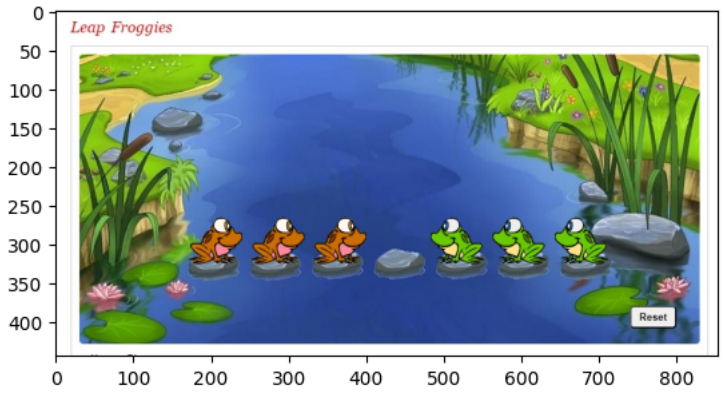
Use this link to play the puzzle. https://www.neok12.com/games/leap-froggies/leap-froggies.htm

```
1 from PIL import Image
2 from matplotlib import pyplot as plt
3 img = Image.open('frog.jpg')
4 plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7927ea71a770>
```



Step 1

First create a list `positions` which contains the characters 'G','B' and '-' in the same sequence as given in the initial display state.

```
1 positions = ["G", "G", "G", "_", "B", "B", "B"]
2 print(positions)
```

```
['G', 'G', 'G', '_', 'B', 'B', 'B']
```

Now print this string `[ 0 , 1 , 2 , 3 , 4 , 5 , 6 ]` and after that print the list `positions`

```
1 print([0,1,2,3,4,5,6])
2 print(positions)
```

```
[0, 1, 2, 3, 4, 5, 6]
['G', 'G', 'G', '_', 'B', 'B', 'B']
```

Take position input from user and write a message as `"Press q to quit else \nEnter position of piece:"`.

```
1 next_move = input("Press q to quit else \nEnter position of piece: ")
2 print(next_move)
```

```
Press q to quit else
Enter position of piece: 2
2
```

Now the taken input is in string format. So first check if the input is `'q'` character. If input is `'q'` then the person is quiting the game so print `'You Lose'`.

```
1 if str(next_move) == "q": print("You Lose")
```

Next if input character is not `'q'` then it has to be some integer. so convert input to integer format.

```
1 if str(next_move) != "q":
2   move = int(next_move)
3   print(move)
```

```
2
```

Step 2

Now we have to check validity of the selected positions or move.
If the entered number isn't between 0 and 6, then print 'Invalid move'.

```
1 if move not in range(0,7): print("Invalid move")
```

A frog should be present on the selected position to make a move. If leaf is selected then it doesn't make sense. Therefore, if entered postition is same as the postition of empty leaf then the move is invalid and print `Invalid Move`

```
1 if move == positions.index("_"): print("Invalid move")
```

Initialize a variable named `pos2` at value 0, to store the index of empty leaf, so that we can use it later.

```
1 pos2 = 0
```

Check if the selected frog is 'G':

(Inside if when it's 'G'. As 'G' is selected frog can move to right only.)

condition 1

If **selected_position + 1** is less than or equal to 6 and **curent_position + 1** contains '-'
then it's a valid move and store that postion in `pos2`.

condition2

Else if **selected_position + 2** is less than or equal to 6 and if **current_position + 2**
contains '-' and if **selected_position + 1** contains 'B' then it's a valid move and store that postion in `pos2`.

condition3:

Else remainig all are invalid, so print `Invalid Move`

```
1 if positions[move] == "G":
2     if move + 1 <= 6 and positions[move + 1] == "_":
3         pos2 = move + 1
4     elif move + 2 <= 6 and positions[move + 2] == "_" and positions[move + 1] == "B":
5         pos2 = move + 2
6     else:
7         print("Invalid move")
```

Check if the selected frog is 'B':

(Inside if when it's 'B'. As 'B' is selected frog can move to left only.)

condition1:

If **selected_position - 1** is more than or equal to 0 and **curent_position - 1** contains '-' then
it's a valid move and and store that postion in `pos2`.

condition2:

Else if **selected_position - 2** is more than or equal to 0 and if **current_position - 2** contains '-'
and if **selected_position - 1** contains 'G' then it's a valid move and and store that postion in `pos2`.

condition3:

Else remainig all are invalid,, so print `Invalid Move`.

```
if positions[move] == "B":
    if move - 1 >= 0 and positions[move - 1] == "_":
        pos2 = move - 1
    elif move - 2 >= 0 and positions[move - 2] == "_" and positions[move - 1] == "G":
        pos2 = move - 2
    else:
        print("Invalid move")
```

Swap the element at selected positions and calculated position2 in the list.
So basically we are moving the frog to next valid position by swapping elelments of array.

```
positions[move], positions[pos2] = positions[pos2], positions[move]
```

Now print the display of the game again to see the change.
If we enter position 2 then the output will look like this:-

```
[ 0 ,  1 ,  2 ,  3 ,  4 ,  5 ,  6 ]
['G', 'G', '-', 'G', 'B', 'B', 'B']
```

```
print(positions)
```

```
['G', 'G', '_', 'G', 'B', 'B', 'B']
```

Check for winning condition by comparing the elements of list. If player has won the game print `'You Win'`

```
if positions == ["B","B","B","_","G","G","G"]: print("You Win")
```

Now the game should keep running until the player quits, so place all conditional statements inside an infinite loop.

1. We have to `'break'` the loop if the player presses `'q'` and quits.
2. If the move made by player is `'Invalid Move'` then we have to `'continue'` without executing remaining part of the selected iteration.
3. If player wins the game we have to `break` the loop.

```
Infinite loop:
    (inside loop)
    1.Take input
    2.Check all valid and invalid conditions of `pos`.
    3.Make the appropriate move by calculating `pos2`.
    4.Display game
    4.Check winning condition
```

So I have create this game and I have done some twicks in it too. Lets see if you can find them.

```
positions = ['G', 'G', 'G', '_', 'B', 'B', 'B']
difficulty_lvl = input("Choose difficulty level from Easy, Medium, Hard: ")
i = 0
i_max = i
if str.lower(difficulty_lvl) == "easy":
    i_max = 25
elif str.lower(difficulty_lvl) == "medium":
    i_max = 20
elif str.lower(difficulty_lvl) == "hard":
    i_max = 16
while i < i_max:  # Adding a condition to limit the attempts
    next_move = input("Press q to quit else \nEnter position of piece: ")
    if str.lower(next_move) == "q":
        print("You Lose")
        break
    if str(next_move) != "q":
        move = int(next_move)
    if move not in range(0,7):
        print("Invalid move")
        i += 1
```

```
        continue
    if move == positions.index("_"):
        print("Invalid move")
        continue
    pos2 = 0
    if positions[move] == "G":
        if move + 1 <= 6 and positions[move + 1] == "_":
            pos2 = move + 1
        elif move + 2 <= 6 and positions[move + 2] == "_" and positions[move + 1] == "B":
            pos2 = move + 2
        else:
            print("Invalid move")
            continue
    if positions[move] == "B":
        if move - 1 >= 0 and positions[move - 1] == "_":
            pos2 = move - 1
        elif move - 2 >= 0 and positions[move - 2] == "_" and positions[move - 1] == "G":
            pos2 = move - 2
        else:
            print("Invalid move")
            continue

    positions[move], positions[pos2] = positions[pos2], positions[move]
    print(positions)
    if i == i_max:
        print("You Lose, Crossed the max number of moves")
        break
    if positions == ["G","G","G","B","B","B","_"]:
        print("You Lose")
        break
    if positions == ["_","G","G","G","B","B","B"]:
        print("You Lose")
        break
    if positions == ["B","B","B","_","G","G","G"]:
        print("You Win")
        break
```

```
Choose difficulty level from Easy, Medium, Hard: HARD
Press q to quit else
Enter position of piece: 4
['G', 'G', 'G', 'B', '_', 'B', 'B']
Press q to quit else
Enter position of piece: 2
['G', 'G', '_', 'B', 'G', 'B', 'B']
Press q to quit else
Enter position of piece: 1
['G', '_', 'G', 'B', 'G', 'B', 'B']
Press q to quit else
Enter position of piece: 3
['G', 'B', 'G', '_', 'G', 'B', 'B']
Press q to quit else
Enter position of piece: 5
['G', 'B', 'G', 'B', 'G', '_', 'B']
Press q to quit else
Enter position of piece: 6
['G', 'B', 'G', 'B', 'G', 'B', '_']
Press q to quit else
Enter position of piece: q
You Lose
```

Thank You!!