CCEE mock test

0:59:56

Module name : Algorithms and Data Structures (Using Java)

Q.1. The time complexity of the binary search algorithm is

- O(logn)
- O(nlogn)
- O(n)
- O(n^2)

Q.2. What is the time comlexity of following code :

```
for(int i = n/2; i <= n; i++){
    for( int j = 1; j <= n; j = j * 2 ){
        System.out.println("Hello, World!");
    }
}
```

- O(n^2)
- O(n)
- O(nlogn)
- O(n/2)

Q.3. The postfix form of the expression (A+B)*C+(D-E)/F+G

- ABC+*DE-F+G/+
- A+BC+*DE-F+G+/
- AB+C*DE-F/+G+
- AB+C*DE/F-+G+

Q.4. Which of the following statement is not true about the doubly linked list?

- We can traverse in both the directions
- It requires extra space
- Implementation of doubly linked list is easier than the singly linked list
- It stores the addresses of the next and the previous node

Q.5. If several elements are competing for the same bucket in the hash table, what is it called?

- Diffusion
- Replication
- Collision
- Duplication

Q.6. The number of edges from the root to the node is called _____ of the tree.

- Height
- Depth
- Length
- Width

Q.7. What does method1 do in following code :

```
class Node {
    int item;
    Node next;

    public Node (int item) {
        this.item = item;
        this.next = null;
    }
}
public class SinglyLinkedList {
    Node head;

    public void insertAtEnd(int item) {
        Node temp = new Node (item);
        if(head == null)
        head = temp;
        else {
            Node curr = head;
            while(curr.next != null){
                curr = curr.next;
            }
            curr.next = temp;
        }
    }
}
```

- Inserting node at end
- Inserting node at beginning
- Deleting node at end
- Deleting node at beginning

Q.8. Consider the stack shown below:

| 25 | 60 | 34 | 7 | 9 | 54 | 76 | 81 | <-- Top

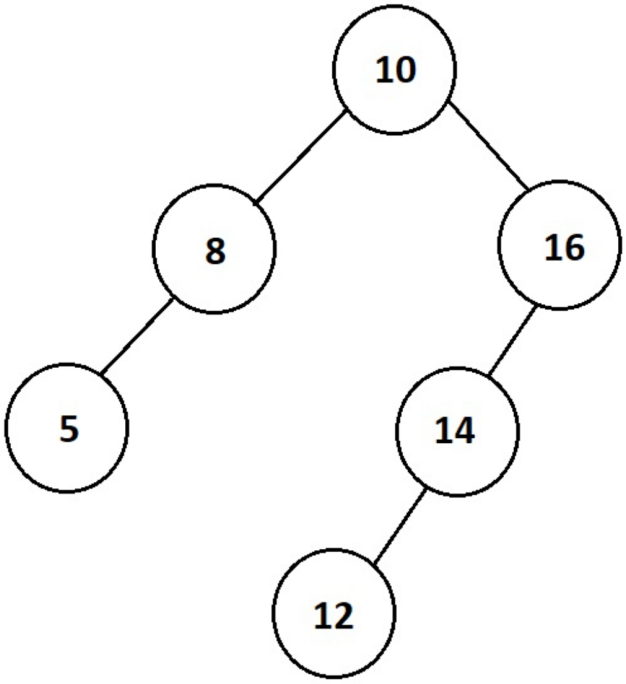
After performing the following operations in sequence, which value will be at the top of the stack ? pop, pop, push 55, pop, pop, push 12, push 48, pop, pop, pop

- 48
- 76
- 9
- 7

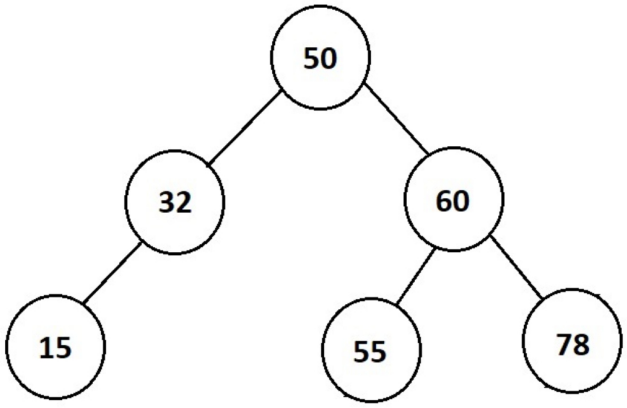
Q.9. The spanning tree of connected graph with 10 vertices contains

- 9 edges
- 10 edges
- 11 edges
- 11 vertices

Q.10. Which of the below diagram is following AVL tree property?
i.



ii.



- only i
- only ii
- both i & ii
- None of the above

Q.11. What is taking a complex problem and breaking it down into a series of small more manageable problems called?

- Decomposition
- Abstraction
- Pattern Recognition
- Algorithms

Q.12. The keys 1, 3, 12, 4, 25, 6, 18, 20, 8 are inserted into empty hash table of length 10 using open addressing with hash function h(i) = i^2 mod 10 and linear

probing, After adding all keys, which key will be at index 7?

- 6
- 12
- 18
- 1

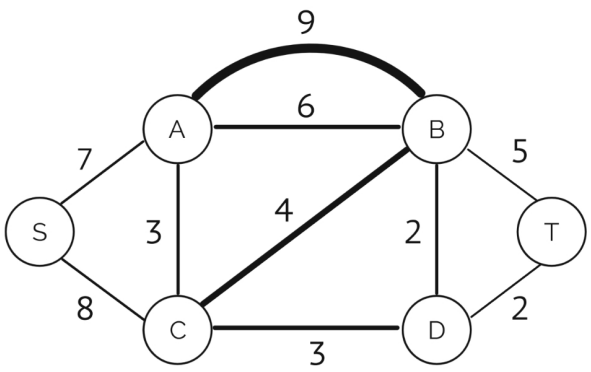
Q.13. Assume that the algorithms considered here sort the input sequences in ascending order. If the input is already in ascending order, which of the following are TRUE?
I. Quicksort runs in Θ(n2) time
II. Bubblesort runs in Θ(n2) time
III. Mergesort runs in Θ(n) time
IV. Insertion sort runs in Θ(n) time

- I and II only
- I and III only
- II and IV only
- I and IV only

Q.14. Algorithm can be represented as

- Flowchart
- Pseudocode
- All of above
- None

Q.15. Find the MST for below figure and List order in which the edged are added in MST using Kruskals algorithm.

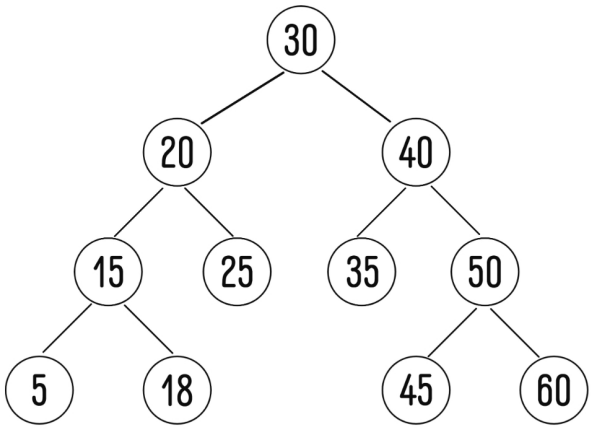


- A-C, B-D, C-D, D-T, S-A
- B-D, D-T, A-C, C-D, S-A
- S-A, B-D, C-D, A-C, D-T
- A-C, C-D, B-D, D-T, S-A

Q.16. Which of the following problems can't be solved using recursion?

- factorial of a number
- nth fibonacci number
- length of a string
- problems without base case

Q.17. What is the inorder traversal following tree?



- 30 , 20 , 15 , 5 , 18 , 25 , 40 , 35 , 50 , 45 , 60
- 5 , 18 , 15 , 25 , 20 , 35 , 45 , 60 , 50 , 40 , 30
- 5 , 15 , 18 , 20 , 25 , 30 , 35 , 40 , 45 , 50 , 60
- 5 , 18 , 15 , 60 , 50 , 40 , 30 , 25 , 20 , 35 , 45

Q.18. A linear list of elements in which deletion can be done from one end (front) and insertion can take place only at the other end (rear) is known as _____

- Queue
- Stack
- Tree
- Linked list

Q.19. Time Complexity of Breadth First Search is? (V - number of vertices, E - number of edges)

- O(V + E)
- O(V)
- O(E)
- O(V*E)

Q.20. Create a Binary search tree for the given set of strings :
JAVA, OS, DBT, WPT, DOTNET, WBJ, ASDM, DSA
What are the leaf nodes generated in the tree?

- WPT, WBJ, OS
- DOTNET, DSA, WPT

ASDM, DSA, WBJ

ASDM, DSA, JAVA

Q.21. What differentiates a circular linked list from a normal linked list?

- You cannot have the 'next' pointer point to null in a circular linked list
- It is faster to traverse the circular linked list
- You may or may not have the 'next' pointer point to null in a circular linked list
- Head node is known in circular linked list

Q.22. Which of the following is recursive postorder traversal function, if class Node is defined as follows?

```
class Node {
    public int value;
    public Node left, right;

    public Node(int element)
    {
        value = element;
        left = right = null;
    }
}
```

```
void traversePostOrder(Node node)
    {
        if (node == null)
            return;
        System.out.print(node.value + " ");
        traversePostOrder(node.left);
        traversePostOrder(node.right);
    }
```

```
void traversePostOrder(Node node)
    {
        if (node == null)
            return;
        traversePostOrder(node.left);
        System.out.print(node.value + " ");
        traversePostOrder(node.right);
    }
```

```
void traversePostOrder(Node node)
    {
        if (node == null)
            return;
        traversePostOrder(node.right);
        traversePostOrder(node.left);
        System.out.print(node.value + " ");
    }
```

```
void traversePostOrder(Node node)
    {
        if (node == null)
            return;
        System.out.print(node.value + " ");
        traversePostOrder(node.left);
        traversePostOrder(node.right);
    }
```

Q.23. What will be the best case time complexity of merge sort?

- O(n logn)

- O(n^2)
- O(n)
- O(n^2 logn)

Q.24. Depth First Search is equivalent to which of the traversal in the Binary Trees?

- Pre-order Traversal
- Post-order Traversal
- Level-order Traversal
- In-order Traversal

Q.25. At least how many comparisons are required for merging two sorted lists of n elements each?

- 2n - 1
- n - 1
- 2n + 1
- n

Q.26. A logical way of getting from the problem to the solution. If the steps you take to solve a problem follow an algorithm then they can be reused and adapted to solve similar problems in the future.

- Decomposition
- Abstraction
- Programming
- Algorithmic Thinking

Q.27. The advantage of link list over array is

- Link list can grow and shrink in size during the time
- Less space is required for storing elements
- Both 1 and 2 are correct
- None of the above

Q.28. Consider the Array: 26, 35, 1, 49, 54, 30, 99. How array will look like after 2 (two) iterations of Bubble Sort?

- 26, 1, 35, 49, 54, 30, 99
- 26, 1, 35, 49, 30, 54, 99
- 1, 26, 35, 30, 49, 54, 99
- None of these

Q.29. If already sorted array is passed to a sorting algorithm, which one will be the slowest?

- Insertion sort
- Selection sort
- Heap sort
- Merge sort

Q.30. Complete the following code if the function implements binary search

```
void binarySearch(int arr[], int first, int last, int key){
    int mid = (first + last)/2;

    ---------------- //code

    if ( first > last ){
       System.out.println("Element is not found!");
    }
  }
```

```
while( first <= last ){
    if ( arr[mid] < key ){
        first = mid + 1;
    }
    else{
        System.out.println("Element is found at index: " + mid);
        last = mid - 1;
    }
    mid = (first + last)/2;
}
```

```
while( first <= last ){
    if ( arr[mid] < key ){
        first = mid + 1;
    }else if ( arr[mid] == key ){
        System.out.println("Element is found at index: " + mid);
        break;
    }else{
        last = mid - 1;
    }
    mid = (first + last)/2;
}
```

```
while( first <= last ){
    if ( arr[mid] > key ){
        first = mid + 1;
    }else if ( arr[mid] < key ){
        System.out.println("Element is found at index: " + mid);
        break;
    }else{
        last = mid - 1;
    }
    mid = (first + last)/2;
}
```

```
while( first <= last ){
    if ( arr[mid] < key ){
        first = mid + 1;
    }else {
        System.out.println("Element is found at index: " + mid);
        break;
    }
}
```

Q.31. Which is the correct list of complexities in increasing order?

- O(1) -> O(nlogn) -> O(n^2) -> O(logn) -> O(n) -> O(n^3)
- O(n) -> O(logn) -> O(n) -> O(1) -> O(n^2) -> O(n^3)

- O(1) -> O(logn) -> O(n) -> O(nlogn) -> O(n^2) -> O(n^3)
- O(1) -> O(n) -> O(logn) -> O(nlogn) -> O(n^2) -> O(n^3)

Q.32. N-Queens Problem can be solved easily by

- Dynamic Programming
- Backtracking Method
- Greedy Method
- Divide and Conquer Method

Q.33. When new data are to be inserted into a data structure, but there is no available space; this situation is usually called

- underflow
- overflow
- housefull
- saturated

Q.34. Which of the following algorithm can be used to detect negative cycle in a graph?

- Prim
- Kruskal
- Dijkstra
- Bellman Ford

Q.35. Which of the following sorting algorithm has minimum worst case time complexity?

- Selection Sort
- Bubble Sort
- Quick Sort
- Merge Sort

Q.36. What is the time complexity of the following code:

```
int a = 0, b = 0;
for (int i = 0; i < n; ++i) {
    for (int j = 0; j < n; ++j) {
        a = a + j;
    }
}
for (int k = 0; k < n; ++k) {
    b = b + k;
}
```

- n^2
- n
- n^3
- nlogn

Q.37. In _____ , keys are stored in linked lists attached to cells of a hash table. Each list contains all the keys hashed to its cell.

- Open hashing
- Close hashing
- linear hashing
- None of the above

Q.38. Breadth First Traversal (BFS) is a method to traverse

- Graph using shortest path
- All successors of a visited node before any successors of any of those successors
- A single path of the graph as far as it can go
- None of these

Q.39. Bellmann ford algorithm provides solution for _____ problems.

- All pair shortest path
- Sorting
- Network flow
- Single source shortest path

Q.40. Applying Krushkal's algorithm to find Minimum Spanning Tree is more suitable for

- Sparse Matrix
- Dense Graph
- Sparse Graph
- None of these

Submit