// Paper 22 Problem-solving and Programming - PRE-RELEASE MATERIAL - 0478/22 - MAY/JUNE 2021

// TASK 1 VARIABLE DECLARATIONS

DECLARE UpTime : ARRAY[1:4] OF STRING

DECLARE UpSeats : ARRAY[1:4] OF INTEGER

DECLARE UpPassengers : ARRAY[1:4] OF INTEGER

DECLARE UpMoneyTotal : ARRAY[1:4] OF REAL

DECLARE DownTime : ARRAY[1:4] OF STRING

DECLARE DownSeats : ARRAY[1:4] OF INTEGER

DECLARE DownPassengers : ARRAY[1:4] OF INTEGER

DECLARE DownMoneyTotal : ARRAY[1:4] OF REAL

UpTime <- {"09:00", "11:00", "13:00", "15:00"}

UpSeats <- {480, 480, 480, 480}

UpPassengers <- {0, 0, 0, 0}

UpMoneyTotal <- {0.0, 0.0, 0.0, 0.0}

DownTime <- {"10:00", "12:00", "14:00", "16:00"}

DownSeats <- {480, 480, 480, 640}

DownPassengers <- {0, 0, 0, 0}

DownMoneyTotal <- {0.0, 0.0, 0.0, 0.0}

DECLARE index : INTEGER //for Loops

// TASK 1 ALGORITHM

PRINT ">>>>> TRAIN JOURNEY DISPLAY <<<<<"

FOR index <- 1 TO 4

PRINT ("Journey No: ", index, "| Departure Hour: ", UpTime[index], "| Tickets available: ", UpSeats[index])

PRINT ("Journey No: ", index, "| Return Hour: ", DownTime[index], "| Tickets available: ", DownSeats[index])

PRINT "---------"

NEXT index

// TASK 2 STARTS ON NEXT PAGE

// TASK 2 VARIABLE DECLARATIONS

DECLARE FreeTickets <- 0 : INTEGER

DECLARE CONSTANT OneWayTicket <- 25.0 : REAL

DECLARE OneWayCost <- 0.0 : REAL

DECLARE choice : BOOLEAN

DECLARE NumOfPassengers, UpTrip, DownTrip, index : INTEGER

// TASK 2 ALGORITHM

PRINT "Do you want to buy ticket(s)? Enter True or False: "

INPUT choice

WHILE choice != True AND choice != False DO

PRINT "Invalid Input! Enter True or False: "

INPUT choice

ENDWHILE

WHILE choice = True DO

PRINT "Enter Journey number for your chosen departure hour: "

INPUT UpTrip

WHILE UpTrip < 1 OR UpTrip > 4 DO

PRINT "Error! Enter Journey number from (1, 2, 3, 4): "

INPUT UpTrip

ENDWHILE

PRINT "----- Return Hours Available -----"

FOR index <- UpTrip TO 4

PRINT "Journey No:", index, " | Return Hour:", DownTime[index], " | Remaining Tickets:", DownSeats[index],

NEXT index

PRINT "Enter Journey number for your chosen Return hour: "

INPUT DownTrip

WHILE DownTrip < UpTrip OR DownTrip > 4 DO

PRINT "Error! Enter Journey number from the given list above: "

INPUT DownTrip

ENDWHILE

PRINT "Enter number of passengers for trip: "

INPUT NumOfPassengers

WHILE NumOfPassengers <= 0 DO

PRINT "Error! Enter number greater than 0: "

INPUT NumOfPassengers

ENDWHILE

IF NumOfPassengers > UpSeat[UpTrip] OR NumOfPassengers > DownSeats[DownTrip]

THEN

PRINT "Tickets not available for chosen hours"

PRINT "Please check the display below for available tickets =>"

ELSE

PRINT "/// Tickets BOOKED! ///"

IF NumOfPassengers >= 10 and NumOfPassengers <= 80

THEN

FreeTickets <- NumOfPassengers DIV 10 // DIV is INTEGER DIVISION

ELSE

FreeTickets <- 0

ENDIF

OneWayCost <- (NumOfPassengers - FreeTickets) \* OneWayTicket

PRINT "Total price for two-way journey: $", OneWayCost \* 2

UpPassengers[UpTrip] <- UpPassengers[UpTrip] + NumOfPassengers

UpSeats[UpTrip] <- UpSeats[UpTrip] - NumOfPassengers

UpMoneyTotal[UpTrip] <- UpMoneyTotal[UpTrip] + OneWayCost

DownPassengers[DownTrip] <- DownPassengers[DownTrip] + NumOfPassengers

DownSeats[DownTrip] <- DownSeats[DownTrip] - NumOfPassengers

DownMoneyTotal[DownTrip] <- DownMoneyTotal[DownTrip] + OneWayCost

ENDIF

PRINT ">>>>> TRAIN JOURNEY DISPLAY <<<<<"

FOR index <- 1 TO 4

IF UpSeats[index] != 0

THEN

PRINT ("Journey No: ", index, "| Departure Hour: ", UpTime[index], "| Tickets Available: ", UpSeats[index])

ELSE

PRINT ("Journey No: ", index, "| Departure Hour: ", UpTime[index], "| Closed!")

ENDIF

IF DownSeats[index] != 0

THEN

PRINT ("Journey No: ", index, "| Return Hour: ", DownTime[index], "| Tickets Available: ", DownSeats[index])

ELSE

PRINT ("Journey No: ", index, "| Return Hour: ", DownTime[index], "| Closed!")

ENDIF

NEXT index

PRINT "Do you want to buy ticket(s)? Enter True or False"

INPUT choice

WHILE choice != True AND choice != False DO

PRINT "Invalid Input! Enter True or False: "

INPUT choice

ENDWHILE

ENDWHILE

// TASK 3 VARIABLE DECLARATIONS

DECLARE TotalAmount <- 0.0 : REAL

DECLARE TotalPassengers, MostPassengers <- 0 : INTEGER

DECLARE MaxTrain : STRING

DECLARE index : INTEGER

// TASK 3 ALGORITHM

PRINT "----- END OF THE DAY -----"

FOR index <- 1 TO 4

PRINT ("Journey No: ", index, "| Train Departure Hour: ", UpTime[index], "| No. of passengers: ", UpPassengers[index],

"| Total money: ", UpMoneyTotal[index])

PRINT ("Journey No: ", index, "| Train Return Hour: ", DownTime[index], "| No. of passengers: ", DownPassengers[index],

"| Total money: ", DownMoneyTotal[index])

PRINT "" // EMPTY LINE

NEXT index

FOR index <- 1 TO 4

TotalPassengers <- TotalPassengers + UpPassengers[index]

TotalAmount <- TotalAmount + (UpMoneyTotal[index] \* 2)

NEXT index

FOR index <- 1 TO 4

IF UpPassengers[index] > MostPassengers

THEN

MostPassengers <- UpPassengers[index]

MaxTrain <- UpTime[index]

ENDIF

IF DownPassengers[index] > MostPassengers

THEN

MostPassengers <- DownPassengers[index]

MaxTrain <- DownTime[index]

ENDIF

NEXT index

PRINT "Total money earned today:", TotalAmount

PRINT "Total passengers travelled today:", TotalPassengers

PRINT "The train journey with the highest number of passengers today:", MaxTrain

Link to the GitHub Pseudocode Repository: https://github.com/Dunroxiz/Pre-release-Material-2021-P22-MJ-CIE