CODING / DSA

By Abhishek Rathor

Instagram: code.abhii07 (SYNTAX ERROR)

☑ Top 50 DSA Interview Questions

🔹 ARRAY

1. Traverse an Array

```
{
 // start from first index
 for i = 0 to n-1
 {
  // print element
  print a[i]
 }
}
```

2. Insert Element in Array

```
{
 // shift elements to right
 for i = n down to pos
 {
  a[i] = a[i-1]
 }

 // insert new value
 a[pos] = value
}
```

3. Delete Element from Array

```
{
 // shift elements to left
 for i = pos to n-2
 {
  a[i] = a[i+1]
 }
}
```

4. Find Maximum Element

```
{
 // assume first element max
 max = a[0]

 // check all elements
 for i = 1 to n-1
 {
  if a[i] > max
   max = a[i]
 }
}
```

---

5. Reverse an Array

```
{
 // point first index
 i = 0

 // point last index
 j = n-1

 // swap till middle
 while i < j
 {
  swap a[i] and a[j]
  i++
  j--
 }
}
```

◆ SEARCHING

6. Linear Search

```
{
 // check each element
 for i = 0 to n-1
 {
  if a[i] == key
   print "Found"
 }
}
```

---

7. Binary Search

```
{
 // set limits
 low = 0
 high = n-1

 // repeat till range exists
 while low <= high
 {
  mid = (low + high) / 2

  if a[mid] == key
   print "Found"
 }
}
```

SORTING

8. Bubble Sort

```
{
 // outer loop
 for i = 0 to n-1
 {
  // inner loop
  for j = 0 to n-i-2
  {
   if a[j] > a[j+1]
    swap a[j] and a[j+1]
  }
 }
}
```

9. Selection Sort

```
{
 // select position
 for i = 0 to n-1
 {
  min = i

  // find smallest
  for j = i+1 to n-1
  {
   if a[j] < a[min]
    min = j
  }

  // swap
  swap a[i] and a[min]
 }
}
```

10. Insertion Sort

```
{
 // start from second element
 for i = 1 to n-1
 {
  key = a[i]
  j = i - 1
  // shift elements
  while j >= 0 and a[j] > key
  {
   a[j+1] = a[j]
   j--
  }

  // insert key
  a[j+1] = key
 }
}
```

STACK

11. Stack Push

```
{
 // move top
 top = top + 1

 // insert element
 stack[top] = value
}
```

12. Stack Pop

```
{
 // remove top element
 value = stack[top]

 // decrease top
 top = top - 1
}
```

13. Check Stack Empty

```
{
 if top == -1
  print "Empty Stack"
}
```

QUEUE

14. Queue Enqueue

```
{
 // move rear
 rear = rear + 1

 // insert element
 queue[rear] = value
}
```

---

15. Queue Dequeue

```
{
 // move front
 front = front + 1
}
```

🔷 LINKED LIST

16. Traverse Linked List

```
{
 // start from head
 ptr = head

 // till end
 while ptr != NULL
 {
  print ptr->data
  ptr = ptr->next
 }
}
```

---

17. Insert at Beginning

```
{
 // link new node
 new->next = head

 // update head
 head = new
}
```

---

18. Insert at End

```
{
 // reach last node
 while ptr->next != NULL
  ptr = ptr->next

 // attach new node
 ptr->next = new
}
```

---

19. Delete First Node

```
{
 // store head
 temp = head

 // move head
 head = head->next

 // delete node
 delete temp
}
```

---

20. Reverse Linked List

```
{
 prev = NULL
 curr = head

 while curr != NULL
 {
  next = curr->next
  curr->next = prev
  prev = curr
  curr = next
 }

 head = prev
}
```

◆ TREE

21. Inorder Traversal

```
{
 if root != NULL
 {
  inorder(root->left)
  print root->data
  inorder(root->right)
 }
}
```

---

22. Preorder Traversal

```
{
 if root != NULL
 {
  print root->data
  preorder(root->left)
  preorder(root->right)
 }
}
```

---

23. Postorder Traversal

```
{
 if root != NULL
 {
  postorder(root->left)
  postorder(root->right)
  print root->data
 }
}
```

---

24. Height of Tree

```
{
 if root == NULL
  return 0

 return 1 + max(height(left), height(right))
}
```

🔷 GRAPH

25. DFS

```
{
 visited[node] = true

 for each neighbour
 {
  if not visited
   DFS(neighbour)
 }
}
```

---

26. BFS

```
{
 enqueue(start)
 visited[start] = true

 while queue not empty
 {
  node = dequeue()

  for each neighbour
  {
   if not visited
   {
    enqueue(neighbour)
    visited[neighbour] = true
   }
  }
 }
}
```

◆ MATHEMATICAL DSA

27. GCD

```
{
 while b != 0
 {
  r = a % b
  a = b
  b = r
 }
}
```

28. LCM

```
{
 lcm = (a * b) / gcd
}
```

29. Fibonacci Series

```
{
 a = 0
 b = 1

 for i = 1 to n
 {
  print a
  c = a + b
  a = b
  b = c
 }
}
```

30. Factorial

```
{
 fact = 1

 for i = 1 to n
  fact = fact * i
}
```

◆ IMPORTANT DSA PROBLEMS

31. Two Sum

```
{
 for i = 0 to n-1
 {
  for j = i+1 to n-1
  {
   if a[i] + a[j] == target
    print "Pair Found"
  }
 }
}
```

32. Kadane Algorithm

```
{
 current = 0
 max = 0

 for i = 0 to n-1
 {
  current = current + a[i]

  if current > max
   max = current

  if current < 0
   current = 0
 }
}
```

33. Check Sorted Array

```
{
 for i = 0 to n-2
 {
  if a[i] > a[i+1]
   print "Not Sorted"
 }
}
```

34. Find Missing Number

```
{
 total = n*(n+1)/2
 missing = total - arraySum
}
```

35. Check Duplicate

```
{
 for i = 0 to n-1
 {
  for j = i+1 to n-1
  {
    if a[i] == a[j]
     print "Duplicate"
  }
 }
}
```

36. Count Frequency

```
{
 for i = 0 to n-1
  freq[a[i]]++
}
```

37. Remove Duplicates

```
{
 for i = 0 to n-1
 {
  if a[i] not in newArray
   insert a[i]
 }
}
```

38. Balanced Parentheses

```
{
 for each character
 {
  if opening
   push stack
  else
   pop stack
 }
}
```

39. Reverse String

```
{
 i = 0
 j = length - 1

 while i < j
 {
  swap s[i] and s[j]
  i++
  j--
 }
}
```

40. Count Vowels

```
{
 count = 0

 for each character
 {
  if vowel
   count++
 }
}
```

---

41. Palindrome String

```
{
 if string == reverse
  print "Palindrome"
}
```

---

42. Palindrome Number

```
{
 if original == reverse
  print "Palindrome"
}
```

---

43. Power of Number

```
{
 result = 1

 for i = 1 to p
  result = result * n
}
```

---

44. Armstrong Number

```
{
 sum = 0
 temp = n

 while temp > 0
 {
  digit = temp % 10
  sum = sum + $digit^3$
  temp = temp / 10
 }
}
```

---

45. Count Digits

```
{
 count = 0
 while n > 0
 {
  count++
  n = n / 10
 }
}
```

46. Sum of Digits

```
{
 sum = 0

 while n > 0
 {
  sum = sum + n % 10
  n = n / 10
 }
}
```

47. Even or Odd

```
{
 if n % 2 == 0
  print "Even"
 else
  print "Odd"
}
```

48. Prime Number

```
{
 flag = true

 for i = 2 to n-1
 {
  if n % i == 0
   flag = false
 }
}
```

49. Second Largest Element

```
{
 largest = -1
 second = -1

 for i = 0 to n-1
 {
  if a[i] > largest
  {
   second = largest
   largest = a[i]
```

```
    }
  }
}
```

50. Count Words

```
{
 count = 1

 for each character
 {
  if character == space
   count++
 }
}


}
```

Connect & Share:

Tag us on your success stories:

- **Instagram**: @code.abhii07
- **YouTube**: SYNTAX ERROR