Assignment 6 – Airline Tickets

Goals

- Loops
- Partially filled lists
- Navigating lists by explicit index

Background

A small airline has just purchased a computer for its new automated reservations system. You've been asked to develop the new system. You're to write an application to assign seats on each flight of the airline's only (small) plane (capacity: **10 seats**). The basic system will have every seat treated exactly the same (all economy). The "advanced" (extra-credit) system will allow differentiation between first class and economy seats.

Requirements

- Create a new Python file. It must begin with comments in the following format (replace the name and email with your actual information and write text for the description):

```
# Name, USC email
# ITP 115, Spring 2020
# Assignment 6
# Description:
# Describe what this program does.
```

- Use a list of strings to represent the seating chart of the plane.
- Initialize all elements of the list to **""** (*an empty String*) to indicate that all seats are empty *(hint: You must make sure to loop through and assign* **""** *to each element).*
- As each seat is assigned, set the corresponding element of the list to the passenger's name to indicate the seat is taken.
- You should be using loops and if/else or switch statements in your code.
- Hint:
    - Create two variables at the beginning of your program
      TOTAL_SEATS = 10 # the plane's TOTAL capacity which doesn't change
      numFilledSeats = 0 # number of seats filled so far

- Your program should display a menu with the following options, and allow the user to continue to interact with your system until they choose to quit.

```
 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit
```

If "Assign Seat" is chosen:

- Your application should ask for the passenger's name.
- Your application should never assign a seat that has already been assigned
- If the airplane is full, display the message **"Next flight leaves in 3 hours."**

If "Print Seat Map" is chosen:

- Print the seat number and passenger name assigned to that seat, printing the word **Empty** if no passenger has been assigned that seat.
- Please note: seats should be numbered starting from **1** and going up through **TOTAL_SEATS** when displayed for the passengers.

If "Print Boarding Pass" is chosen:

- Let the user type their name **OR** their seat number. Depending on what they type, find the rest of the information (name or seat number) then print the boarding pass.

Finding names by seat number

- Finding the name based on seat number is relatively simple
- You do need to error-check carefully – seat 1 is the first spot in the airplane, not the second *(hint: think about how lists are indexed)*

Finding the seat number by names

- Finding the seat number based on the name is more difficult
- You must look through the entire list to find the person *(i.e. a matching string, and you probably want to ignore the case of the letters)*
- If the passenger cannot be found, display a message "**`No passenger with that information could be found.`**"
- Pseudocode
    - Create variable *(e.g. seatNumber)* to represent the correct seat of the name you are looking for, and assign it a sentinel / default value *(hint: whatever you choose should NOT be a valid seat number)*
    - Look through each position in the string list
        - If you find a match between the list value and the name you are searching for, assign the current index to *seatNumber*
    - When you have finished looking through the list, there are two possibilities
        - If *seatNumber* is still the default / sentinel value, then the passenger could not be found
        - Otherwise *(i.e. seatNumber is assigned a valid choice)*, *seatNumber* represents the correct seat of the passenger

Modifications for Optional Extra Credit for Advanced System

If "Assign Seat" is chosen:

- After getting passenger name, display the following alternatives:
  `Type 1 for First Class or Type 2 for Economy.`

- If the user types 1, your application should assign a seat in the first class sections (seats 1-4). If the user types 2, your application should assign a seat in the economy section (seats 5-10). Note: use **must** use an additional constant within your program to indicate the number of first class seats.)
- Your application should never assign a seat that has already been assigned. When the economy section is full, your application should ask the person it it's acceptable to be placed in the first-class section (and vice versa). If yes, make the appropriate seat assignment. If no, display the message "**Next flight leaves in 3 hours.**")

If "Print Seat Map" is chosen:

- Print "First Class" or "Economy" before the seat number.

Sample Output

```
 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 1
Please enter your first name: Shawn
 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 2
****************************************
      Seat #1:   Shawn
      Seat #2:   Empty
      Seat #3:   Empty
      Seat #4:   Empty
      Seat #5:   Empty
      Seat #6:   Empty
      Seat #7:   Empty
      Seat #8:   Empty
      Seat #9:   Empty
      Seat #10:  Empty
****************************************

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 4
```

```
Please enter choice: 5
Please enter choice: 1
Please enter your first name: Beyoncé
 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 2
**************************************
      Seat #1:   Shawn
      Seat #2:   Beyoncé
      Seat #3:   Empty
      Seat #4:   Empty
      Seat #5:   Empty
      Seat #6:   Empty
      Seat #7:   Empty
      Seat #8:   Empty
      Seat #9:   Empty
      Seat #10:  Empty
**************************************

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 3
Type 1 to get Boarding Pass by seat number
Type 2 to get Boarding Pass by name
1
What is the seat number: 90
Invalid number--no boarding pass found

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 3
Type 1 to get Boarding Pass by seat number
Type 2 to get Boarding Pass by name
1
What is the seat number: 2

======= BOARDING PASS =======
```

```
      Seat #: 2
      Passenger Name: Beyoncé
============================

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 3
Type 1 to get Boarding Pass by seat number
Type 2 to get Boarding Pass by name
2
Enter passenger name: Shawn

======= BOARDING PASS =======
      Seat #: 1
      Passenger Name: Shawn
============================

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> 3
Type 1 to get Boarding Pass by seat number
Type 2 to get Boarding Pass by name
2
Enter passenger name: Blue Ivy
No passenger with name Blue Ivy was found

 1: Assign Seat
 2: Print Seat Map
 3: Print Boarding Pass
-1: Quit

> -1
Have a nice day!
```

Deliverables and Submission Instructions

- Create a folder on your computer called **`ITP115_A6_`*`LastName_FirstName`*** (replace *LastName* with your last/family name and FirstName with your first name).
- Inside the folder, put your python source code.
- Compress the folder (make a zip file). This cannot be done within PyCharm. Find the folder on your computer and compress it.
  a. Windows*:*
     1. Using File Explorer, select your lab file
     2. Right click
     3. Send to ->
     4. Compressed (zipped) folder
  b. Mac OSX:
     1. Using Finder, select your lab file
     2. Right click
     3. Compress "*FileName*"
- Upload the zip file to your Blackboard section:
  1. On Blackboard, click on the Assignments item in the course menu on the left.
  2. Click on the specific item for this assignment (starts with A and a number).
  3. Click on the Browse My Computer button and select your zip file.
  4. Click the Submit button.

Grading

Item	Points
Handles Menu Logic Properly	5
Correctly sets up list	3
Correctly maintains partially filled list	3
Assign Seat code works as specified	3
Print seat map works as specified	3
Boarding Pass by seat number works	3
Boarding Pass by passenger name works	5
Boarding Pass printed according to specs	5
Code validity / checking input	5
Total	**35**

** Points will be deducted for poor code style, lack of error checking, improper submission.*