```
// Paper 2 Problem-solving and Programming - PRE-RELEASE MATERIAL - 0478/22 - MAY/JUNE 2021

// TASK 1 VARIABLE DECLARATIONS

DECLARE UpTime : ARRAY[0:3] OF STRING
DECLARE UpSeats : ARRAY[0:3] OF INTEGER
DECLARE UpPassengers : ARRAY[0:3] OF INTEGER
DECLARE UpMoneyTotal : ARRAY[0:3] OF REAL

DECLARE DownTime : ARRAY[0:3] OF STRING
DECLARE DownSeats : ARRAY[0:3] OF INTEGER
DECLARE DownPassengers : ARRAY[0:3] OF INTEGER
DECLARE DownMoneyTotal : ARRAY[0:3] 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 <- 0 TO 3
    IF UpSeats[index] != 0
        THEN
            PRINT ("Train No: ", index, "| Train Departure Hour: ", UpTime[index], "| Remaining Tickets: ", UpSeats[index])
        ELSE
            PRINT ("Train No: ", index, "| Train Departure Hour: ", UpTime[index], "| Closed!")
    ENDIF

    IF DownSeats[index] != 0
        THEN
            PRINT ("Train No: ", index, "| Train Return Hour: ", DownTime[index], "| Remaining Tickets: ", DownSeats[index])
        ELSE
            PRINT ("Train No: ", index, "| Train Return Hour: ", DownTime[index], "| Closed!")
    ENDIF
NEXT index
```

```
// 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)? 'True' for yes and 'False' for no"
INPUT choice
WHILE choice != True AND choice != False DO
    PRINT "Enter 'True' for yes and 'False' for no: "
    INPUT choice
ENDWHILE

WHILE choice = True DO
    PRINT "Enter Train number corresponding to your departure hour: "
    INPUT UpTrip
    WHILE UpTrip < 0 AND UpTrip > 3 DO
        PRINT "Error! Enter train number from (0, 1, 2, 3): "
        INPUT UpTrip
    ENDWHILE

    PRINT "----- Return Hours Available -----"
    FOR index <- UpTrip TO 3
        PRINT "Train No:", index, " | Return Hour:", DownTime[index], " | Remaining Tickets:", DownSeats[index],
    NEXT index

    PRINT "Enter Train number corresponding to your Return hour: "
    INPUT DownTrip
    WHILE DownTrip < UpTrip OR DownTrip > 3:
        PRINT "Error! Enter Train number from the given list above: "
        INPUT DownTrip
    ENDWHILE

    PRINT "Enter number of passengers for trip: "
    INPUT NumOfPassengers
    WHILE NumOfPassengers <= 0:
        PRINT "Error! Enter number greater than 0: "
        INPUT NumOfPassengers
    ENDWHILE

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

```
        PRINT "Seats not available for chosen hours"
        PRINT "Please check the display below for available Seats =>"
    ELSE
        PRINT "/// SEATS BOOKED ///"
        IF NumOfPassengers >= 10 and NumOfPassengers <= 80
            THEN
                FreeTickets = NumOfPassengers DIV 10 // DIV is INTEGER DIVISION
            ELSE
                FreeTickets = 0
        ENDIF

        OneWayCost = (NumOfPassengers - FreeTickets) * OneWayTicket

        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 <- 0 TO 3
    IF UpSeats[index] != 0
        THEN
            PRINT ("Train No: ", index, "| Train Departure Hour: ", UpTime[index], "| Remaining Tickets: ", UpSeats[index])
        ELSE
            PRINT ("Train No: ", index, "| Train Departure Hour: ", UpTime[index], "| Closed!")
    ENDIF

    IF DownSeats[index] != 0
        THEN
            PRINT ("Train No: ", index, "| Train Return Hour: ", DownTime[index], "| Remaining Tickets: ", DownSeats[index])
        ELSE
            PRINT ("Train No: ", index, "| Train Return Hour: ", DownTime[index], "| Closed!")
    ENDIF
NEXT index

PRINT "Do you want to buy ticket(s)? 'True' for yes and 'False' for no"
INPUT choice
WHILE choice != True AND choice != False DO
    PRINT "Enter 'True' for yes and 'False' for no: "
    INPUT choice
ENDWHILE
ENDWHILE
```

```
// TASK 3 VARIABLE DECLARATIONS
DECLARE TotalAmount <- 0.0 : REAL
DECLARE TotalPassengers, MostPassengers <- 0 : REAL
DECLARE MaxTrain : STRING
DECLARE index : INTEGER

// TASK 2 ALGORITHM

PRINT "----- END OF THE DAY -----"
FOR index <- 0 TO 3
    PRINT ("Train No: ", index, "| Train Departure Hour: ", UpTime[index], "| Number of passengers: ", UpPassengers[index],
           "| Total money: ", UpMoneyTotal[index])
    PRINT ("Train No: ", index, "| Train Return Hour: ", DownTime[index], "| Number of passengers: ", DownPassengers[index],
           "| Total money: ", DownMoneyTotal[index])
    PRINT ""  // EMPTY LINE
NEXT index

FOR index <- 0 TO 3
    TotalPassengers = TotalPassengers + UpPassengers[index]
    TotalAmount = TotalAmount + (UpMoneyTotal[index] * 2)
ENDIF

FOR index <- 0 TO 3
    IF UpPassengers[count] > MostPassengers
        THEN
            MostPassengers = UpPassengers[count]
            MaxTrain = UpTime[count]
    ENDIF
    IF DownPassengers[count] > MostPassengers
        THEN
            MostPassengers = DownPassengers[count]
            MaxTrain = DownTime[count]
    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
```