35. Search Insert Position

Search Insert Position

```
class Solution:
    def searchInsert(self, nums: List[int], target: int) -> int:
        for i in range(len(nums)):
            if (nums[i]==target):
                return i
            if(nums[i]>target):
                return i
        return i+1
```

69. Sqrt(x)

Sqrt(x)

```
class Solution:
    def mySqrt(self, x: int) -> int:

        if(x < 4): return 1 if (x!=0) else 0

        start=2;
        end=x//2;
        #int mid;

        while(start<=end):
            mid=(start+end)//2

            if(mid**2==x):
                return mid

            if(mid**2<x):
                start=mid+1

            elif(mid**2>x):
                end=mid-1
        return end;
```

167. Two Sum II - Input array is sorted

349. Intersection of Two Arrays

Intersection of Two Arrays

```
class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        i=0
        j=0

        res=set()
        nums1.sort()
        nums2.sort()

        while(i<len(nums1) and j<len(nums2)):
            if nums1[i]<nums2[j]:
                i+=1
                continue

            if nums1[i]>nums2[j]:
                j+=1
                continue

            res.add(nums1[i])
            i+=1
            j+=1
        return res
```

374. Guess Number Higher or Lower

Guess Number Higher or Lower

```
class Solution:
    def guessNumber(self, n: int) -> int:
        start=0;
        end=n;
        #int mid;

        while(start<=end):
            mid=(start+end)//2

            g=guess(mid)

            if(g==0):
                return mid

            elif(g==-1):
                end=mid-1

            else:
                start=mid+1
        return start
```

744. Find Smallest Letter Greater Than Target

Find Smallest Letter Greater Than Target

```
class Solution:
    def nextGreatestLetter(self, letters: List[str], target: str) -> str:
        if target<letters[0] or target >= letters[-1]:
            return letters[0]

        i=0
        while(target>=letters[i]):
            i+=1
        return letters[i]
```

704. Binary Search

Binary Search

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
        start=0;
        end=len(nums)-1;
        #int mid;

        while(start<=end):
            mid=(start+end)//2

            if(nums[mid]==target):
                return mid

            if(nums[mid]<target):
                start=mid+1

            elif(nums[mid]>target):
                end=mid-1
        return -1;
```

852. Peak Index in a Mountain Array

Peak Index in a Mountain Array

```
class Solution:
    def peakIndexInMountainArray(self, A: List[int]) -> int:
        if len(A)<2:
            return 0

        else:
            for i in range(len(A)-1):
                if A[i]>A[i+1]:
                    return i
```

1337. The K Weakest Rows in a Matrix

The K Weakest Rows in a Matrix

```
class Solution:
    def kWeakestRows(self, mat: List[List[int]], k: int) -> List[int]:
        sam=[]

        for i in range(len(mat)):
            sam.append((sum(mat[i]),i))
        sam.sort()
        res=[]
        for i in range(k):
            res.append(sam[i][1])
        return res
```

1351. Count Negative Numbers in a Sorted Matrix

Count Negative Numbers in a Sorted Matrix `class Solution: def countNegatives(self, grid: List[List[int]]) -> int: count = 0 for i in range(len(grid)): count += len(list(x for x in grid[i] if x < 0)) return count`