PROGRAMMING PROJECT TASK

AQA NEA Report

BARTHOLOMEW SCHOOL  Centre Number: 62233

2021

For candidates entering for June 2021 8520 examination

Analysis

Text

Description automatically generated

Note: some prerequisites:

Text, letter

Description automatically generatedAlso, airports.txt file was edited to include the aircraft information as I believed It was a more elegant solution

There are also some differences between the pseudocode and the python script, due to syntax differences and functions needing to be written differently.

Text, letter

Description automatically generated

Text, letter

Description automatically generated

Task 1

Design

**f → open(“airports.txt”, “r”) # Opens file and assigns name “f”**

**reader → csv.reader(f, delimiter=”,”)**

**FOR row IN reader # For loop running through all rows in f**

**IF foreignCode in row[0] THEN**

**OUTPUT “Your destination airport is”, row[1]**

**ENDIF**

**ELSE**

**OUTPUT “Error: airport not found”**

**ENDFOR**

Implementation

Text

Description automatically generated

Testing

Text

Description automatically generated

Above is the output from the file reading in airportDetails().

Task 2

Design

**SUBROUTINE**

**menu():**

**OUTPUT “ # Prints menu options**

**1: Enter Airport Details**

**2: Enter Flight Details**

**3: Enter Price Plan and Calculate Profit**

**4: Clear Data**

**5: Quit**

**”**

**choice → USERINPUT “Enter menu choice (1-5): ” # User chooses menu option**

**IF choice = 1 THEN # User chose option 1**

**airportDetails()**

**ELSE IF choice = 2 THEN # User chose option 2**

**flightDetails()**

**ELSE IF choice = 3 THEN # User chose option 3**

**priceProfit()**

**ELSE IF choice = 4 THEN # User chose option 4**

**Information → [] # Empties all data in information**

**OUTPUT “Entered data has been cleared”**

**menu()**

**ELSE IF choice = 5 THEN # User chose option 5**

**OUTPUT “Thanks for using airport program”**

**quit() # Python function to quit running program**

**ELSE # User chose an incorrect option**

**OUTPUT “Invalid Choice”**

**menu()**

**ENDIF**

Implementation

Text

Description automatically generated

TestingText

Description automatically generated

Task 3

Design

**ELSE IF choice = 5 THEN # User chose option 5**

**OUTPUT “Thanks for using airport program”**

**quit() # Python program quit function**

**ENDIF**

Implementation

Graphical user interface, text

Description automatically generated

Testing

Text

Description automatically generated

Task 4

Design

**SUBROUTINE**

**airportDetails()**

**localCode → USERINPUT “Enter three digit airport code: ”**

**IF localCode.lower() != “lpl” and localCode.lower() != “boh” THEN**

**OUTPUT “Error: Invalid Code”**

**menu()**

**ENDIF**

**information.append(localCode)**

**foreignCode → USERINPUT “Enter three digit foreign airport code: ”.upper()**

**f → open(“airports.txt”, “r”) # Opens file**

**reader → csv.reader(f, delimiter=”,”)**

**FOR row IN reader # For loop running through all rows in f**

**IF foreignCode in row[0] THEN**

**OUTPUT “Your destination airport is”, row[1]**

**ENDIF**

**ELSE**

**OUTPUT “Error: airport not found”**

**ENDFOR**

**Information.append(foreignCode.upper())**

**close (f)**

**menu()**

Implementation

Text

Description automatically generated

Testing

Text

Description automatically generated

Task 5

Design

**SUBROUTINE**

**flightDetails():**

**aircraft → USERINPUT “Enter aircraft type (medium narrow body, medium wide body, large narrow body):”**

**IF aircraft.lower() != “medium narrow body” AND aircraft.lower() != “medium wide body” AND aircraft.lower() != “large narrow body” THEN**

**OUTPUT “Error: invalid aircraft type”**

**menu()**

**ENDIF**

**information.append(aircraft)**

**f → open(“airports.txt”, “r”)**

**reader → csv.reader(f, delimiter=”,”)**

**FOR row IN reader THEN**

**IF aircraft IN row[0] THEN**

**aircraftType.append(row)**

**OUTPUT “Cost per 100km =”, row[1], “flight range =”, row[2], “capacity (standard) =”, row[3], “minimum first class =”, row[4]**

**ENDIF**

**ELSE**

**OUTPUT “Error: aircraft type not found”**

**ENDFOR**

**firstClass → USERINPUT “Please enter the a mount of first-class seats on aircraft:”**

**IF firstClass > 0 THEN**

**IF firstClass < row[4] THEN**

**OUTPUT “Error: first class seats less than minimum for”, row[0], “aircraft”**

**menu()**

**ELSE IF firstClass > (row[3] / 2) THEN**

**OUTPUT “Error: first class seats surpass maximum for”, row[0], “aircraft”**

**ENDIF**

**ENDIF**

**Information.append(firstClass)**

**Standard → row[3] – (firstClass \* 2)**

**Information.append(standard)**

**OUTPUT “Standard seats on aircraft is”, standard**

**menu()**

**ENDIF**

Implementation

Text

Description automatically generated

Testing

Text

Description automatically generated

Task 6

Design

**SUBROUTINE**

**priceProfit():**

**IF information[0] = “” THEN**

**OUTPUT “Error: local airport not found”**

**menu()**

**ELSE IF information[1] = “” THEN**

**OUTPUT “Error: foreign airport not found”**

**menu()**

**ELSE IF information[2] = “” THEN**

**OUTPUT “Error: aircraft type not found”**

**menu()**

**ELSE IF information[3] = “” THEN**

**OUTPUT “Error: no# first-class seats not found”**

**menu()**

**ELSE IF information[4] = “” THEN**

**OUTPUT “Error: no# standard seats not found”**

**menu()**

**ENDIF**

**airportDistance → distanceCheck()**

**standardPrice → USERINPUT “Enter price of standard seat:”**

**firstClassPrice = USERINPUT “Enter price of first-class seat:”**

**flightCostPerSeat = (aircraftType[0][1] \* airportDistance) / 100**

**OUTPUT “Flight cost per seat is”, flightCostPerSeat**

**flightCost → flightCostPerSeat \* (firstClass + standard)**

**OUTPUT “Flight cost total is”, flightCost”**

**flightIncome → (firstClass \* firstClassPrice) + (standard \* standardPrice)**

**OUTPUT “Flight income is”, flightIncome**

**flightProfit → flightIncome – flightCost**

**OUTPUT “Flight profit is”, flightProfit**

**menu()**

**SUBROUTINE**

**distanceCheck()**

**f = open(“airports.txt”, “r”)**

**reader = csv.reader(f, delimiter=”,”)**

**IF information[0].lower() = “lpl” THEN**

**FOR row IN reader THEN**

**IF information[1] IN row THEN**

**airportDistance → row[2]**

**ELSEIF information[2] IN row THEN**

**maxDistance → row[2]**

**ENDIF**

**ENDFOR**

**ELSE IF information[0].lower() = “boh” THEN**

**FOR row IN reader THEN**

**IF information[1] IN row THEN**

**airportDistance → row[2]**

**IF information[2] IN row THEN**

**maxDistance → row[2]**

**ENDFOR**

**ENDIF**

**IF maxDistance <= airportDistance THEN**

**OUTPUT “Error: distance exceeds maximum travel distance of”, information[2]**

**menu()**

**ENDIF**

**close (f)**

**RETURN airportDistance**

Implementation

Text

Description automatically generatedText

Description automatically generated

Testing

Text

Description automatically generated

Text

Description automatically generated

Directly above is the error output of the distanceCheck subroutine when the aircraft type cannot fly the required distance to the foreign airport. For example, the flight distance from LPL to JFK is 5326 miles, and the medium narrow body type can only travel 2650 miles, so cannot travel the distance to JFK, which the subroutine recognises.

Task 7

Design

Takes place inside the menu() subroutine.

**ELSE IF choice = 4**

**Information = []**

**OUTPUT “Entered data has been cleared”**

**menu()**

**ENDIF**

Implementation

Graphical user interface, text

Description automatically generated

Testing

A close up of a screen

Description automatically generated