4 June:

Date : __/__/____

1. Subarrays GCD - final void Intervals:

Approach:

use a sliding window / two pointer style with GCD tracking:

→ Iterate over each possible starting index i.
→ Initialize current GCD = 0.
→ for each j from i to n-1
→ Compute currentGCD = gcd(currentGCD, A[j])
→ if currentGCD < k → break
→ if currentGCD == k → increment count

Code:

```
public class SubarrayGCD {
    int count = 0;
    for(int i=0; i<n; i++) {
    int currentGCD = 0;
    for(int j=i; j<n; j++) {
currentGCD = gcd(currentGCD, A[j]);
    if (currentGCD <k)
        break;
    if (currentGCD == k)
        count++;
    }
}
Return count;
}
}.
```

Date : __/__/__

Dry Run:

→ i=0
  j=0 → count=1.
  j=1 → count=2.
  j=2 → count=3
  j=4 → count=5

→ i=1.
  j=1 → GCD(0,4)=4
  j=2 → count=6.
  j=3 → count=7
  j=4 → count=8.

→ i=2.
  j=2 → GCD(0,6)=6
  j=3 → count=9.
  j=4 → count=10.

→ i=3
  j=3 → count=11
  j=4. → count=12.

→ i=4.
  j=4 → GCD(0,8)=8.

Count reaches = 12.

Time Complexity: $O(N \log A_{max})$.

Space Complexity: O(1)

Date : __/__/____

2. Jump Game Minimum steps

Approach:
→ maxReach = J[0], steps = J[0], jumps = 1.
→ At each position:
update maxReach.
Consume a step.
If no step left: break
otherwise increment jump & reset steps from new maxReach.

Java Code:

```
if (N==1) return 0;
if (J[0]==0) return -1;
int maxReach = J[0];
int steps = J[0];
int jumps = 1;
for(int i=1; i<N; i++){
    if(i==N-1)
        return jumps;
maxReach = Math.max(maxReach, i+J[i]);
        steps--;
    if(steps==0){
        jumps++;
    if(i>=maxReach)
            return -1;
    steps = maxReach-i;
    }
}
return -1;
}
}
```

Dry Run:

maxReach = 2, steps = 2, jumps = 1.

→ i = 1

maxReach = max(2, 1+3) = 4

steps = 1.

→ i = 2.

maxReach = max(4, 2+1) = 4.

steps = 0 → need to jump.

jumps = 2.

if i >= maxReach → no (2 < 4)

steps = 4 - 2 = 2.

→ i = 3.

maxReach = max(4, 3+1) = 4.

steps = 1.

→ i = 4

maxReach = max(4, 4+4) = 8.

steps = 0 → need to jump.

jumps = 3.

if (i >= maxReach → no (4 < 8)

steps = 8 - 4 = 4.

→ i = 5

Already reached last index

Time : O(N)

Space : O