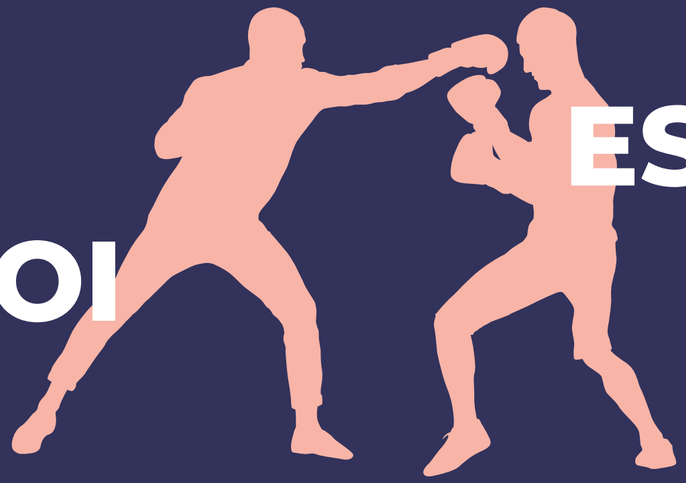
LEETCODATAAAAA
DEL 18/10/2023

NOI



ESERCIZI

Bacaro
Tech

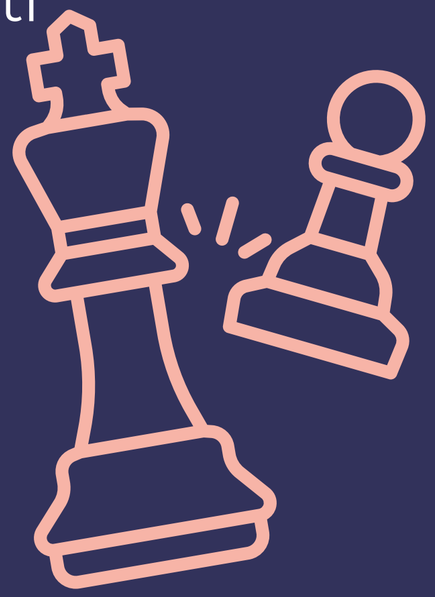
CODE AND FUN

Bacaro
Tech

CODE AND FUN

ESERCIZIO 1
AVAILABLE CAPTURES FOR ROOK

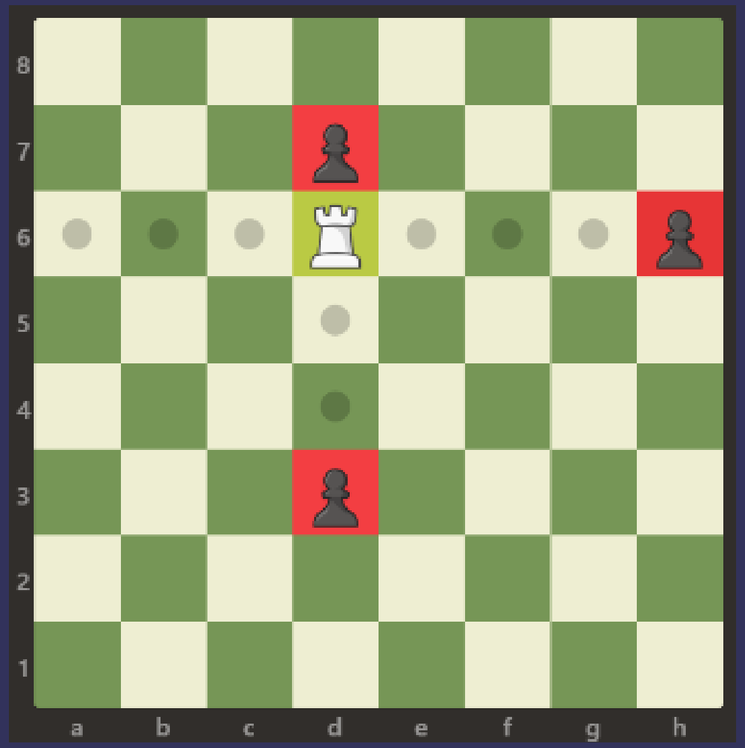
Link esercizio: https://leetcode.com/problems/available-captures-for-rook/description/
Difficoltà: easy
Tempo standard: 45 min
Tempo di recupero: 10/15 minuti
Area di interesse: Array



Bacaro
Tech

CODE AND FUN

CAPIAMO IL PROBLEMA
AVAILABLE CAPTURES FOR ROOK



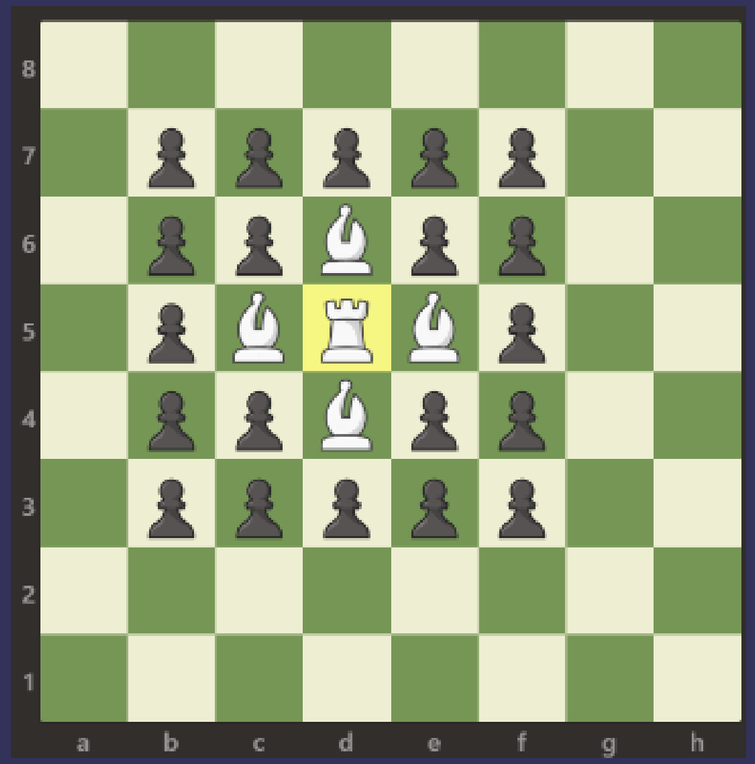
In questo caso l’algoritmo deve restituire come output 3, in quanto la torre bianca ha 3 pezzi neri “sotto attacco”, mangiabili con una sola mossa.

OUTPUT: 3

Bacaro
Tech

CODE AND FUN

CAPIAMO IL PROBLEMA
AVAILABLE CAPTURES FOR ROOK



In questo caso l’algoritmo deve restituire come output 0, in quanto DIRETTAMENTE la torre bianca non ha nessun pezzo nero “sotto attacco”, mangiabile con una sola mossa.

OUTPUT: 0

IT'S TIME TO CODING!

Bacaro
Tech

CODE AND FUN

AVAILABLE CAPTURES FOR ROOK
QUALI SONO LE DOMANDE?

Dove sta la torre bianca? e come distinguerla sulla matrice?

Come si fa a capire se un pezzo è “sotto attacco”? E come si fa determinare se la linea di tiro è pulita, ovvero non sono pezzi bianchi in mezzo alla traettoria?

Come si può scrivere il codice più efficente possibile in termini di spazio e tempo?

Bacaro Tech

CODE AND FUN

AVAILABLE CAPTURES FOR ROOK
SOLUZIONE PROPOSTA

```java
public int numRookCaptures(char[][] board){
    //ricerca della torre
    int x=-1, y=-1;
    for(int i=0; i<8; i++){
        for(int j=0; j<8; j++){
            if(board[i][j]=='R'){
                x = i;
                y = j;
                break;
            }
        }
    }

    //conta degli elementi
    int s = 0;
```

Ricerca della posizione contenente la torre bianca.

Sono costretto a scorrere tutta la matrice, in quanto potrebbe essere in qualsiasi posizione!

Bacaro Tech

CODE AND FUN

AVAILABLE CAPTURES FOR ROOK
SOLUZIONE PROPOSTA

```
//conta degli elementi neri a dx della torre in poi
for(int i=x; i<8; i++){
    if(board[i][y]=='p'){
        s++;
        break;
    }
    if(board[i][y]=='B') break;
}

//conta degli elementi neri a sx della torre in poi
for(int i=x; i>=0; i--){
    if(board[i][y]=='p'){
        s++;
        break;
    }
    if(board[i][y]=='B') break;
}
```

Ricerca degli elementi che sono in traiettoria rispetto alla posizione della torre.

Dividiamo questo task vedendo prima gli elementi che si trovano sull’asse x e poi sull’asse y nella prossima slide.

Bacaro
Tech

CODE AND FUN

AVAILABLE CAPTURES FOR ROOK
SOLUZIONE PROPOSTA

```
//conta degli elementi neri al di sotto della torre in poi
for(int j=y; j<8; j++){
    if(board[x][j]=='p'){
        s++;
        break;
    }
    if(board[x][j]=='B') break;
}

//conta degli elementi neri al di sopra della torre in poi
for(int j=y; j>=0; j--){
    if(board[x][j]=='p'){
        s++;
        break;
    }
    if(board[x][j]=='B') break;
}

//si ritorna la somma
return s;
```

Ricerca degli elementi che sono in traiettoria rispetto alla posizione della torre.

In questo momento ricerchiamo tutti gli elmenti che si trovano sull’asse y.

Bacaro
Tech

CODE AND FUN

ESERCIZIO 2
SORT ARRAY BY INCREASING FREQUENCY

Link esercizio: https://leetcode.com/problems/sort-array-by-increasing-frequency/description/
Difficoltà: easy
Tempo standard: 45 min
Tempo di recupero: 10/15 minuti
Area di interesse: Array, Ordinamenti

Bacaro
Tech

CODE AND FUN

CAPIAMO IL PROBLEMA
SORT ARRAY BY INCREASING FREQUENCY

```
Input: nums = [1,1,2,2,2,3]
Output: [3,1,1,2,2,2]
```

L’algoritmo deve mettere in ordine l’array che riceve in input in base alla frequenza degli elementi.

IT'S TIME TO CODING!

Bacaro
Tech

CODE AND FUN

SORT ARRAY BY INCREASING FREQUENCY

QUALI SONO LE DOMANDE?

Come si può tener traccia delle diverse frequnze?

Come è possibile ordinare prima le frequenze degli elementi e successivamente ordinare gli elementi stessi all’interno dell’array?

Come si può scrivere il codice più efficente possibile in termini di spazio e tempo?

Bacaro
Tech

CODE AND FUN

SORT ARRAY BY INCREASING FREQUENCY
SOLUZIONE PROPOSTA

```java
public int[] frequencySort(int[] nums) {

    //mappa per identificare la frequenza degli elementi
    Map<Integer, Integer> tempList = new TreeMap<>();

    //mappa che indica per un certo indice quanti sono gli elementi associati
    //nell'esempio delle slide
    //1 -> [1,1]
    //2 -> [2,2,2]
    //3 -> [3]
    Map<Integer, ArrayList<Integer>> countList = new TreeMap<>();
```

Setup delle mappe per organizzare i dati.

La prima verrà usata per le frequenze degli elementi, mentre la seconda per i sotto array con i diversi elementi.

Bacaro
Tech

CODE AND FUN

SORT ARRAY BY INCREASING FREQUENCY
SOLUZIONE PROPOSTA

```
//aggiornamento della mappa delle frequanze
for(int a: nums){
    if(tempList.containsKey(a))
     tempList.put(a, tempList.get(a)+1);
    else
     tempList.put(a,1);
}

//aggiornamento della mappa del inserimento degli array di elementi
for(Map.Entry<Integer,Integer> entry: tempList.entrySet()){
     int val = entry.getValue(); int key = entry.getKey();
     if(!countList.containsKey(val))
         countList.put(val, new ArrayList<Integer>());
     countList.get(val).add(key);
}
```

Questi 2 cicli for servono per popolare le strutture dati, sia che si vada in insert di una nuova chiava, e sia che si vada in update di una chiave già presente nella mappa.

Bacaro
Tech

CODE AND FUN

SORT ARRAY BY INCREASING FREQUENCY
SOLUZIONE PROPOSTA

```
//businness logic di ordinamento
int index = 0;
for(Map.Entry<Integer,ArrayList<Integer>> entry: countList.entrySet()){
    int val = entry.getKey(); ArrayList<Integer> array = entry.getValue();
    Collections.sort(array);// Sort each Array
    for(int a = array.size()-1; a>-1; a--){
        int value = array.get(a);
        for(int b = 0; b < val; b++){
            nums[index] = value;
            index++;
        }
    }
}
return nums;
```

Vera busiennes logic dell’algoritmo.

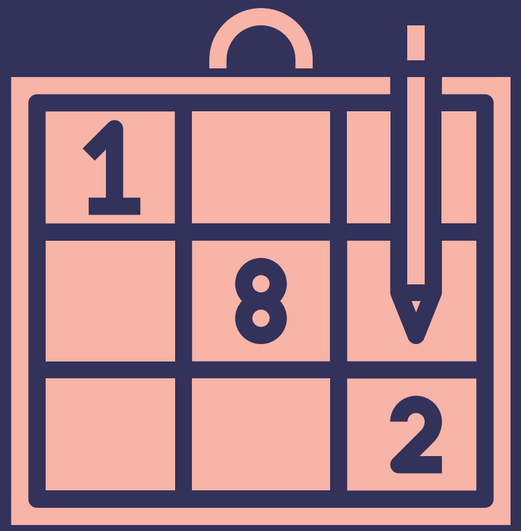
Viene messo in ordine l’array delle frequenze e grazie alle chiavi in ordine si concatenano i diversi sottoarray contenuti nell’altra mappa.

Bacaro
Tech

CODE AND FUN

ESERCIZIO 3
VALID SUDOKU

Link esercizio: https://leetcode.com/problems/valid-sudoku/description/
Difficoltà: medio
Tempo standard: 60 min
Tempo di recupero: 10/15 minuti
Area di interesse: Matrici



Bacaro
Tech

CODE AND FUN

CAPIAMO IL PROBLEMA
VALID SUDOKU

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

In questo caso l'algoritmo deve restituire come output true, in quanto ogni riga e colona rispettano le proprietà del sudoku.

OUTPUT: TRUE

IT'S TIME TO CODING!

Bacaro
Tech

CODE AND FUN

VALID SUDOKU
QUALI SONO LE DOMANDE?

Come si può controllare che sulle righe e sulle colonne non ci siano ripetizioni?

Come si possono controllare i sottoquadrati di 3x3?

Capito come eseguire questi controlli, come si possono memorizzare queste informazioni per ottenere il massimo dalla struttura dati “ospitante”?

Come si può scrivere il codice più efficente possibile in termini di spazio e tempo?

Bacaro
Tech

CODE AND FUN

VALID SUDOKU
SOLUZIONE PROPOSTA

```
Set seen = new HashSet();
for (int i=0; i<9; ++i) {
    for (int j=0; j<9; ++j) {
        char number = board[i][j];
        if (number != '.')
            if (!seen.add(number + " in row " + i) ||
                !seen.add(number + " in column " + j) ||
                !seen.add(number + " in block " + i/3 + "-" + j/3))
                return false;
    }
}
return true;
```

L’algoritmo sfrutta un hashset nel quale verranno memorizzate delle chiavi composte da più informazioni.

In questo modo si può scansionare l’intera matrice sfruttando al massimo la mappa sottostante!

Bacaro
Tech

CODE AND FUN

VI RINGRAZIA TUTTI PER AVER PARTECIPATO!