**CS1073  
FR03B**

**Assignment #5**

**Daniyal Khan  
3765942**

**Question I:**

/\*\*

This class represents a train ticket

@author Daniyal Khan 3765942

\*/

public class TrainTicket {

/\*\*

Name of the passenger

\*/

private final String name;

/\*\*

Number of checked bags of the passenger

\*/

private int bags;/\*\*

The trip distance the passenger will be travelling

\*/

private final double distance;

/\*\*

Contructs a train ticket object given the name, number of bags and trip distance of the passenger

@param name Name of the passenger

@param bags Number of checked bags

@param distance Trip distance

\*/

public TrainTicket(String name, int bags, double distance) {

this.name = name;

this.bags = bags;

this.distance = distance;

}

/\*\*

Returns the name of the passenger

@return Name of the passenger

\*/

public String passengerName() {

return name;

}

/\*\*

Returns the checked bags of the passenger

@return Number of checked bags of the passenger

\*/

public int checkedBags() {

return bags;

}

/\*\*

Returns trip distance the passenger will be travelling

@return Trip Distance

\*/

public double tripDistance() {

return distance;

}

/\*\*

Returns the cost of ticket depending upon the rate

@param rate Rate of the trip

@return Cost of the ticket

\*/

public double computeCost(double rate) {

return rate\*distance + bags\*45;

}

}

/\*\*

This class is a subclass of TrainTicket and represents Business ticket

@author Daniyal Khan 3765942

\*/

public class BusinessTicket extends TrainTicket {

/\*\*

If the seat is reclining or not

\*/

private boolean recliningSeat;

/\*\*

Contructs a BusinessTicket object given the name, number of bags, trip distance and if seat is reclining or not

@param name Name of the passenger

@param bags Number of checked bags

@param distance Trip Distance

@param recliningSeat If the seat reclines or not

\*/

public BusinessTicket(String name, int bags, double distance, boolean recliningSeat) {

super(name, bags, distance);

this.recliningSeat = recliningSeat;

}

/\*\*

Calculates the cost of ticket for Business category

@param rate Rate of the trip

@return Total Cost of the ticket

\*/

public double computeCost(double rate) {

double totalCost = super.computeCost(rate) + 104.50;

if (recliningSeat) {

totalCost += 55.75;

}

return totalCost;

}

}

/\*\*

This class is a subclass of TrainTicket and represents Sleeper car tickets

@author Daniyal Khan 3765942

\*/

public class SleepCarTicket extends TrainTicket {

/\*\*

The number of nights passenger will be staying

\*/

private int numNights;

/\*\*

Contucts a SleepCarTicket object given the name, number of bags, distance of trip and number of nights

@param name Name of the passenger

@param bags Number of bags

@param distance Trip Distance

@param numNights Number of nights

\*/

public SleepCarTicket(String name, int bags, double distance, int numNights) {

super(name, bags, distance);

this.numNights = numNights;

}

/\*\*

Calculates the cost of ticket for Sleep Car category

@param rate Rate of the trip

@return Total Cost of the ticket

\*/

public double computeCost(double rate) {

double totalCost = super.computeCost(rate) + 167;

totalCost += 95; // cost of first night

if (numNights > 1) {

totalCost += (numNights - 1)\*80; // cost of every night excluding first night

}

return totalCost;

}

}

import java.text.NumberFormat;

/\*\*

This is a driver class for TrainTicket

@author Daniyal Kahan 3765942

\*/

public class TicketApp {

public static void main(String[] args) {

NumberFormat format = NumberFormat.getCurrencyInstance();

format.setMaximumFractionDigits(2);

format.setMinimumFractionDigits(2);

TrainTicket grace = new TrainTicket("Grace Hopper", 2, 1150);

TrainTicket joy = new TrainTicket("Joy Buolamwini", 1, 820);

TrainTicket tim = new TrainTicket("Tim Berners-Lee", 2, 755);

TrainTicket margaret = new BusinessTicket("Margaret Hamilton", 0, 690, true);

TrainTicket barbara = new BusinessTicket("Barbara Liskov", 3, 1402, false);

TrainTicket shafi = new BusinessTicket("Shafi Goldwasser", 1, 975, true);

TrainTicket alan = new SleepCarTicket("Alan Turing", 4, 3327, 3);

TrainTicket lynn = new SleepCarTicket("Lynn Conway", 0, 1766, 1);

TrainTicket jeannette = new SleepCarTicket("Jeannette Wing", 3, 5003, 4);

System.out.println("Passenger & Trip Distance\tBags\tCost");

System.out.println("=========================\t=====\t========");

System.out.println(grace.passengerName() + " (" + grace.tripDistance() + "km) " + "\t" + grace.checkedBags() + "\t" + format.format(grace.computeCost(1.35)));

System.out.println(joy.passengerName() + " (" + joy.tripDistance() + "km) " + "\t" + joy.checkedBags() + "\t" + format.format(joy.computeCost(0.75)));

System.out.println(tim.passengerName() + " (" + tim.tripDistance() + "km) " + "\t" + tim.checkedBags() + "\t" + format.format(tim.computeCost(1.15)));

System.out.println(margaret.passengerName() + " (" + margaret.tripDistance() + "km) " + "\t" + margaret.checkedBags() + "\t" + format.format(margaret.computeCost(1.20)));

System.out.println(barbara.passengerName() + " (" + barbara.tripDistance() + "km) " + "\t" + barbara.checkedBags() + "\t" + format.format(barbara.computeCost(0.95)));

System.out.println(shafi.passengerName() + " (" + shafi.tripDistance() + "km) " + "\t" + shafi.checkedBags() + "\t" + format.format(shafi.computeCost(0.85)));

System.out.println(alan.passengerName() + " (" + alan.tripDistance() + "km) " + "\t" + alan.checkedBags() + "\t" + format.format(alan.computeCost(0.85)));

System.out.println(lynn.passengerName() + " (" + lynn.tripDistance() + "km) " + "\t" + lynn.checkedBags() + "\t" + format.format(lynn.computeCost(0.65)));

System.out.println(jeannette.passengerName() + " (" + jeannette.tripDistance() + "km)" + "\t" + jeannette.checkedBags() + "\t" + format.format(jeannette.computeCost(0.65)));

}

}

**Output:**

Passenger & Trip Distance Bags Cost

========================= ====== =========

Grace Hopper (1150.0km) 2 $1,642.50

Joy Buolamwini (820.0km) 1 $660.00

Tim Berners-Lee (755.0km) 2 $958.25

Margaret Hamilton (690.0km) 0 $988.25

Barbara Liskov (1402.0km) 3 $1,571.40

Shafi Goldwasser (975.0km) 1 $1,034.00

Alan Turing (3327.0km) 4 $3,429.95

Lynn Conway (1766.0km) 0 $1,409.90

Jeannette Wing (5003.0km) 3 $3,888.95

**Question II:**

import java.util.Scanner;

/\*\*

This class converts integers to ancient letters

@author Daniyal Khan 3765942

\*/

public class ArabicToAncient {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.print("Enter a positive integer: ");

int input = scan.nextInt();

scan.nextLine();

System.out.println("The ancient number for " + input + " is: ");

while (input != 0) {

int remainderValue = input % 16;

if (remainderValue == 0) {

System.out.print("X");

}

if (remainderValue >= 4) {

while(remainderValue >=4) {

System.out.print("M");

remainderValue -= 4;

}

}

if (remainderValue < 4) {

while (remainderValue > 0) {

System.out.print("|");

remainderValue--;

}

System.out.println();

}

input /= 16;

}

}

}

**Output:**

A computer screen shot of a program

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a program

Description automatically generated