Round 1

1. Which Data Structure is mainly used for implementing the recursive algorithm?
a) Queue
b) Stack
c) Linked List
d) Tree
Answer: b

2. What's happen if base condition is not defined in recursion?
a) Stack underflow
b) Stack Overflow
c) None of these
d) Both a and b
Answer: b

3. Choose the correct answer.
a) Recursion is always better than iteration.
b) Recursion uses more memory compared to iteration.
c) Recursion uses less memory compared to iteration.
d) Iterative function is always better and simpler to write than recursion.
Answer: b

4. Recursion is similar to which of the following?
a) Switch Case
b) Loop
c) If-else
d) None
Answer: b

5. Recursion uses more memory compared to iteration.
a) True
b) False
c) Both
d) None of the above
Answer: a

6. Which of the following problem cannot be solved using recursion?
a) Tower of Hanoi
b) Fibonacci series
c) Tree Traversal
d) Problems without base case
Answer: d

7. Which searching can be performed recursively?
a) Linear search
b) Binary search
c) Both
d) None
Answer: c

8. In recursion the condition after which the function will stop calling itself is
a) Base condition
b) Function call
c) Both
d) None
Answer: a

9. What is the average case time complexity of binary search using recursion?
a) O(nlogn)
b) O(logn)
c) O(n)
d) O(n^2)
Answer: b

10. What are the applications of binary search?
a) To find the lower/upper bound in an ordered sequence
b) Union of intervals
c) Debugging
d) All of the mentioned
Answer: d

11. Binary Search can be categorized into which of the following?
a) Brute force technique
b) Divide and conquer
c) Greedy algorithm
d) Dynamic programming
Answer: b

12. Binary Search works when the array is ____________.
a) random
b) sequential
c) sorted
d) any of these
Answer: d

13.The maximum comparisons needed in Binary Search on array of size 32 is _________.
a) 16

b) 4
c) 5
d) 8
Answer: c

14. Binary search terminates ....
a) when (starting index = = ending index)
b) when (starting index < ending index)
c) when (starting index > ending index)
d) None of these
Answer: a

15. If the middle element is greater than the target value then binary search searches the target element in the second half of the array? True | False
a) False
b) None
c) True
d) Both a and c
Answer: a

16. Which of the following is not a limitation of the binary search algorithm?
a) must use a sorted array
b) requirement of the sorted array is expensive when a lot of insertions and deletions are needed
c) there must be a mechanism to access the middle element directly
d) binary search algorithm is not efficient when the data elements are more than 1000.
Answer: d

17. If the number of records to be sorted is small, then ...... sorting can be efficient.
a) Merge
b) Heap
c) Selection
d) Bubble
Answer: c

18. The complexity of the sorting algorithm measures the ...... as a function of the number n of items to be sorter.
a) average time
b) running time
c) average-case complexity
d) case-complexity
Answer: b

19. Binary search algorithm cannot be applied to ...
a) sorted linked list

b) sorted binary trees
c) sorted linear array
d) pointer array
Answer: a

20. Sorting algorithm can be characterized as ......
A. Simple algorithm which requires the order of n2 comparisons to sort n items.
B. Sophisticated algorithms that require the O(nlog2n) comparisons to sort items.
C. Both of the above
D. None of the above
Answer: c

21. State True or False for internal sorting algorithms.
i) Internal sorting are applied when the entire collection if data to be sorted is small enough that the sorting can take place within main memory.
ii) The time required to read or write is considered to be significant in evaluating the performance of internal sorting.
A. i-True, ii-True
B. i-True, ii-False
C. i-False, ii-True
D. i-False, ii-False
Answer: b

22. Which of the following sorting algorithm is of divide and conquer type?
A. Bubble sort
B. Insertion sort
C. Merge sort
D. Selection sort
Answer: c

23. Which of the following sorting algorithm is of priority queue sorting type?
A. Bubble sort
B. Insertion sort
C. Merge sort
D. Selection sort
Answer: d

24. Partition and exchange sort is ....
A. quick sort
B. tree sort
C. heap sort
D. bubble sort
Answer: a

25. Which of the following is an internal sorting?

A. Tape Sort
B. 2-way Merge Sort
C. Merge Sort
D. Tree Sort
Answer: d

26. The operation that combines the element is of A and B in a single sorted list C with n=r+s element is called ....

A. Inserting
B. Mixing
C. Merging
D. Sharing
Answer: c

27. Merging k sorted tables into a single sorted table is called ......

A. k way merging
B. k th merge
C. k+1 merge
D. k-1 merge
Answer: a

28. What is the type of null data type?

A. string
B. null
C. number
D. object
Answer: d

29. If the number of records is to be sorted large and the key is long, then ...... sorting can be efficient.

A. Merge
B. Heap
C. Quick
D. Bubble
Answer: c

30. Given an array arr = {5, 6, 7, 8, 9} and key = 8; How many iterations are done until the element is found?

A. 1
B. 3
C. 4
D. 2
Answer: d

Round 2

Problem Statement1

You are given two strings X and Y of the same length. Each string contains N Lower case character (from 'a' to 'z'). A shift operation will remove the first character of a string and add the same character at the end of that string. For example, after you perform a shift operation on a string 'abcd', the new string will be 'bcda'. If you perform this operation two times, the new string will be 'cdab'. You need to use some (maybe none) shift operations on the string Y to maximize the length of the longest common prefix of X and Y. If more than one result can be found, pick the one that uses the smallest number of shift operations.

Input

- The first line of the input contains a single integer N. The second and the third line contains the string X and Y respectively.

Output

- Contains a single integer which is the number of shift operations.

Constraints

30 points:

- $1 \le N \le 5000$

30 points:

- $1 \le N \le 10^4$

40 points:

- $1 \le N \le 10^6$

Sample Input:

```
5
ccadd
bddcc
```

Sample Output:

```
3
```

Js Template:

```
function main() {
		let n = parseInt(readline());
		let a = readline();
		let b = readline();
	// write your logic here

}
```

Solution :-

https://ideone.com/OwQDLN

Problem Statement2:

You are given a score log of a football match between two teams. Every time when one of the teams scored a goal, the name of that team was written in the score log on a separate line.
At the end of the match, the team that scored strictly more goals than the other team wins the match. If both teams score an equal number of goals, the match ends in a tie. Determine the name of the winning team or that the match ended in a tie.

Input

- The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows.
- The first line of each test case contains a single integer n denoting the number of records in the score log (the total number of goals scored during the match).
- Each of the following n lines contains a string s denoting the name of the team that scored a goal.

Output

- For each test case, print a single line containing one string — the name of the winning team or "Draw" (without quotes) in case of a tie.

Constraints

- $1 \le T \le 10^5$
- $0 \le n \le 10^5$
- $1 \le |s| \le 20$
- s consists only of lowercase English letters
- sum of n over all test cases $\le 10^5$
- for each test case, there are at most two different team names

Sample Input 1

```
2
4
ab
bc
bc
ab
3
xxx
yyy
yyy
```

Sample Output 1

```
Draw
  yyy
```

JS Template:

```
function main() {
        let t = parseInt(readline());
```

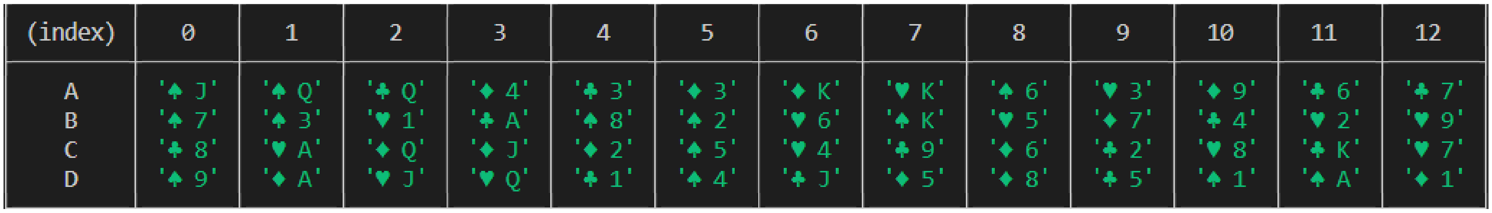
```
   while(t--)
           {
              let n = parseInt(readline());
              let s = [];

               for(let i=0;i<n;i++)
               {
                   s[i] = readline();
               }
// write your logic here
           }
   }
```

Solution:-

https://ideone.com/foTL5t

Round 3

Create a playing card shuffler where it should shuffle the 52 deck card to the given number of persons
Below image i have given 4 persons A, B, C, D



(index)	0	1	2	3	4	5	6	7	8	9	10	11	12
A	'♠ J'	'♠ Q'	'♣ Q'	'♦ 4'	'♣ 3'	'♦ 3'	'♦ K'	'♥ K'	'♠ 6'	'♥ 3'	'♦ 9'	'♣ 6'	'♣ 7'
B	'♠ 7'	'♠ 3'	'♥ 1'	'♣ A'	'♠ 8'	'♠ 2'	'♥ 6'	'♠ K'	'♥ 5'	'♦ 7'	'♣ 4'	'♥ 2'	'♥ 9'
C	'♣ 8'	'♥ A'	'♦ Q'	'♦ J'	'♦ 2'	'♠ 5'	'♥ 4'	'♣ 9'	'♦ 6'	'♣ 2'	'♥ 8'	'♣ K'	'♥ 7'
D	'♠ 9'	'♦ A'	'♥ J'	'♥ Q'	'♣ 1'	'♠ 4'	'♣ J'	'♦ 5'	'♦ 8'	'♣ 5'	'♠ 1'	'♠ A'	'♦ 1'

Solution :-
https://jsfiddle.net/saravanansIb/64gk3jc1/16/

Solution:

```
const cardType = {
  club: '♣',
  diamonds: '♦',
  hearts: '♥',
  spades: '♠'
}

const cards = ['K', 'Q', 'J', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A'];

const shuffle = (person) => {
    const user = {};
    person.forEach(item => user[item] = []);
  let allCards = Object.values(cardType).map(item =>
    cards.map(_card => `${item} ${_card}`)
  ).flat();
  while (allCards.length) {
    person.forEach(item => {
        let selectedCard = allCards[Math.floor(Math.random() * allCards.length)];
        if (selectedCard) user[item].push(selectedCard);
        allCards = allCards.filter(item => item !== selectedCard);

    });
  }
  console.table(user)

}
shuffle(['A', 'B', 'C', 'D', 'E'])
```