MCQs

1. **LinkedList is which type of data structure? (Easy) (crs-be-programming)**
    a. Static
    b. Dynamic (correct)
    c. Both a and b
    d. None of the above

2. **Which of the following is true about linear and binary search? (Medium) (crs-be-programming)**
    a. Linear search iterates over all the elements in linear sequence while binary search follows divide and conquer paradigm to optimised the search operation
    b. Binary search can only be applied on sorted arrays, while linear search dont have any such restrictions
    c. If target element is not present in binary search, it would lead to infinite loop
    d. Only a and b (correct)

3. **Which of the following information are stored in Doubly Linked List?(Medium)(crs-be-programming)**
    a. Value of node
    b. Address of next node
    c. Address of the previous node
    d. All of the above (correct)

4. **In a circular linked list (Medium)(crs-be-programming)**
    a. Components are all linked together in some sequential manner.
    b. There is no beginning and no end (correct)
    c. Components are arranged hierarchically.
    d. Forward and backward traversal within the list is permitted.

5. **Suppose we want to search for an element in an unsorted array of size 1000, which algorithm would be optimal to use? (easy) (crs-be-programming)**
    a. Linear search (correct)
    b. Binary search
    c. Either of the above will give the same performance
    d. None of these

6. **What is the complexity of searching an element using Binary Search? (Easy) (crs-be-programming)**
    a. $O(n^2)$
    b. O(1)
    c. O(log n) (correct)
    d. O(n)

7. **In circular linked list, insertion of node requires modification of? (Medium)(crs-be-programming)**
    a. One pointer

b. Two pointer (correct)
c. Three pointer
d. None

8. **In doubly linked list, insertion of node requires modification of? (Medium)(crs-be-programming)**
   a. One pointer
   b. Two pointer (correct)
   c. Three pointer
   d. None

9. **Suppose you are given a linkedlist with head node, what will be the time complexity of searching an element in that Linkedlist? (easy) (crs-be-programming)**
   a. O(1)
   b. O(logn)
   c. O(n) (correct)
   d. $O(n^2)$

10. **Consider an implementation of an circular doubly linked list. Suppose it has its representation with a head pointer only. Given the representation, which of the following operation can be implemented in O(1) time? (Difficult) (crs-be-programming)**
    i) Insertion at the front of the linked list
    ii) insertion at the end of the linked list
    iii) Deletion of the front node of the linked list
    iv) Deletion of the end node of the linked list
    a. a) I and II
    b. b) I and III
    c. c) I, II and III
    d. d) I,II,III and IV (correct)

11. **In linked list each node contain minimum of two fields. One field is data/value field to store the data second field is? (easy) (crs-be-programming)**
    a. Pointer to character
    b. Pointer to integer
    c. Pointer to node (correct)
    d. Node

12. **What is tail pointer in LinkedList? (easy) (crs-be-programming)**
    a. The pointer to last node (correct)
    b. The pointer to middle node
    c. The pointer to first node
    d. None of the above

13. **Which of the following points is/are true about Singly Linked List data structure when it is compared with array?(Medium) (crs-be-programming)**

a. Random access is faster in LinkedList as compared to Arrays.
b. Insertion at the tail can be done in O(1)
c. The size of the array has to be pre-decided, linked lists can change their size any time. (correct)
d. All of these

14. Which of the following is the correct expression to find mid index in binary sarch algorithm? (Medium) (crs-be-programming)

a. (start + end)/2
b. start/2 + end/2
c. (end-start)/2 + start (correct)
d. All of the above

15. Suppose you are given a linkedlist with head and tail nodes, what will be the time complexity of deleting a node from the end? (easy) (crs-be-programming)

a. O(1) (correct)
b. O(logn)
c. O(n)
d. O(n^2)

16. What is the output of the following function for `node` pointing to the first node of the following linked list? 1->2->3->4->5->6 (Difficult) (crs-be-programming)

```
function fun(node)
{
  console.log(node.data);

  if(node.next != NULL )
    fun(node.next.next);
  console.log(node.data);
}
```

a. 1 4 6 6 4 1
b. 1 3 5 1 3 5
c. Runtime error (correct)
d. 1 3 5 5 3 1

17. Given an array arr = {5,6,77,88,99} and key = 88; How many iterations are done until the element is found using Binary search? (Medium) (crs-be-programming)

a. 1
b. 3
c. 4
d. 2 (correct)

18. What operation is the following code performing? Choose the most appropriate answer.(Difficult) (crs-be-programming)

```
Function xyz()
{
        if(head == null)
                return Number.MIN_VALUE;
        let value;
        Node temp = head;
        while(temp.next != head)
                temp = temp.next;
        if(temp == head)
        {
                value = head.value;
                head = null;
                return value;
        }
        temp.setNext(head.next);
        value = head.value;
        head = head.next;
        return value;
}
```

a. Return data from the end of the list
b. Returns the data and deletes the node at the end of the list
c. Returns the data from the beginning of the list
d. Returns the data and deletes the node from the beginning of the list (correct)

19. What type of value can be stored in Linkedlist? (Difficult) (crs-be-programming)

a. Integer
b. String
c. Boolean
d. Any type of data (correct)

20. Which of the following are properties of LinkedList? (easy) (crs-be-programming)

a. Elements in Linked list are not stored at contiguous memory locations. i.e. they are stored at different locations in the memory.
b. Successive elements are connected by link or pointers.
c. Grows and shrinks in size during program execution and allocates memory as the list grows.
d. All of the above (correct)

21. Which of the following LinkedList stores previous and next node addresses without forming circle? (easy) (crs-be-programming)

a. Singly LinkedList
b. Doubly LinkedList (correct)
c. Circular LinkedList
d. None of the above

22. Which of the following is not a type of LinkedList? (easy) (crs-be-programming)

a. Singly LinkedList
b. Doubly LinkedList
c. Circular LinkedList
d. Rectangular LinkedList (correct)

23. What does the following function do for a given Linked List with the first node as *head*? (Medium) (crs-be-programming)

```
void fun1(struct node* head)
{
  if(head == NULL)
    return;

  fun1(head->next);
  printf("%d  ", head->data);
}
```

a. Prints all nodes of linked lists
b. Prints all nodes of linked list in reverse order (correct)
c. Prints alternate nodes of Linked List
d. Prints alternate nodes in reverse order

24. What is the output of the following function for `node` pointing to the first node of the following linked list? 1->2->3->4->5->6 (Difficult) (crs-be-programming)

```
function fun(node)
{
  if(node == null)
    return;
  console.log(node.data);

  if(node.next != NULL )
    fun(node.next.next);
  console.log(node.data);
}
```

a. 1 4 6 6 4 1
b. 1 3 5 1 3 5
c. 1 2 3 5
d. 1 3 5 5 3 1 (correct)

25. If you are given a sorted array and you want to perform a search operation on it, which searching method will you use and why? (easy) (crs-be-programming)

a. Binary Search, search time complexity is O(n)
b. Binary Search, search time complexity is O(logn) (correct)
c. Linear Search, search time complexity is O(n)
d. Binary Search, search time complexity is O(nlogn)

26. Which of the following statement is incorrect about doubly linkedlist? (medium) (crs-be-programming)

a. Doubly linkedlist allows element to traverse in forward and backward direction
b. Previous of head and next of tail, both points to null
c. Doubly linkedlist uses more memory then singly linkedlist
d. None of the above (correct)

27. Which of the following statement is correct about ternary search? (medium) (crs-be-programming)

a. In ternary search, array is divided into three parts
b. Ternary search is more optimal than binary search
c. Ternary search is computationally more expensive(less optimal) than binary search
d. Only a and c (correct)

28. What is the time complexity of searching an element in an unsorted array using binary search? (Medium) (crs-be-programming)

a. O(n)
b. O(logn)
c. O(nlogn) (correct)
d. O(n^2)

29. What is the monotonicity of a function? (medium) (crs-be-programming)

a. The monotonicity of a function tells if the function is increasing or decreasing. (correct)
b. The monotonicity of a function tells if the function accepts arguments.
c. Both a and b
d. None of the above

30. What is the space complexity of linear search algorithm? (easy) (crs-be-programming)

a. O(n)
b. O(nlogn)
c. O(logn)
d. O(1) (correct)

Round 2

1. Max Product

Problem Statement

Given an array of integers nums, you have to choose two different indices i and j of that array. Return the maximum value of (nums[i])*(nums[j])

Constraint

- **2** <= **nums.length** <= **500**
- **1** <= **nums[i]** <= **10^3**

Input Format

- Space separated integers

Output Format

- Return top 2 max element product

Sample Input 1

9 5 12 7 8

Sample Output 1

108

Explanation of Sample 1

12 and 9 are top 2 max element and their product is 108

Sample Input 2

3 2

Sample Output 2

6

Explanation of Sample 2

3*2 = 6

Sample Input 3

1 2 3 4

Sample Output 3

12

Explanation of Sample 3

3 * 4 = 12

Solution:

```
process.stdin.resume();
process.stdin.setEncoding('utf8');

let inputString = '';
let currentLine = 0;

process.stdin.on('data', inputStdin => {
    inputString += inputStdin;
});

process.stdin.on('end', _ => {
    input = inputString.trim().split(" ").map(string => {
        return parseInt(string.trim());
    });

    console.log(maxProduct(input));
});

function maxProduct(input) {
  input.sort((a, b) => a-b);

  return input[input.length-1] * input[input.length-2];
}
```

2. Merge two Sorted Array

Problem Statement

You will be given two arrays as input which are already sorted, merge them into a single array sorted in non-decreasing order.

Constraints

- 1 <= m, n <= 200 , m and n are length of two arrays nums1 and nums2
- 2 <= m + n <= 200
- -10^9 <= nums1[i], nums2[j] <= 10^9

Input Format

- Two lines, each line containing space separated integers

Output Format

- Print the new array

Sample Input 1

1 3 5 7
2 4 6 8

Sample Output 1

1 2 3 4 5 6 7 8

Explanation of Sample 1

If we merge both the arrays it will be become as the above output i.e. 1 2 3 4 5 6 7 8 , as we need to maintain the sorted order while merging both the arrays

Sample Input 2
```
1 1 1
2 3 4
```
Sample Output 2
```
1 1 1 2 3 4
```

Sample Input 3
```
1
1
```
Sample Output 3
```
1 1
```
Explanation of Sample 3
Both arrays contain only 1 element each, merging them will yield array with 2 elements [1, 1]

Solution:
```
process.stdin.resume();
process.stdin.setEncoding('utf8');

let inputString = '';
let currentLine = 0;

process.stdin.on('data', inputStdin => {
    inputString += inputStdin;
});

process.stdin.on('end', _ => {
    inputString = inputString.trim().split("\n").map(string => {
        return string.trim().split(" ").map(x => parseInt(x));
    });

    console.log(mergeSortedArrays(inputString[0], inputString[1]));
});

function mergeSortedArrays(arr1, arr2) {
  let result = [];

  let i = 0, j = 0;
  while(i < arr1.length && j < arr2.length){
      if(arr1[i] <= arr2[j]){
          result.push(arr1[i++]);
      } else {
          result.push(arr2[j++]);
      }
  }
  while(j < arr2.length){
    result.push(arr2[j++]);
  }
  while(i < arr1.length){
    result.push(arr1[i++]);
  }
  return result.join(" ");
}
```

3. Add 2 LinkedLists

Problem Statement

You will be given 2 numbers represented using LinkedList. Write a function that returns the LinkedList that is the representation of the sum of 2 input numbers. It is not allowed to modify the lists. Each node in a LinkedList represent each digit of the input number.

Hint: Can reversing the Linkedlist help? Or maybe getting the numbers from LL, adding them and creating new one?

Constraints

- 1 <= input numbers <= 99999
- 0 <= node.value <= 9

Input Format

- Two lines, each line containing space-separated integers representing LinkedList

Output Format

- Print the sum linkedlist

Sample Input 1

1 2 3
1 2 3

Sample Output 1

2 4 6

Explanation of Sample 1

123 + 123 = 246

Sample Input 2

1 3 2
1 2 3 1

Sample Output 2

1 3 6 3

Explanation of Sample2

132 + 1231 = 1363

Sample Input 3

5 6 3

8 4 2

Sample Output 3

1 4 0 5

Explanation of Sample 3

563 + 842 = 1405

Solution:

```
process.stdin.resume();
process.stdin.setEncoding('utf8');

let inputString = '';
let currentLine = 0;

process.stdin.on('data', inputStdin => {
    inputString += inputStdin;
});

process.stdin.on('end', _ => {
    inputString = inputString.trim().split("\n").map(string => {
        return string.trim().split(" ").map(x => parseInt(x));
    });

    main(inputString[0], inputString[1]);
});

class LinkedListNode{
    constructor(value){
        this.value = value;
        this.next = null;
    }
}

function takeInput(nodes){
    let index = 0;
    if(!nodes[index]) return null;

    let head = new LinkedListNode(nodes[index++]);

    let temp = head;

    while(nodes.length > index){
        let node = new LinkedListNode(nodes[index++]);
        temp.next = node;
        temp = node;
    }
    return head;
}

function sum(root1, root2) {
  //write your logic here
  let no1 = 0;
  let no2 = 0;

  while(root1 != null){
      no1 = no1 * 10 + root1.value;
      root1 = root1.next;
  }
  while(root2 != null){
      no2 = no2 * 10 + root2.value;
      root2 = root2.next;
  }

  let no3 = no1 + no2;
  let node = null;
  if(no3 > 0){
      node = new LinkedListNode(no3%10);
      no3 = Math.floor(no3/10);
  }
  let prev = node;
  while(no3 > 0){
      let temp = new LinkedListNode(no3%10);
      no3 = Math.floor(no3/10);
      temp.next = prev;
      prev = temp;
  }

  return prev;
}

function main(input1, input2){
    let list1 = takeInput(input1);
    let list2 = takeInput(input2)

    let resultLL = sum(list1, list2);
    let output = [];
    while(resultLL != null){
        output.push(resultLL.value);
        resultLL = resultLL.next;
    }
    console.log(output.join(" "))
}
```

Feature:

1. **Design Twitter**
   Design a simplified version of Twitter. Users can login, post tweets, follow/unfollow other users
   POST /login -> signup is not required(optional), on login you can create user if not exist
   POST /tweet -> post a tweet
   POST /follow -> pass follow/unfollow status along with userId and other userId to which user wants to follow/unfollow

2. **Enhancement**
   User A should be able to fetch the list of users followed by A i.e. A is following
   Users should be able to see the news feed. Feeds should be ordered by date posted.
   GET /newsfeed

Solution:
https://github.com/shrey8599/Twitter-backend