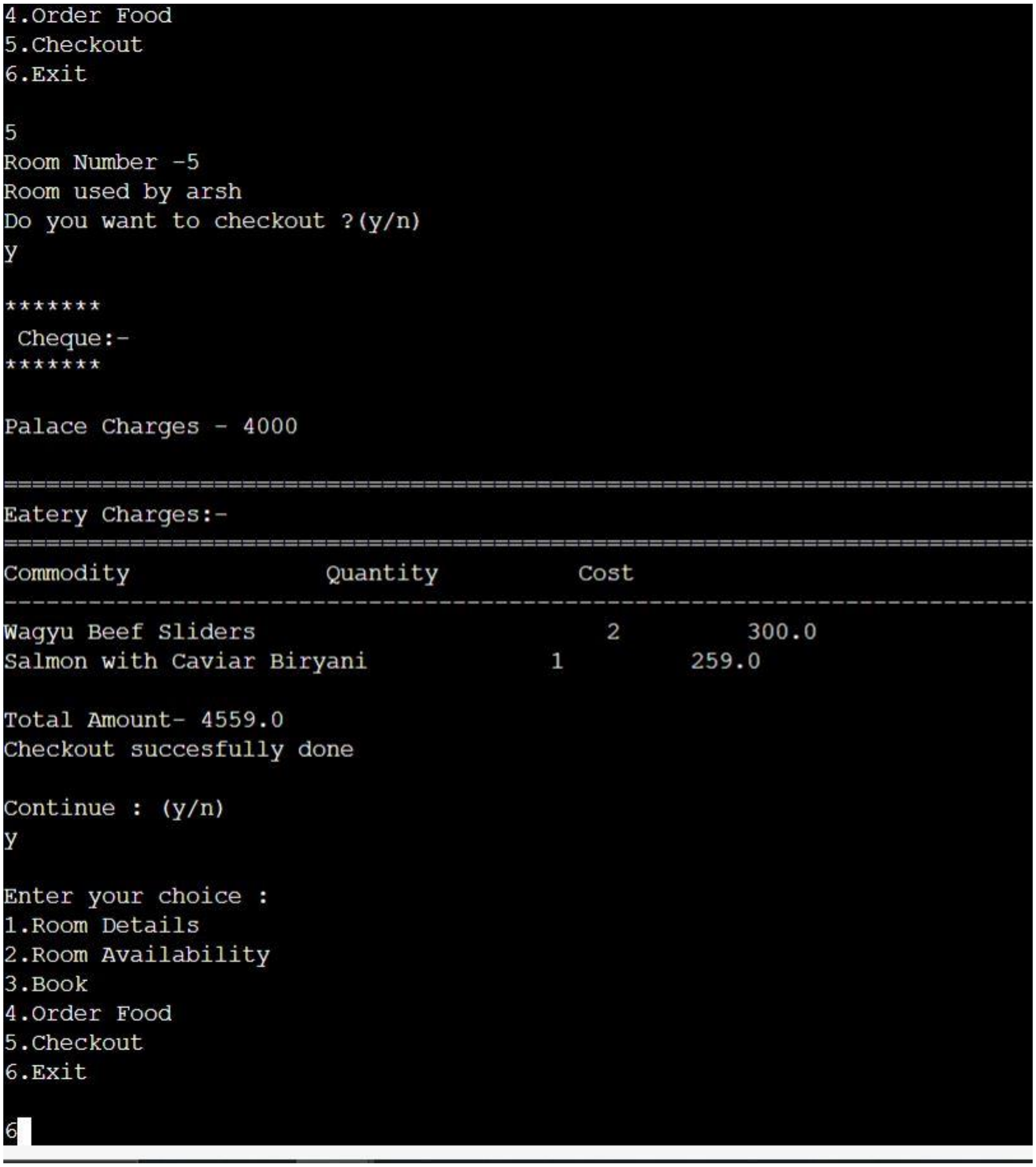
```
2

Choose room type :
1.Firstclass Atlantis
2.Businessclass Atlantis
3.Firstclass Service Apartment
4.Businessclass Service Apartment

1
Number of rooms available : 10

Continue : (y/n)
y

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

3

Choose room type :
1.Firstclass Atlantis
2.Businessclass Atlantis
3.Firstclass Service Apartment
4.Businessclass Service Apartment

1

Choose room number from :
1,2,3,4,5,6,7,8,9,10,
Enter room number: 5
```



```
Choose room number from :
1,2,3,4,5,6,7,8,9,10,
Enter room number: 5

Enter customer name: arsh
Enter contact number: 758411
Enter gender: male
Enter second customer name: agrim
Enter contact number: 764823
Enter gender: female
Room Booked

Continue : (y/n)
y

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

4
Room Number -5

==================================================
   WELCOME TO NOBU DUBAI
==================================================

 1.Tiger Prawns                   AED 220
 2.Salmon with Caviar             AED 259
 3.Rosemary Lamb                  AED 125
 4.Wagyu Beef Sliders             AED 150

4
Quantity- 2
Do you want to order anything else ? (y/n)
y
```



```
Do you want to order anything else ? (y/n)
y
2
Quantity- 1
Do you want to order anything else ? (y/n)
n

Continue : (y/n)
y

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

5
Room Number -5
Room used by arsh
Do you want to checkout ?(y/n)
y

*******
 Cheque:-
*******

Palace Charges - 4000

==========================================================================================
Eatery Charges:-
==========================================================================================
Commodity              Quantity          Cost
------------------------------------------------------------------------------------------
Wagyu Beef Sliders                         2          300.0
Salmon with Caviar Biryani             1         259.0

Total Amount- 4559.0
```



```
4.Order Food
5.Checkout
6.Exit

5
Room Number -5
Room used by arsh
Do you want to checkout ?(y/n)
y

*******
 Cheque:-
*******

Palace Charges - 4000

==========================================================================
Eatery Charges:-
==========================================================================
Commodity               Quantity          Cost
--------------------------------------------------------------------------
Wagyu Beef Sliders                          2         300.0
Salmon with Caviar Biryani              1         259.0

Total Amount- 4559.0
Checkout succesfully done

Continue : (y/n)
y

Enter your choice :
1.Room Details
2.Room Availability
3.Book
4.Order Food
5.Checkout
6.Exit

6
```

BIBLIOGRAPHY

1. www.onlinegdb.com
2. www.geeksforgeeks.com
3. www.youtube.com
4. Cay Horstmann, Object Oriented Design &amp; Patterns, John Wiley &amp; Sons, 2006, 2 nd Edition
5. Herbert Schildt, The complete Reference Java 2, 5th Edition, Tata McGraw Hill.
6. James W. Cooper, Java TM Design Patterns – A Tutorial, Addison-Wesly, 2000.