**Instructions**

The purpose of the exercise is to evaluate your approach to software development including object-oriented design, design patterns and testing.

* Complete the exercise in the language of your choice.
* We recommend you spend 2-3 hours on your submission.
* Structure your code as if this were a real production application.
* State any assumptions you make as comments in the code. If any aspect of the specification is unclear, state your interpretation of the requirement in a comment.
* Please include instructions on how to run your program.

**The problem**

A small airline wants a simple program that produces flight summary reports based on flight, route and passenger data.

There are three types of passenger the airline will cater for:

1. General – normal fare-paying passengers.
2. Loyalty Members – repeat customers who get benefits for choosing to fly with the airline.
3. Airline Employees – employees of the airline who fly with the airline for free.

For each flight the airline charges a base ticket price for a specific route. Loyalty members can choose to pay with their loyalty points instead. Loyalty points are worth £1 each. Airline employees always fly free. All passengers are allocated 1 bag and loyalty members are allowed 1 extra bag. For simplicity, we assume that every passenger will bring at least 1 bag.

**Your task**

Write a console application that accepts two filenames, the first is an input file, containing route, plane and passenger data, the second is an output file to which the flight summary report must be written.

**Input**

The format of the input file is a set of lines that represent either plane, route or passenger information. Your program should read each line in the input file and process each instruction.

For example:

add route London Dublin 100 150 75

add aircraft Gulfstream-G550 8

add airline Trevor 54

add general Mark 35

add loyalty Joan 56 100 FALSE TRUE

An input file must add only one route and one aircraft.

**Input format specification**

The format of instruction lines is specified below in [ABNF](https://en.wikipedia.org/wiki/Augmented_Backus%E2%80%93Naur_Form).

If you are not familiar with ABNF, take some time to read the [Wikipedia entry](https://en.wikipedia.org/wiki/Augmented_Backus%E2%80%93Naur_Form).

instruction-line = add-route CRLF add-aircraft CRLF 1\*add-passenger

add-route = "add route" SP origin SP destination SP cost-per-passenger SP ticket-price SP minimum-takeoff-load-percentage

add-aircraft = "add aircraft" SP aircraft-title SP number-of-seats

add-passenger = "add" SP (general-passenger / airline-passenger / loyalty-passenger) CRLF

general-passenger = "general" SP first-name SP age

airline-passenger = "airline" SP first-name SP age

loyalty-passenger = "loyalty" SP first-name SP age SP current-loyalty-points SP using-loyalty-points SP using-extra-baggage

origin = identifier ; the name of the origin city

destination = identifier ; the name of the destination city

cost-per-passenger = numeric ; the cost to the airline per passenger of flying

; the route in whole £

ticket-price = numeric ; the price of the ticket in whole £

minimum-takeoff-load-percentage = percentage ; the minimum percentage of the plane's capacity

; that must be used for the route to be able to

; fly

aircraft-title = identifier ; the name of the plane

number-of-seats = numeric ; the total number of seats on the plane

first-name = identifier ; the first name of the passenger

age = numeric ; the age of the passenger in years

current-loyalty-points = numeric ; the number of loyalty points the customer

; currently has, before embarking on the

; current flight

using-loyalty-point = boolean ; whether or not the passenger is using

; loyalty points to pay for the flight

; if the number of loyalty points is less

; than the ticket cost then the customer

; pays the remainder

using-extra-baggage = boolean ; whether or not the passenger is bringing

; an extra bag

percentage = %d0-100

identifier = 1\*ALPHA

numeric = 1\*DIGIT

boolean = "TRUE" / "FALSE"

**Output**

Your program should read the input file, compute a flight summary report and write it to the output file in the following format, again in ABNF:

output-line = total-passenger-count SP

general-passenger-count SP

airline-passenger-count SP

loyalty-passenger-count SP

total-number-of-bags SP

total-loyalty-points-redeemed SP

total-cost-of-flight SP

total-unadjusted-ticket-revenue SP

total-adjusted-revenue SP

can-flight-proceed

total-passenger-count = numeric ; total number of passengers on the flight

general-passenger-count = numeric ; number of general passengers on the flight

airline-passenger-count = numeric ; number of airline passengers on the flight

loyalty-passenger-count numeric ; number of loyalty passengers on the flight

total-number-of-bags = numeric ; the total number of bags on the plane

total-loyalty-points-redeemed = numeric ; the total number of loyalty points redeemed by

; all passengers

total-cost-of-flight = numeric ; the total cost to the airline of running the flight

total-unadjusted-ticket-revenue = numeric ; the total ticket revenue, ignoring loyalty

; and airline passenger adjustments

total-adjusted-revenue = numeric ; the total ticket revenue, after adjusting for

; loyalty members points and airline passengers

can-flight-proceed = boolean ; can the flight proceed, according to the rules

; defined below

numeric = ["-"] 1\*DIGIT

boolean = "TRUE" / "FALSE"

**Flight rules**

A flight proceeds only if all of the following rules are met:

1. The total adjusted revenue for the flight exceeds the total cost of the flight.
2. The number of passengers does not exceed the number of seats on the plane.
3. The percentage of booked seats exceeds the minimum set for the route.

**Example input and output**

**Input**

add route London Dublin 100 150 75

add aircraft Gulfstream-G550 8

add general Mark 35

add general Tom 15

add general James 72

add airline Trevor 54

add loyalty Alan 65 50 FALSE FALSE

add loyalty Susie 21 40 TRUE FALSE

add loyalty Joan 56 100 FALSE TRUE

add general Jack 50

**Output**

8 4 1 3 9 40 800 1200 1010 TRUE

This flight can proceed.

**Input**

add route London Dublin 100 150 75

add aircraft Gulfstream-G550 12

add general Mark 35

add general Tom 15

add general James 72

add general Jack 50

add airline Jane 75

add general Steve 20

**Output**

6 5 1 0 6 0 600 900 750 FALSE

This flight cannot proceed, it is less than 75% full.