TCSS 143
Programming Assignment 2

Due: **See Canvas** *(by 11:55pm submitted electronically*).

NOTE: Be sure to adhere to the **University's Policy on Academic Integrity** as discussed in class. Programming assignments are to be written individually and submitted programs must be the result of your **own** efforts. Any suspicion of academic integrity violation will be dealt with accordingly

Purpose: This assignment is designed to demonstrate your understanding of basic Java Concepts

Program 2 is worth 100 points. The points will be assigned according to the following scheme:

Program compiles	20 Points
Documentation	
Block Comments, Inline Comments	5 Points
Indentation, Whitespace	5 Points
Hierarchy Diagram	**5 Points**
Process	
Objects	30 Points
- Fields	
- Constructors	
- Methods	
Two-Dimensional Array	10 Points
Test Client	10 Points
Output	
Correctly printed output (on monitor)	15 Points
Total	**100 Points**

Assignment:

The purpose of this programming project is to demonstrate understanding of 2 dimensional arrays and the creation, implementation, and use of a simple class.

You are to design a class "`LostPuppy.java`" which represents a puppy lost in a multi-floor building that contains the same number of rooms on each floor. During instantiation (or creation) of an object of this class, each room on each floor will be initialized as empty (you will actually use the space ' ' character for this purpose) and a random room will be chosen where the puppy is lost. For this purpose, the character "P" will be placed in this random location. Further details on the constructor is listed below.

An object of this class is used as a game for two players to take turns searching for the puppy, one room at a time until the unfortunate little canine is found. The instantiation of this object and the search will be performed by a "driver" program which has been provided to you allowing you to only have to concentrate on developing the class (the driver program is in the file "`PuppyPlay.java`")

DETAILS

Because the puppy is lost (or placed) in a random location of the building, you will need to import the Random class in the `LostPuppy` class.

Name your class "`LostPuppy.java`".

Fields (of course, all fields are private):

- A character (char) array named `myHidingPlaces`. This represents the building where the rows are floors and the columns are rooms on each floor (this building has an unusual numbering system; the floors and rooms both start at zero).
- Two integers that will hold the floor and room where the puppy is lost, named `myFloorLocation` and `myRoomLocation`.
- A char named `myWinner` which will be assigned the player's character when a player finds the puppy (the driver program uses digits '1' and '2' to more clearly distinguish the players from the puppy).
- A boolean named `myFound` which is set to true when the puppy is found.

Constructor:

- Receives two integer parameters as the user's input for the number of floors and rooms of the building in which the puppy is lost.
- The constructor instantiates the 2D array "`myHidingPlaces`" as a character array with the first parameter for the rows (`theFloors`) and the second parameter as the columns (`theRooms`).
- Initialize `myHidingPlaces`' cells, each to contain a space ' ' (done with single quotes)

- Set `myFloorLocation` (floor puppy is on) randomly based using the first parameter
- Set `myRoomLocation` (room puppy is in) randomly based using the second parameter
- Set `myHidingPlaces[myFloorLocation][myRoomLocation]` to the char 'P'
- Set `myWinner` to a single space
- Set `myFound to` false

Methods:

- **`roomSearchedAlready`** receives the floor and room to be searched and returns true if the room has already been searched, false otherwise.
- **`puppyLocation`** receives the floor and room to be searched and returns true if the floor and room are where the puppy is lost, false otherwise.
- **`indicesOK`** receives the floor and room to be searched and returns true if the floor and room values are within the array indices range, false otherwise (used to check that these indices will not cause an error when applied to the array).
- **`numberOfFloors`** returns how many floors are in the building (the first floor starts at zero).
- **`numberOfRooms`** returns how many rooms are on each floor of the building (the first room starts at zero and all floors have the same number of rooms).
- **`searchRoom`** receives the floor and room to be searched and also the current player (as a char type) and returns true if the puppy is found, false otherwise. If the puppy is NOT found **`searchRoom`** also sets the `myHidingPlaces` array at the received floor and room location to the received player value (a '1' or a '2') OR sets the `myWinner` field to the current player.
- **`toString`** displays the current hidingPlaces array and it's contents EXCEPT the location of the puppy which remains hidden until he/she is found at which point toString will be called (by the driver) and both the player who found the puppy and a 'P' will be displayed in the same cell....

  NOW, and perhaps the awkward portion of the toString output. Normally, when displaying a 2D array, the [0][0] cell is displayed in the upper left corner as with matrices. However, because the puppy decided to get lost in a building and not a matrix, it would make more visual sense to have the first floor (row 0) displayed on the bottom, second floor above it... and finally the top floor, well... on top! To save words, look closely at the sample run provided on the next page. Your output should look the same as what is seen on the next page in the sample run.

(The driver program `PuppyPlay.java`, randomly chooses which player should begin)

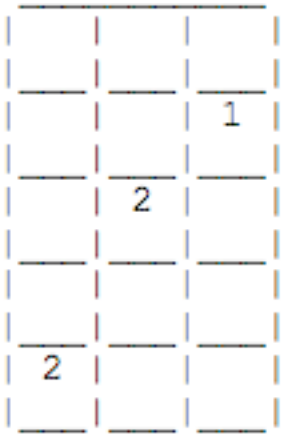
Sample Output:

```
----jGRASP exec: java PuppyPlay
To find the puppy, we need to know:
        How many floors are in the building
        How many rooms are on the floors
Please enter the number of floors: 5

Please enter the number of rooms on the floors: 3

Floor and room numbers start at zero '0'
Player 2, enter floor and room to search separated by a space:
0 0

[0],[0]
```

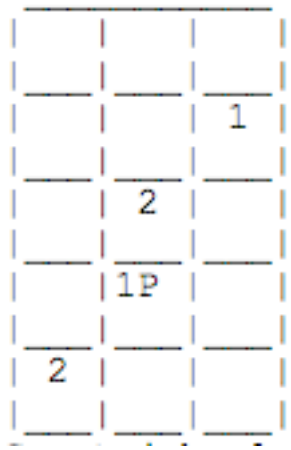

```
Player 1, enter floor and room to search separated by a space:
3 2

[3], [2]
```


```
Player 2, enter floor and room to search separated by a space:
2 1
```

```
[2], [1]
```




```
Player 1, enter floor and room to search separated by a space:

1 1

[1], [1]
```




```
Great job player 1!  Would you like to find another puppy [Y/N]? n
```

How to submit your assignment:

1. Upload your Java files on Canvas.
2. There will be a submission link for Assignment 2.