

Q:-1. What will be the output of the following program?

```
public static void main(String[] args) {  
    String str = "san foundry";  
    int len = str.length();  
    Stack stack = new Stack(len);  
    for (int i = 0; i < len; i++) {  
        stack.push(str.charAt(i));  
    }  
    for (int i = 0; i < len; i++) {  
        stack.pop();  
    }  
}
```

- a) yrdnuof nas
- b) foundry nas
- c) sanfoundry
- d) san foundry

Answer: a

Explanation: First, the string 'san foundry' is pushed one by one into the stack.

When it is popped, the output will be as 'yrdnuof nas'.

Q:-2. Find the output of following code.

```
import java.util.Stack;  
public class Main {  
    public static void main(String[] args) {  
        Stack<Integer> stack = new Stack<>();  
        stack.push(5);  
        stack.push(10);  
        stack.push(15);  
        stack.pop();  
        stack.push(20);  
        System.out.println(stack.search(10));  
    }  
}
```

- a) true
- b) 2
- c) false
- d) -1

Answer: b) 2

Explanation: After operations, stack = [5, 10, 20]. 10 is 2 positions from top (1-based index).

Q:-3. Find the output of following code.

```
import java.util.Stack;  
public class Main {  
    public static void main(String[] args) {
```

```
Stack<Character> stack = new Stack<>();
String s = "abc";
for (char c : s.toCharArray()) {
    stack.push(c);
}
stack.pop();
stack.push('d');
System.out.println(stack.peek());
}
```

- a) a
- b) b
- c) c
- d) d

Answer: d) d

Explanation: Last pushed character is 'd', which replaces popped 'c'.

Q:-4.what the output of following

```
import java.util.Stack;
public class Main {
    public static void main(String[] args) {
        Stack<Integer> stack = new Stack<>();
        stack.push(1);
        stack.push(2);
        stack.push(3);
        Stack<Integer> temp = new Stack<>();
        while (!stack.isEmpty()) {
            temp.push(stack.pop());
        }
        System.out.println(temp.pop());
    }
}
```

- A) 3
- B) 2
- C) 1
- D) StackUnderflowException

Correct Answer: C) 1

Explanation: Original stack [1,2,3] reversed to [3,2,1], then popped gives 1.

Q:-4. Find the output of following code.

```
import java.util.Stack;
public class Test {
    public static void main(String[] args) {
        Stack<String> stack = new Stack<>();
        stack.push("Java");
        stack.push("Python");
    }
}
```

```
        stack.push("C++");  
        System.out.println(stack.search("Java"));  
    }  
}
```

- A) 1
- B) 2
- C) 3
- D) -1

Correct Answer: C) 3

Explanation: Stack = ["Java", "Python", "C++"]; "Java" is 3rd from top.

5. Which of the following statement(s) about stack data structure is/are NOT correct?

- a) Linked List are used for implementing Stacks
- b) Top of the Stack always contain the new node
- c) Stack is the FIFO data structure
- d) Null link is present in the last node at the bottom of the stack

Answer: c

Explanation: Stack follows LIFO.

6. Consider the following operation performed on a stack of size 5.

```
Push(1);  
Pop();  
Push(2);  
Push(3);  
Pop();  
Push(4);  
Pop();  
Pop();  
Push(5);
```

After the completion of all operation, the number of elements present in stack is?

- a) 1
- b) 2
- c) 3
- d) 4

Answer: a

Explanation: Number of elements present in stack is equal to the difference between number of push operations and number of pop operations. Number of elements is  $5-4=1$ .

7. How many stacks are required for reversing a word algorithm?

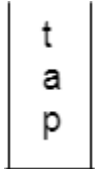
- a) one
- b) two
- c) three

d) four

Answer: a

Explanation: Only 1 stack is required for reversing a word using stack. In that stack, push and pop operations are carried out.

8. What will be result if the given stack is popped?



- a) pat
- b) tap
- c) atp
- d) apt

Answer: b

Explanation: The word 'pat' is pushed on to the stack. When the characters of the stack are popped one by one, the word 'tap' is obtained.

9. What will be output if the following sequence of operations are executed?

Push(a,s);  
Push(b,s);  
Pop(b);  
Push(c,s);

- a) abc
- b) b
- c) ac
- d) acb

Answer: b

Explanation: The element 'b' is popped out of the stack. Hence the output of the following sequence of operations will be 'b'.

10. What are the set of functions that are to be executed to get the following output?

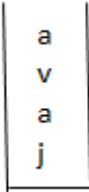
cat

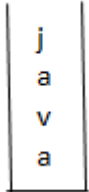
- a) push(c, s); push(a, s); push(t, s);  
pop(s); pop(s); pop(s);
- b) push(c,s); pop(s); push(a,s); pop(s);push(t,s);pop(s);
- c) pop(c ); pop(a); pop(t);
- d) push(c,s); push(a,s); pop(t);

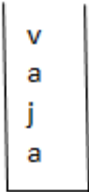
Answer: b

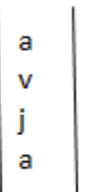
Explanation: During push operation, the characters 'c', 'a', 't' are inserted into the stack and popped immediately after push.

11. How will your stack look like if the word 'java' is pushed?

a) 

b) 

c) 

d) 

Answer: a

Explanation: When a character is pushed, it stays on the top of the stack. While popping, the word occurs in reverse order since stack follows LIFO principle.

Q12:-. Find the error (if any) in the following code snippet for pop operation.

```
void pop() //removing an element from a stack
{
    printf("%s", stack[top++]);
}
```

- a) run time error
- b) compile time error
- c) pop operation is performed, but top moved in wrong direction
- d) pop operation is performed properly

Answer: c

Explanation: The statement `printf("%s", stack[top++])` does a pop, but top gets incremented which is not correct. The statement `stack[top++]` should

be replaced with `stack[top-1]` in order to pop an operand and maintain stack properly.

13. What will be the output of the following program?

```
main()
{
    char str[]="san foundry";
    int len = strlen(str);
    int i;

    for(i=0;i<len;i++)
        push(str[i]); // pushes an element into stack

    for(i=0;i<len;i++)
        pop(); //pops an element from the stack
}
```

a) sanfoundry  
b) san foundry  
c) yrdnuof nas  
d) foundry nas

Answer: c

Explanation: First, the string 'san foundry' is pushed one by one into the stack.

When it is popped, the output will be as 'yrdnuof nas'.

Q14:-Consider these functions:

`push()` : push an element into the stack

`pop()` : pop the top-of-the-stack element

`top()` : returns the item stored in top-of-the-stack-node

What will be the output after performing these sequence of operations

`push(20);`

`push(4);`

`top();`

`pop();`

`pop();`

`push(5);`

`top();`

a) 20

b) 4

c) stack underflow

d) 5

Answer: d

Explanation: 20 and 4 which were pushed are popped by the two `pop()` statements, the recent `push()` is 5, hence `top()` returns 5.

Q:-15. Find the output of following code.

```
import java.util.PriorityQueue;
public class Main {
    public static void main(String[] args) {
        PriorityQueue<String> pq = new PriorityQueue<>();
        pq.offer("Cat");
        pq.offer("Dog");
        pq.offer("Apple");
        System.out.println(pq.poll());
    }
}
```

- A) Cat
- B) Dog
- C) Apple
- D) null

Answer: C) Apple

Explanation: PriorityQueue orders strings lexicographically; "Apple" is first.

Q:-16. Find the output of following code.

```
import java.util.LinkedList;
import java.util.Queue;
public class Main {
    public static void main(String[] args) {
        Queue<Integer> queue = new LinkedList<>();
        for (int i = 1; i <= 3; i++) {
            if (i % 2 == 0) {
                queue.offer(i);
            }
        }
        System.out.println(queue);
    }
}
```

- A) [1, 2, 3]
- B) [2]
- C) [1, 3]
- D) []

Answer: B) [2]

Explanation: Only 2 is even and added to queue.

17.The result of evaluating the postfix expression 5, 4, 6, +, \*, 4, 9, 3, /, +, \* is?

- a) 600
- b) 350

- c) 650  
d) 588

Answer: b

Explanation: The postfix expression is evaluated using stack. We will get the infix expression as

$(5*(4+6))*(4+9/3)$ . On solving the Infix Expression, we get  
 $(5*(10))*(4+3)$   
 $= 50*7$   
 $= 350$ .

18. Convert the following infix expressions into its equivalent postfix expressions.

- $(A + B \wedge D)/(E - F)+G$   
 a)  $(A B D \wedge + E F - / G +)$   
 b)  $(A B D + \wedge E F - / G +)$   
 c)  $(A B D \wedge + E F / - G +)$   
 d)  $(A B D E F + \wedge / - G +)$

Answer: a

Explanation: The given infix expression is  $(A + B \wedge D)/(E - F)+G$ .

$(A B D \wedge + ) / (E - F) +G$

$(A B D \wedge + E F - ) + G$ . '/' is present in stack.

$A B D \wedge + E F - / G +$ . Thus Postfix Expression is  $A B D \wedge + E F - / G +$ .

19. Convert the following Infix expression to Postfix form using a stack.  $x + y * z + (p * q + r) * s$ , Follow usual precedence rule and assume that the expression is legal.

- a)  $xyz*+pq*r+s*+$   
 b)  $xyz*+pq*r+s+*$   
 c)  $xyz+*pq*r+s*+$   
 d)  $xyzp+**qr+s*+$

Answer: a

Explanation: The Infix Expression is  $x + y * z + (p * q + r) * s$ .

$(x y z ) + (p * q + r) * s$ . '+', '\*' are present in stack.

$(x y z * + p q * r) * s$ . '+' is present in stack.

$x y z * + p q * r + s * +$ . Thus Postfix Expression is  $x y z * + p q * r + s * +$ .

20. What is the key advantage of implementing a queue using a linked list?

- a) Faster access to elements  
 b) No need to track rear and front  
 c) Dynamic size (no fixed size limit)  
 d) Uses less memory than arrays

Answer: c) Dynamic size (no fixed size limit)

Explanation: A linked list-based queue can grow and shrink as needed.



21. Which of the following defines a correct Node structure in Java for a queue?

a)

```
class Node {  
    int data;  
    Node next;  
}
```

b)

```
class Node {  
    int value;  
    Node prev;  
}
```

c)

```
class Node {  
    int data;  
    Node left, right;  
}
```

d)

```
class Node {  
    Node data;  
    int next;  
}
```

**Answer: a)**

**Explanation:** A singly linked list node for queue uses a data field and a next pointer.

22. What will be the output of this code?

```
class Node {  
    int data;  
    Node next;  
}
```

```
class Queue {  
    Node front, rear;  
  
    void enqueue(int x) {  
        Node newNode = new Node();  
        newNode.data = x;  
        if (rear == null) {  
            front = rear = newNode;  
        } else {  
            rear.next = newNode;  
            rear = newNode;  
        }  
    }  
}
```

```
int peek() {  
    return front.data;  
}  
  
public class Main {  
    public static void main(String[] args) {  
        Queue q = new Queue();  
        q.enqueue(10);  
        q.enqueue(20);  
        System.out.println(q.peek());  
    }  
}
```

- a) 10
- b) 20
- c) 0
- d) Error

**Answer: a) 10**

**Explanation:** The first enqueued element (10) is at the front.

**23. After enqueueing 10 and 20 and dequeuing once, what will front point to?**

- a) Node with data 10
- b) Node with data 20
- c) null
- d) rear

**Answer: b) Node with data 20**

**Explanation:** After one dequeue, front moves to the next node (which had 20).

**24. When will rear become null in a queue using linked list?**

- a) On inserting first element
- b) On dequeuing all elements
- c) After reaching maximum capacity
- d) When front is null

**Answer: b) On dequeuing all elements**

**Explanation:** When queue becomes empty, both front and rear are set to null.

**25. Which method checks if the queue is empty?**

- a)  

```
boolean isEmpty() {  
    return front == null;  
}
```
- b)  

```
boolean isEmpty() {
```

```
        return rear == front;
    }
    c)
    boolean isEmpty() {
        return rear == null;
    }
    d)
    boolean isEmpty() {
        return front == -1;
    }
```

**Answer: a)**

**Explanation:** In a linked list, empty queue is identified when front == null.

**26. In a stack using a singly linked list, which end is used for push and pop operations?**

- a) Tail
- b) Head (Top of stack)
- c) Middle
- d) Rear

**Answer: b) Head (Top of stack)**

**Explanation:** Push and pop happen at the head for  $O(1)$  time complexity.

**27. What is the time complexity of push and pop operations in a linked list-based stack?**

- a)  $O(n)$
- b)  $O(\log n)$
- c)  $O(n \log n)$
- d)  $O(1)$

**Answer: d)  $O(1)$**

**Explanation:** Both operations involve inserting or deleting from the head node.

**28. What will be the output of the following Java code?**

```
class Node {
    int data;
    Node next;
}

class Stack {
    Node top;

    void push(int x) {
        Node temp = new Node();
        temp.data = x;
        temp.next = top;
        top = temp;
    }
}
```

```
int peek() {  
    return top.data;  
}  
  
public class Main {  
    public static void main(String[] args) {  
        Stack s = new Stack();  
        s.push(10);  
        s.push(20);  
        System.out.println(s.peek());  
    }  
}
```

- a) 10
- b) 20
- c) 0
- d) Error

**Answer: b) 20**

**Explanation:** 20 is the topmost element after two push operations.

**29. What condition is used to check if the stack is empty?**

- a) top == -1
- b) top == null
- c) top.next == null
- d) top.data == 0

**Answer: b) top == null**

**Explanation:** In linked list implementation, null top indicates empty stack.

**30. Which of the following defines a correct Node structure for a stack?**

- a)  
class Node {  
 int value;  
 Node prev;  
}
- b)  
class Node {  
 int data;  
 Node next;  
}
- c)  
class Node {  
 Node left, right;  
 int value;  
}

d)  
class Node {  
    int data;  
    Node rear;  
}

**Answer: b)**

**Explanation:** A stack uses a singly linked list where each node points to the next.

**31. What will happen when you pop from an empty stack (linked list implementation)?**

- a) Returns 0
- b) Throws an exception or shows underflow
- c) Returns null
- d) Adds a node

**Answer: b) Throws an exception or shows underflow**

**Explanation:** Stack underflow occurs when trying to pop from an empty stack.

**32. What happens in the pop() operation in a stack using linked list?**

- a) top is set to null
- b) top is moved to the next node
- c) A new node is created
- d) Rear is incremented

**Answer: b) top is moved to the next node**

**Explanation:** The top is updated to the next node to remove the top element.

**33. After pushing 5, 10, 15, and popping once, what will be at the top?**

- a) 5
- b) 10
- c) 15
- d) null

**Answer: b) 10**

**Explanation:** After popping 15, the last pushed value, 10 becomes top.

**34. Which keyword is used in Java to create new nodes dynamically in stack implementation?**

- a) newNode()
- b) create
- c) allocate
- d) new

**Answer: d) new**

**Explanation:** new is used in Java to dynamically allocate memory for objects.

35:-What will be the output?

```
class Node {
    int data;
    Node next;
}

class Stack {
    Node top;

    void push(int x) {
        Node temp = new Node();
        temp.data = x;
        temp.next = top;
        top = temp;
    }

    void pop() {
        if (top != null) top = top.next;
    }

    int peek() {
        return top.data;
    }
}

public class Main {
    public static void main(String[] args) {
        Stack s = new Stack();
        s.push(10);
        s.push(20);
        s.pop();
        System.out.println(s.peek());
    }
}
```

- a) 20
- b) 10
- c) null
- d) Compilation Error

**Answer: b) 10**

**Explanation:** After pushing 10 and 20, 20 is popped, so 10 becomes the new top.