Day-49: Search a 2D Matrix (LeetCode-74)

Problem Statement:
You are given a matrix where each row is sorted from left to right and the first element of each row is greater than the last element of the previous row. Determine whether a target value exists in the matrix.

Example:
Matrix:
[[1,3,5,7], [10,11,16,20], [23,30,34,60]]
Target = 3

Approach 1: Brute Force
Check every element of the matrix.

Time Complexity: O(n × m)
Space Complexity: O(1)

Java Code:
```
for(int i=0;i  for(int j=0;j    if(matrix[i][j]==target) return true;
  }
}
return false;
```

Python Code:
```
for row in matrix:
  for val in row:
    if val==target: return True
return False
```

C++ Code:
```
for(auto row:matrix){
  for(int val:row){
    if(val==target) return true;
  }
}
return false;
```

Approach 2: Binary Search on Flattened Matrix
Treat the matrix as a sorted 1D array and apply binary search.

Time Complexity: O(log(n × m))
Space Complexity: O(1)

Java Code:
```
int low=0, high=rows*cols-1;
while(low<=high){
  int mid=(low+high)/2;
  int val=matrix[mid/cols][mid%cols];
  if(val==target) return true;
  else if(val  else high=mid-1;
}
return false;
```

Python Code:
```
low,high=0,rows*cols-1
```

```
while low<=high:
  mid=(low+high)//2
  val=matrix[mid//cols][mid%cols]
  if val==target: return True
  elif val  else: high=mid-1
return False
```

C++ Code:

```
int low=0, high=rows*cols-1;
while(low<=high){
  int mid=(low+high)/2;
  int val=matrix[mid/cols][mid%cols];
  if(val==target) return true;
  else if(val  else high=mid-1;
}
return false;
```

Interview Tip:

LeetCode-74 is best solved using binary search by flattening the matrix logically into a 1D sorted array.