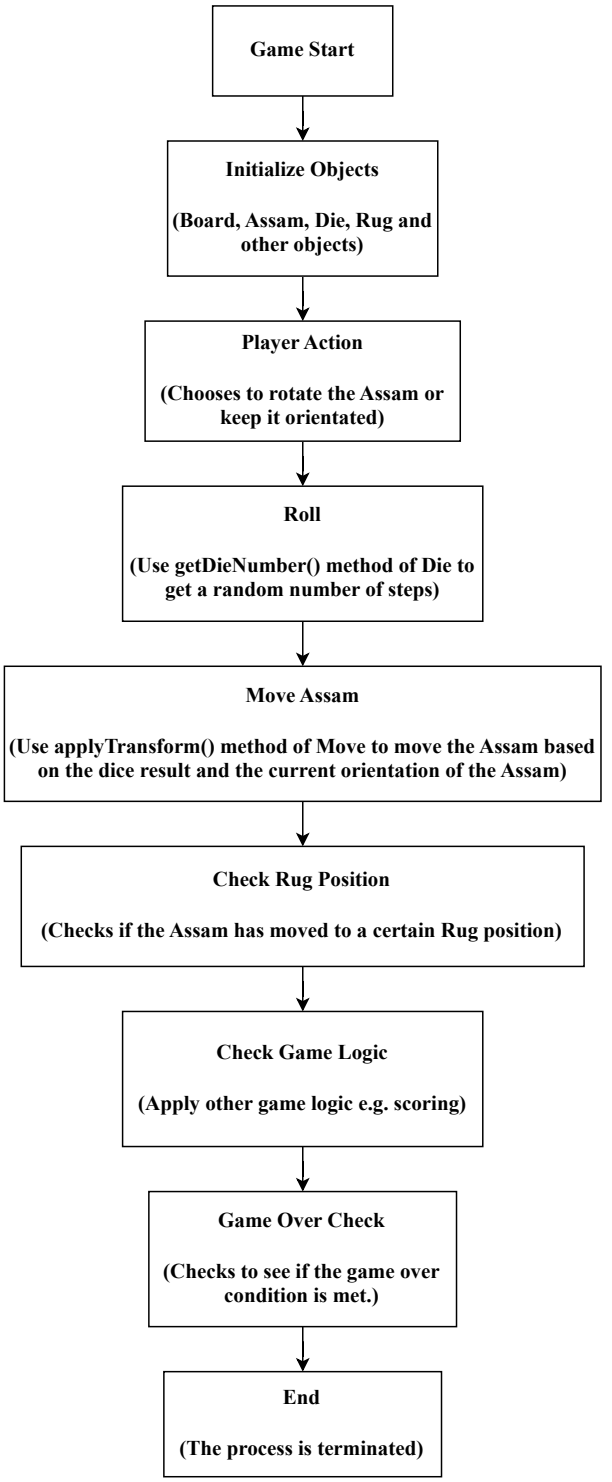
Diagram: Flow Chart



Game Design Idea

1. Game Start

(1). About board

There is a 7 x 7 board, we consider the (0,0) is in the bottom left corner, so:

1). (x, y) stand for the position. ($0 \le x \le 6$, $0 \le y \le 6$)

2). when (x+1), it means guest move to the right side of the Board.

⚠ if the guest face down side and move right,
then it should be (x-1), similar situation for y

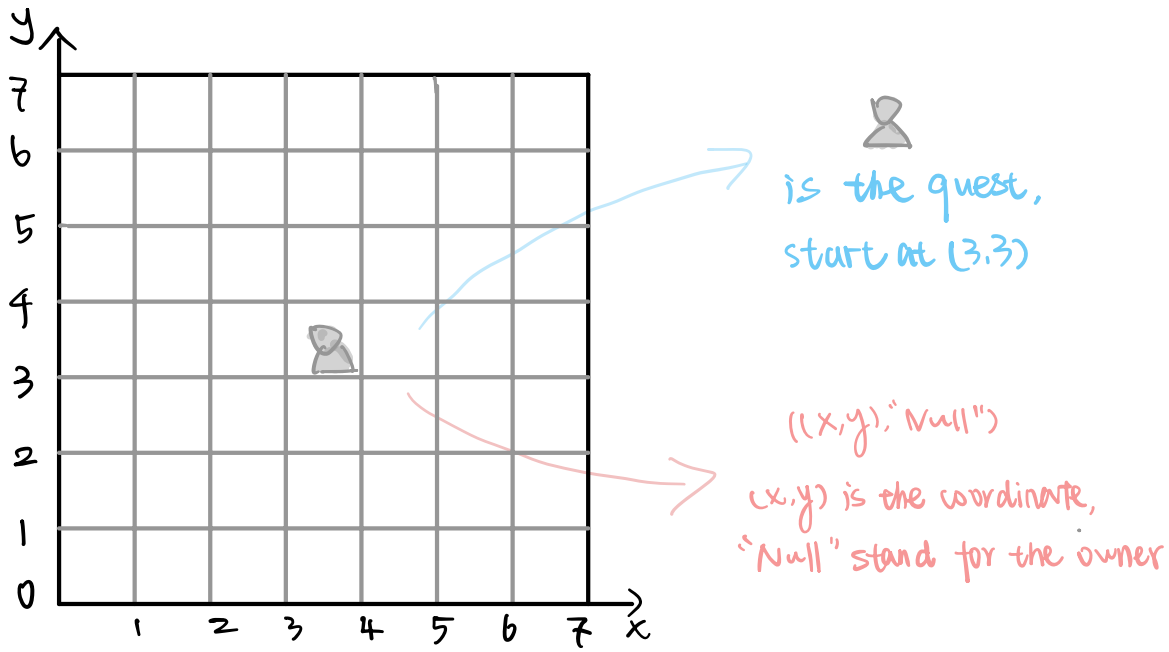
3). when the customer front close beyond the board, then will turn left.
(It's up to the Customer !)

(2). About player

1). Color : Bule, Red, Yellow, Brown

2). Number : [2, 4]

3). Choose the number of players : two will be ① ②, three will be ① ② ③, four will be ① ② ③ ④.



2. Game Process

(1). We have four lists to store every single players' places.

eg list < IntPair > red Position ..... (This is in "Board" class).

(2). Then these lists will be coverted to lists of IntPair.

eg list < list < IntPair >> red Position ..... (This is in "Rug" class).

It's aim to make adjacent rugs into a whole piece of rug.

(3). "Die" class will generate a random number for "Move" class.

(4). "Move" class will take it as an input and apply a transform to "Assam".

(5). Then "Assam" will change his status according to the status his has right now (which is recorded in "Assam" class).

(6). Check Assam's position. If he stand on other's place, check which player own this rug.

(7). Then assame Assam stand on (x, y), we check list < list <IntPair>>, find out which list < IntPair > has (x,y) as an element.

(8). Once we find it out, calculate the length of this list <IntPair>, the result is the money that need to be paid.

3. Game Over

(1). Then we check out if the Money that player own is less than 0, or if the number of rugs left to place is equal to 0.

If one of the above conditions meet, the player exit the game, and the game ends when we know that all the players have met the exit condition.

(2). When the game over, the score will be calculated and result will be output by scoring.