FOOMO AI Interview – Medium Coding Round

These questions are designed for medium-level screening of fresher software engineers. Show the code snippets to candidates and ask them about logic, output, complexity, and possible optimizations.

Question 1 — Longest Substring Without Repeating Characters

```
s = "abcabcbb"
seen = set()
l = 0
max_len = 0

for r in range(len(s)):
    while s[r] in seen:
        seen.remove(s[l])
        l += 1
    seen.add(s[r])
    max_len = max(max_len, r - l + 1)

print(max_len)
```

Question 2 — Product of Array Except Self

```
arr = [1,2,3,4]
n = len(arr)

left = [1]*n
right = [1]*n

for i in range(1, n):
    left[i] = left[i-1] * arr[i-1]

for i in range(n-2, -1, -1):
    right[i] = right[i+1] * arr[i+1]

result = [left[i]*right[i] for i in range(n)]
print(result)
```

Question 3 — Detect Cycle Idea

```
slow = 0
fast = 0
arr = [1,2,3,4,2]

while fast < len(arr)-1:
    slow += 1
    fast += 2
    if slow == fast:
        print("Cycle detected")
        break
```

Question 4 — Subarray Sum Equals K

```
arr = [1,1,1]
k = 2

count = 0
prefix = 0
seen = {0:1}

for num in arr:
    prefix += num
    if prefix - k in seen:
        count += seen[prefix - k]
    seen[prefix] = seen.get(prefix, 0) + 1

print(count)
```

Question 5 — Merge Intervals

```
intervals = [[1,3],[2,6],[8,10],[15,18]]
intervals.sort()

merged = [intervals[0]]

for start, end in intervals[1:]:
    last_end = merged[-1][1]
    if start <= last_end:
        merged[-1][1] = max(last_end, end)
    else:
        merged.append([start, end])

print(merged)
```

Question 6 — Group Anagrams

```
words = ["eat","tea","tan","ate","nat","bat"]
groups = {}

for word in words:
    key = "".join(sorted(word))
    groups.setdefault(key, []).append(word)

print(list(groups.values()))
```

Question 7 — Top K Frequent Elements

```
nums = [1,1,1,2,2,3]
k = 2

freq = {}
for n in nums:
    freq[n] = freq.get(n, 0) + 1

sorted_items = sorted(freq, key=freq.get, reverse=True)
print(sorted_items[:k])
```

Question 8 — Longest Consecutive Sequence

```
nums = [100,4,200,1,3,2]
num_set = set(nums)
longest = 0

for n in num_set:
    if n - 1 not in num_set:
        length = 1
        while n + length in num_set:
            length += 1
        longest = max(longest, length)

print(longest)
```

Question 9 — Kth Largest Element

```
nums = [3,2,1,5,6,4]
k = 2

nums.sort(reverse=True)
print(nums[k-1])
```

Question 10 — Combination Sum

```
nums = [2,3,6,7]
target = 7
result = []

def dfs(remain, path, start):
    if remain == 0:
        result.append(path)
        return
    if remain < 0:
        return

    for i in range(start, len(nums)):
        dfs(remain - nums[i], path + [nums[i]], i)

dfs(target, [], 0)
print(result)
```

Question 11 — Rotate Array

```
nums = [1,2,3,4,5,6,7]
k = 3

k %= len(nums)
nums[:] = nums[-k:] + nums[:-k]

print(nums)
```

Question 12 — Find Duplicate Number

```
nums = [1,3,4,2,2]
seen = set()

for n in nums:
    if n in seen:
        print(n)
        break
    seen.add(n)
```

Question 13 — Search in Rotated Sorted Array

```
nums = [4,5,6,7,0,1,2]
target = 0

l, r = 0, len(nums)-1

while l <= r:
    mid = (l+r)//2
    if nums[mid] == target:
        print(mid)
        break
    elif nums[l] <= nums[mid]:
        if nums[l] <= target < nums[mid]:
            r = mid - 1
        else:
            l = mid + 1
    else:
        if nums[mid] < target <= nums[r]:
            l = mid + 1
        else:
            r = mid - 1
```

Question 14 — Minimum Window Length ≥ Target

```
nums = [2,3,1,2,4,3]
target = 7

l = 0
curr_sum = 0
min_len = float('inf')

for r in range(len(nums)):
    curr_sum += nums[r]
    while curr_sum >= target:
        min_len = min(min_len, r - l + 1)
        curr_sum -= nums[l]
        l += 1

print(min_len)
```

Question 15 — Spiral Matrix Traversal

```
matrix = [[1,2,3],
          [4,5,6],
          [7,8,9]]

result = []
while matrix:
    result += matrix.pop(0)
    matrix = list(zip(*matrix))[::-1]

print(result)
```

Question 16 — Number of Islands Idea

```
grid = [["1","1","0"],
        ["1","0","0"],
        ["0","1","1"]]

count = sum(row.count("1") for row in grid)
print(count)
```

Question 17 — Valid Parentheses

```
s = "()[]{}"
stack = []
pairs = {')':'(', ']':'[', '}':'{'}

valid = True
for ch in s:
    if ch in pairs.values():
        stack.append(ch)
    else:
        if not stack or stack.pop() != pairs[ch]:
            valid = False

print(valid)
```

Question 18 — Sort Colors

```
nums = [2,0,2,1,1,0]
nums.sort()
print(nums)
```

Question 19 — Next Greater Element

```
nums = [2,1,2,4,3]
stack = []
res = [-1]*len(nums)

for i in range(len(nums)):
    while stack and nums[i] > nums[stack[-1]]:
        idx = stack.pop()
        res[idx] = nums[i]
    stack.append(i)

print(res)
```

Question 20 — Word Break Check

```
s = "leetcode"
wordDict = {"leet","code"}

dp = [False]*(len(s)+1)
dp[0] = True

for i in range(1, len(s)+1):
    for j in range(i):
        if dp[j] and s[j:i] in wordDict:
            dp[i] = True

print(dp[-1])
```