****

|  |
| --- |
| Course ProjectSecond Semester-2021Session 1441/1442 AH |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Student #1 ID: |  | | Student #1 Name: | | | |  | | | | | |
|  | Student #2 ID: |  | | Student #2 Name: | | | |  | | | | | |
|  | Student #3 ID: |  | | Student #3 Name: | | | |  | | | | | |
|  | Student #4 ID: |  | | Student #4 Name: | | | |  | | | | | |
|  | Student #5 ID: |  | | Student #5 Name: | | | |  | | | | | |
|  | Student #6 ID: |  | | Student #6 Name: | | | |  | | | | | |
|  | Group No. | | |  | | | | | | | | | |
|  |  | | | | | | | | | | | | |
|  | **Department:** | | | | Department of Computer Science | | | | | | | | |
|  | **Course Code:** | | | | COCS 203 | | | | | | | | |
|  | **Course Name:** | | | | Programming II | | | | | | | | |
|  | **Deadline:** | | | | Wednesday, April 14, 2021 | | | | | | | | |
| **Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Examiner:** | | |
| **Students🡪** | | | | **S1** | | **S2** | **S3** | **S4** | **S5** | **S6** |
| **Outcome-2**  **(10 Marks)** | | | **2.1** | /4 | | /4 | /4 | /4 | /4 | /4 |
| **2.2** | /4 | | /4 | /4 | /4 | /4 | /4 |
| **2.3** | /2 | | /2 | /2 | /2 | /2 | /2 |
| **Outcome-5**  **(10 Marks)** | | | **5.1** | /2 | | /2 | /2 | /2 | /2 | /2 |
| **5.2** | /4 | | /4 | /4 | /4 | /4 | /4 |
| **5.3** | /4 | | /4 | /4 | /4 | /4 | /4 |
| **Total Marks**  **(In Figure)** | | | | /20 | | /20 | /20 | /20 | /20 | /20 |

**Project Specifications**

**Description**

Resturant is a system, which can be used by any company to reserve orders for his customers. It offers some plates to be ordered and served for the customers

**The Initial Procedure of the Program**

The program will display information to help user select a function.

**The Menu Procedure of the Program (output)**

\*\*\*\*\* Welcome to Resturant \*\*\*\*\*

1. Add An Customer

2. Add a Reservation for a customer

3. Pay for a Reservation

4. Print Customers

5. Add an Employee

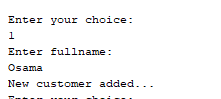
6. print Employees

7. For Exit

8. For Help

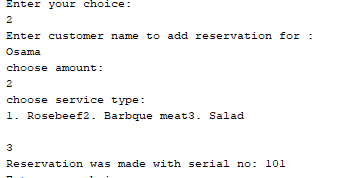
1. **Add a Customer**

This functions will add a new customer to the list customers in restaurant it can be accomplished by calling the function



1. **Add a Reservation for a Customer**

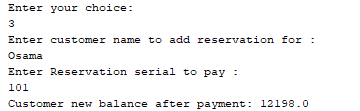
This will call the function addCustomer() that will add a new object of class customer that is generated from the random values.



1. **Pay for a Reservation**

To make the payment first the user is asked to enter the name for the customer to search for if not found it will exit , if found then we search the customer reservations stored in the customer object which has the reservations array list if not found a message with not found will appear.

After we got the reservation correctly then we call method pay() which will substract the items calculated value from the customer main balance and output it on the console as follows:



1. **Print Customers**

To print employees if they’re exist



1. **Add an Employee**

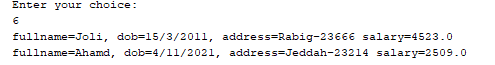
This will call the function addWroker() that will add a new object of class worker that is generated from random

Functions.



1. **Print Employees**

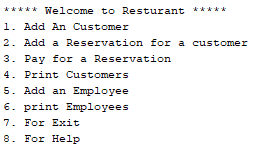
To print employees if they’re exist



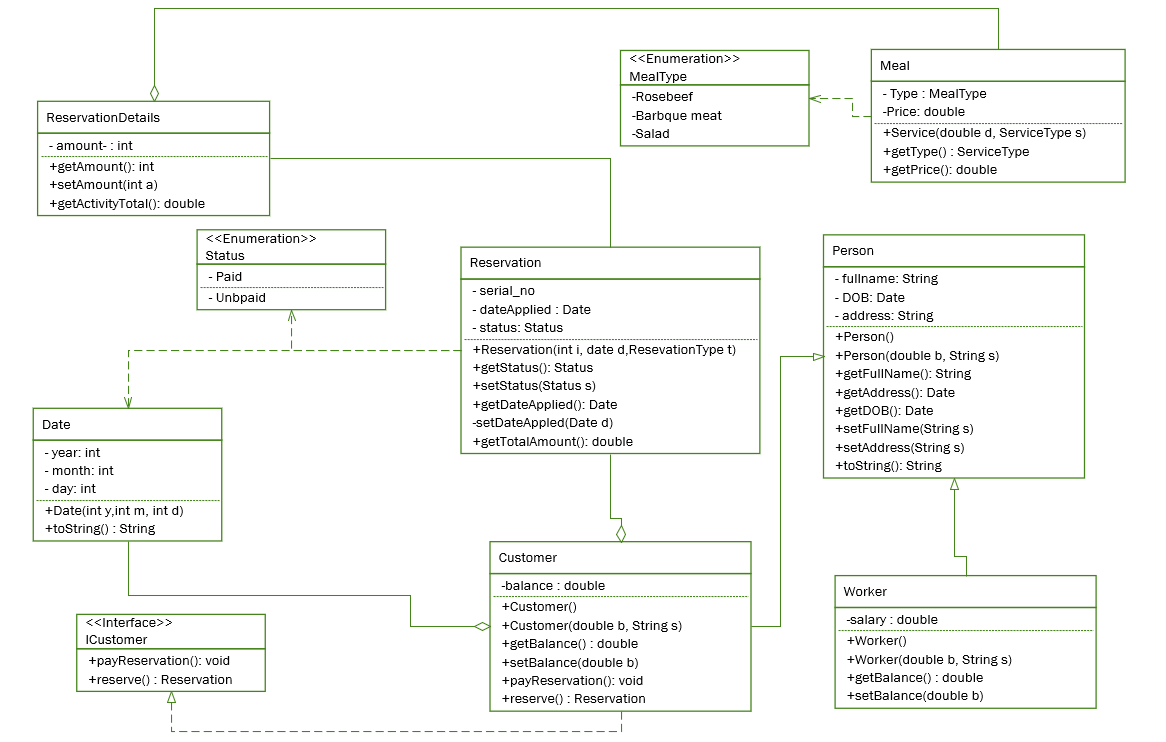
1. **Exit**

Exit from the Restaurant System: this option will be used to end your program. ,

1. **Help**



**Class Diagram**

****

**Person Class is an abstract class and two classes Inherit from it:**

1. **Customer**
2. **Worker**

**Customer has an aggregation relationship with Date**

**Customer Implemeing an interface with two methods**

* **PayReservation()**
* **Reserve()**

**Customer may have 1 or more reservation**

**Each reservation may have 1 or more details**

**Reservation depends on Status**

**Every Reservation Detail has one or more meals**

**Every meal depends on meals type enumeration.**

# Appendix: Code

**package resturant;**

**import java.util.Random;**

# public class Date {

**private int day;**

**private int month;**

**private int year;**

**public Date(int year, int month, int day) {**

**this.year = year;**

**this.month = month;**

**this.day = day;**

**}**

**public Date() {**

**throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.**

**}**

**@Override**

**public String toString() {**

**return day + "/" + month + "/" + year;**

**}**

**}**

**package resturant;**

# public abstract class Person {

**private String fullname;**

**private Date dob;**

**private String address;**

**public Person(){}**

**public Person(String fullname, Date dob, String address) {**

**this.fullname = fullname;**

**this.dob = dob;**

**this.address = address;**

**}**

**public String getFullname() {**

**return fullname;**

**}**

**public void setFullname(String fullname) {**

**this.fullname = fullname;**

**}**

**public Date getDob() {**

**return dob;**

**}**

**public void setDob(Date dob) {**

**this.dob = dob;**

**}**

**public String getAddress() {**

**return address;**

**}**

**public void setAddress(String address) {**

**this.address = address;**

**}**

**@Override**

**public String toString() {**

**return "fullname=" + fullname + ", dob=" + dob + ", address=" + address;**

**}**

**}**

**package resturant;**

**import java.util.ArrayList;**

# public class Customer extends Person {

**private double balance;**

**private ArrayList<Reservation> reservations;**

**public Customer(){}**

**public Customer( String fullname, Date dob, String address,**

**double balance ) {**

**super(fullname, dob, address);**

**this.balance = balance;**

**this.reservations = new ArrayList<Reservation>();**

**}**

**public Customer(ArrayList<Reservation> reservations, String fullname, Date dob, String address,**

**double balance) {**

**super(fullname, dob, address);**

**this.balance = balance;**

**this.reservations = reservations;**

**}**

**public double getBalance() {**

**return balance;**

**}**

**public void setBalance(double balance) {**

**this.balance = balance;**

**}**

**public ArrayList<Reservation> getReservations() {**

**return reservations;**

**}**

**public void setReservations(ArrayList<Reservation> reservations) {**

**this.reservations = reservations;**

**}**

**public void payReservation(int serialNo){**

**for(Reservation r: reservations){**

**if(r.getSerialNo() == serialNo){**

**double amount = r.getReservationAmount();**

**this.balance -= amount;**

**}**

**else if(r.getSerialNo() != serialNo){**

**System.out.println("Reservation was not found!");**

**}**

**}**

**}**

**public void reserve(Reservation r){**

**this.reservations.add(r);**

**}**

**@Override**

**public String toString() {**

**return super.toString()+" balance=" + balance;**

**}**

**}**

**package resturant;**

# public class Worker extends Person{

**private double salary;**

**public Worker(double salary) {**

**this.salary = salary;**

**}**

**public Worker(double salary, String fullname, Date dob, String address) {**

**super(fullname, dob, address);**

**this.salary = salary;**

**}**

**public double getSalary() {**

**return salary;**

**}**

**public void setSalary(double salary) {**

**this.salary = salary;**

**}**

**@Override**

**public String toString() {**

**return super.toString()+" salary=" + salary;**

**}**

**}**

**package resturant;**

# public interface ICustomer {

**public void reserve(Reservation r);**

**public void payReservation(int serialNo);**

**}**

**package resturant;**

# enum MealType {Rosebeef, grilled\_Chiken, barbque\_meat, salad}

# public class Meal {

**private MealType type;**

**private double price;**

**public Meal(MealType type, double price) {**

**this.type = type;**

**this.price = price;**

**}**

**@Override**

**public String toString() {**

**return "Activity{" + "type=" + type + ", price=" + price + '}';**

**}**

**public MealType getType() {**

**return type;**

**}**

**public double getPrice() {**

**return price;**

**}**

**}**

**package resturant;**

**import java.util.ArrayList;**

# enum Status {paid, unpaid}

# public class Reservation {

**private int serialNo;**

**private Date dateApplied ;**

**private Status status;**

**private ArrayList<ReservationDetail> reservations;**

**public Reservation(int serialNo, Date dateApplied, Status status) {**

**this.serialNo = serialNo;**

**this.dateApplied = dateApplied;**

**this.status = status;**

**this.reservations = new ArrayList<ReservationDetail>() ;**

**}**

**public Reservation(int serialNo, Date dateApplied, Status status, ArrayList<ReservationDetail> reservations) {**

**this.serialNo = serialNo;**

**this.dateApplied = dateApplied;**

**this.status = status;**

**this.reservations = reservations;**

**}**

**Reservation() {**

**throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.**

**}**

**public int getSerialNo() {**

**return serialNo;**

**}**

**public void setSerialNo(int serialNo) {**

**this.serialNo = serialNo;**

**}**

**public Date getDateApplied() {**

**return dateApplied;**

**}**

**public void setDateApplied(Date dateApplied) {**

**this.dateApplied = dateApplied;**

**}**

**public Status getStatus() {**

**return status;**

**}**

**public void setStatus(Status status) {**

**this.status = status;**

**}**

**public ArrayList<ReservationDetail> getReservations() {**

**return reservations;**

**}**

**public void setReservations(ArrayList<ReservationDetail> reservations) {**

**this.reservations = reservations;**

**}**

**public void addDetail(ReservationDetail rd){**

**this.reservations.add(rd);**

**}**

**public double getReservationAmount(){**

**double total = 0;**

**for(ReservationDetail r: reservations){**

**total += r.getActivityTotal();**

**return total;**

**}**

**return 0;**

**}**

**}**

**package resturant;**

# public class ReservationDetail {

**private Meal activityType;**

**private int amount;**

**public ReservationDetail(Meal activityType, int amount) {**

**this.activityType = activityType;**

**this.amount = amount;**

**}**

**public int getAmount() {**

**return amount;**

**}**

**public void setAmount(int amount) {**

**this.amount = amount;**

**}**

**public double getActivityTotal(){**

**return this.activityType.getPrice()\*this.amount;**

**}**

**}**

**package resturant;**

**import java.util.ArrayList;**

**import java.util.Random;**

**import java.util.Scanner;**

# public class Resturant {

**private static ArrayList<Meal> activities = new ArrayList();**

**private static ArrayList<Customer> customers = new ArrayList();**

**private static ArrayList<Worker> workers = new ArrayList();**

**private static int serialSequence = 100;**

**public static void main(String[] args) {**

**activityList();**

**Scanner i = new Scanner(System.in);**

**menu();**

**System.out.println("Enter your choice: ");**

**int option = i.nextInt();**

**int index = 0;**

**String name = "";**

**while(option != 7){**

**switch(option){**

**case 1:**

**Customer c = addCustomer();**

**customers.add(c);**

**System.out.println("New customer added...");**

**break;**

**case 2:**

**System.out.println("Enter customer name to add reservation for : ");**

**name = i.next();**

**index = getCustomerIndex(name);**

**if(index >=0){**

**Reservation newReserve = addReservationDetail();**

**customers.get(index).reserve(newReserve);**

**System.out.println("Reservation was made with serial no: "+newReserve.getSerialNo());**

**}**

**else{**

**System.out.println("Not found!");**

**}**

**break;**

**case 3:**

**System.out.println("Enter customer name to add reservation for : ");**

**name = i.next();**

**index = getCustomerIndex(name);**

**if(index >=0){**

**System.out.println("Enter Reservation serial to pay : ");**

**int serialNo = i.nextInt();**

**int resIndex = getCustomerReservationIndex(customers.get(index), serialNo);**

**if(resIndex >= 0){**

**int serialToPay = customers.get(index).getReservations().get(resIndex).getSerialNo();**

**customers.get(index).payReservation( serialToPay);**

**System.out.println("Customer new balance after payment: "+ customers.get(index).getBalance());**

**}**

**else{**

**System.out.println("Reservation serial was Not found!");**

**}**

**}**

**else{**

**System.out.println("Not found!");**

**}**

**break;**

**case 4:**

**printCustomers();**

**break;**

**case 5:**

**Worker w = addWorker();**

**workers.add(w);**

**System.out.println("New Worker added..."+w.getFullname());**

**break;**

**case 6:**

**printWorkers();**

**break;**

**case 8:**

**menu();**

**break;**

**default:**

**System.out.println("Invalid Choice !");**

**break;**

**}**

**// menu();**

**if( option != 7){**

**System.out.println("Enter your choice: ");**

**option = i.nextInt();**

**}**

**}**

**System.out.println("System has shut down...");**

**}**

**public static void activityList(){**

**activities.add (new Meal (MealType.Rosebeef, 150));**

**activities.add( new Meal (MealType.barbque\_meat, 100));**

**activities.add( new Meal (MealType.salad, 250));**

**}**

**public static Date randomDate(){**

**Random r = new Random();**

**int year = r.nextInt(2021 - 2000 + 1) + 2000;**

**int month = r.nextInt(12 - 1 + 1) + 1;**

**int day = r.nextInt(30 - 1 + 1) + 1;**

**Date d = new Date(year,month,day);**

**return d;**

**}**

**public static Customer addCustomer(){**

**Random r = new Random();**

**Scanner i = new Scanner(System.in);**

**int random = r.nextInt(4 - 0 + 1) + 0;**

**int randomBalance = r.nextInt(25000 - 5000 + 1) + 5000;**

**String[] addresses = {"Jeddah-23214","Riyad-23312","Tabuk-23155","Dammam-23143","Rabig-23666"};**

**System.out.println("Enter fullname: ");**

**String fullname = i.next();**

**// System.out.println("Enter salary");**

**// double salary = i.nextDouble();**

**Customer c = new Customer(fullname, randomDate(), addresses[random], randomBalance);**

**return c;**

**}**

**public static void printCustomers(){**

**if(customers.isEmpty()){**

**System.out.println("No customers found!");**

**// System.out.println("Enter your choice: ");**

**}**

**else{**

**customers.forEach((c) -> {**

**System.out.println(c.toString());**

**});**

**}**

**}**

**public static void test(){**

**System.out.println(activities.get(0).toString());**

**}**

**public static int getCustomerReservationIndex(Customer c, int serialPassed){**

**int index=0;**

**for(Reservation reserve : c.getReservations()){**

**if(reserve.getSerialNo() == serialPassed){**

**return index;**

**}**

**index++;**

**}**

**return -1;**

**}**

**public static Reservation addReservationDetail(){**

**Reservation r = new Reservation(++serialSequence, randomDate(), Status.paid);**

**Scanner i = new Scanner(System.in);**

**System.out.println("choose amount: ");**

**int amount = i.nextInt();**

**System.out.println("choose service type: \n"+**

**"1. Rosebeef"+**

**"2. Barbque meat"+**

**"3. Salad\n"**

**);**

**int choice = i.nextInt();**

**ReservationDetail rd ;**

**switch(choice){**

**case 1:**

**rd = new ReservationDetail(activities.get(0), amount);**

**r.addDetail(rd);**

**break;**

**case 2:**

**rd = new ReservationDetail(activities.get(1), amount);**

**r.addDetail(rd);**

**break;**

**case 3:**

**rd = new ReservationDetail(activities.get(2), amount);**

**r.addDetail(rd);**

**break;**

**}**

**return r;**

**}**

**public static int getCustomerIndex(String name){**

**int index = 0;**

**boolean found = false;**

**for(Customer customer: customers){**

**if( customer.getFullname().toLowerCase().equals(name.toLowerCase())){**

**found = true;**

**if( found)**

**return index;**

**}**

**index++;**

**}**

**return -1;**

**}**

**public static Worker addWorker(){**

**Random r = new Random();**

**Scanner i = new Scanner(System.in);**

**int random = r.nextInt(4 - 0 + 1) + 0;**

**int randomSalary = r.nextInt(8000 - 2500 + 1) + 2500;**

**String[] addresses = {"Jeddah-23214","Riyad-23312","Tabuk-23155","Dammam-23143","Rabig-23666"};**

**String[] randomWorkerName = {"Ahamd","Faisal","Sahar","Luna","Joli"};**

**Worker c = new Worker(randomSalary, randomWorkerName[random], randomDate(), addresses[random]);**

**return c;**

**}**

**public static void printWorkers(){**

**if(workers.isEmpty()){**

**System.out.println("No employees found!");**

**// System.out.println("Enter your choice: ");**

**}**

**else{**

**workers.forEach((c) -> {**

**System.out.println(c.toString());**

**});**

**}**

**}**

**public static void menu (){**

**System.out.println(**

**"\*\*\*\*\* Welcome to Resturant \*\*\*\*\*\n" +**

**"1. Add An Customer\n" +**

**"2. Add a Reservation for a customer\n" +**

**"3. Pay for a Reservation \n" +**

**"4. Print Customers \n" +**

**"5. Add an Employee \n" +**

**"6. print Employees \n" +**

**"7. For Exit \n" +**

**"8. For Help \n"**

**);**

**}**

**}**