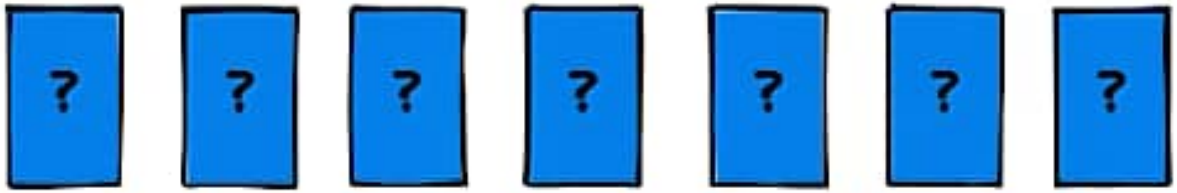
QUESTION 1: Alice has some cards with numbers written on them. She arranges the cards in decreasing order, and lays them out face down in a sequence on a table. She challenges Bob to pick out the card containing a given number by turning over as few cards as possible. Write a function to help Bob locate the card.



```
ef card_game(cards, query):

    # Create a variable position with the value 0
    position = 0

    #print cards and query
    print("cards : " , cards)
    print("query : ", query)

    # Set up a loop for repetition
    while position < len(cards):

        #print the position
        print("position : ", position)

        # Check if element at the current position matche the query
        if cards[position] == query:
            # Answer found!
            print("\nAnwser is ", position)
            return position

         # If its not the position then increment the position
        position += 1

        # Check if we have reached the end of the array
        if position == len(cards):
            # Number not found, return -1
            return -1
    return -1           You, 1 second ago • Uncommitted changes

funtion calling
ard_game([13, 11, 10, 7, 4, 3, 2, 1, 0], 7)
```

```
cards :  [13, 11, 10, 7, 4, 3, 2, 1, 0]
query :  7
position :  0
position :  1
position :  2
position :  3

Anwser is  3
```

```
{'input': {'cards': [13, 11, 10, 7, 4, 3, 1, 0], 'query': 1}, 'output': 6},
{'input': {'cards': [4, 2, 1, -1], 'query': 4}, 'output': 0},
{'input': {'cards': [3, -1, -9, -127], 'query': -127}, 'output': 3},
{'input': {'cards': [6], 'query': 6}, 'output': 0},
{'input': {'cards': [9, 7, 5, 2, -9], 'query': 4}, 'output': -1},
{'input': {'cards': [], 'query': 7}, 'output': -1},
{'input': {'cards': [8, 8, 6, 6, 6, 6, 6, 3, 2, 2, 2, 0, 0, 0], 'query': 3},
 'output': 7},
{'input': {'cards': [8, 8, 6, 6, 6, 6, 6, 6, 3, 2, 2, 2, 0, 0, 0],
  'query': 6},
 'output': 2}]
```

Data Structure and algorithm

Binary Search, Linked list and Complexity Analysis

Introduction to Binary Search [Including LL] and complexity Analysis with Python

- why should learn DSA

1. you can think about a problem systematically and solve it systematically step by step
2. you envision different input, output, and edge case for programs you write
3. you can communicate your ideas clearly to co-workers and incorporate their suggestions
4. Mostly importantly, you can convert your thoughts and ideas into working codes that is also reusable

The Method

- Systematic strategy for solving problems

1. State the ~~prog~~ problem clearly. Identify the input and output ~~format~~
2. Come up with some example input and output. Try to cover all edges
3. Come up with correct solution for the problem. State it in plain English
4. Implement the solution ~~for the problem~~ and test it with example input. Fix bugs, if any
5. Analyze the algorithm's complexity and identify inefficiencies, if any
6. Apply code techniques to overcome the inefficiency. Repeat 3 to 6

Q => on DSA

These are the cards (list of cards), which is need to find a specific card with any number. So we are going to take a computer based example.

[7, 10, 13, 4, 1, 3, 11, 0]

These are the list of numbers. Then we need to find a number which I am considering (7)

And arrange it in decreasing order

Sorted list → [13, 11, 10, 7, 4, 3, 1, 0]

0 1 2 3 4 5 6 7

Queried number

Position

Answer

Main programme based solution using Python

- Arrange the list in decrease order

Input

1. cards : (Arrange in decrease order)

[13, 11, 10, 7, 4, 3, 1, 0]

2. query : (Number to be find) 7

Output

1. Position of query 7 => (3)

Now we determine the input and output

Step :-

```
def card_game(cards, query):
    pass
```

pass => Function doesn't work with empty so for we just pass with pass

pass

In the test case (There sover test),

1. The number query occur somewhere in the middle of the list card
2. query is the first element in the card.
3. query is the last element in the card
4. The list cards contains just one element, which is query
5. The list cards does not contain number query
6. The list card is empty.
7. The list cards contain repeating number
8. The number query occur at more than one position in cards

(tag can you the any more variation)

There is techinques to find the position of the query. which is brutefore method also called linear Search

which is iterate through the every number in the list and return when it find. It search number in a linear fasion.

Programme [Python]

```
def card_game(cards, query):
    position = 0
    while position < len(cards):
        if cards[position] == query:
            print(position)
            return position

        position += 1
        if position == len(cards):
            return -1

    return -1

card_game([13, 11, 10, 7, 4, 3, 2, 0], 7)

# Output => 3
```