1. Binary Search

Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.
You must write an algorithm with O(log n) runtime complexity.

Example 1:
Input: nums = [-1,0,3,5,9,12], target = 9
Output: 4
Explanation: 9 exists in nums and its index is 4

Code

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
        l=0
        r=len(nums)-1
```

2. Search Insert Position

Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.
You must write an algorithm with O(log n) runtime complexity.

Example 1:
Input: nums = [1,3,5,6], target = 5
Output: 2

Code

```
class Solution:
    def searchInsert(self, nums: List[int], target: int) -> int:
        R, L = 0, len(nums)-1
```

3. Find Smallest Letter Greater Than Target

You are given an array of characters letters that is sorted in non-decreasing order, and a character target. There are at least two different characters in letters.
Return the smallest character in letters that is lexicographically greater than target. If such a character does not exist, return the first character in letters.

Example 1:
Input: letters = ["c","f","j"], target = "a"
Output: "c"
Explanation: The smallest character that is lexicographically greater than 'a' in letters is 'c'.

Code

```
class Solution:
    def nextGreatestLetter(self, letters: List[str], target: str) -> str:
        idx = bisect_right(letters, target)
```

4. Count Negative Numbers in a Sorted Matrix

Given a m x n matrix grid which is sorted in non-increasing order both row-wise and column-wise, return the number of negative numbers in grid.

Example 1:
Input: grid = [[4,3,2,-1],[3,2,1,-1],[1,1,-1,-2],[-1,-1,-2,-3]]
Output: 8
Explanation: There are 8 negatives number in the matrix.

Code

```
class Solution:
    def countNegatives(self, grid: List[List[int]]) -> int:
        c = 0
```

5. Guess Number Higher or Lower

We are playing the Guess Game. The game is as follows:
I pick a number from 1 to n. You have to guess which number I picked.
Every time you guess wrong, I will tell you whether the number I picked is higher or lower than your guess.
You call a pre-defined API int guess(int num), which returns three possible

results:
-1: Your guess is higher than the number I picked (i.e. num > pick).
1: Your guess is lower than the number I picked (i.e. num < pick).
0: your guess is equal to the number I picked (i.e. num == pick).
Return the number that I picked.

Example 1:
Input: n = 10, pick = 6
Output: 6

Code

```
class Solution:
    def guessNumber(self, n: int) -> int:
        left, right = 1, n
```

6. First Bad Version

You are a product manager and currently leading a team to develop a new product. Unfortunately, the latest version of your product fails the quality check. Since each version is developed based on the previous version, all the versions after a bad version are also bad.
Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad.
You are given an API bool isBadVersion(version) which returns whether version is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

Example 1:
Input: n = 5, bad = 4
Output: 4
Explanation:
call isBadVersion(3) -> false
call isBadVersion(5) -> true
call isBadVersion(4) -> true
Then 4 is the first bad version.

Code

```
class Solution:
    def firstBadVersion(self, n: int) -> int:
        left, right = 1, n
```

7. Valid Perfect Square

Given a positive integer num, return true if num is a perfect square or false otherwise.
A perfect square is an integer that is the square of an integer. In other words, it is the product of some integer with itself.
You must not use any built-in library function, such as sqrt.

Example 1:
Input: num = 16
Output: true
Explanation: We return true because 4 * 4 = 16 and 4 is an integer.

Code

```
class Solution:
    def isPerfectSquare(self, num: int) -> bool:
```

8. Sqrt(x)

Given a non-negative integer x, return the square root of x rounded down to the nearest integer. The returned integer should be non-negative as well.
You must not use any built-in exponent function or operator.
For example, do not use pow(x, 0.5) in c++ or x ** 0.5 in python.

Example 1:
Input: x = 4
Output: 2
Explanation: The square root of 4 is 2, so we return 2.

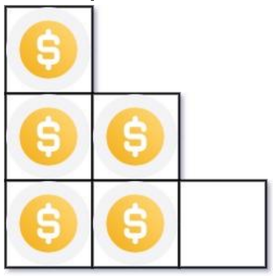
Code

```
class Solution:
    def mySqrt(self, x: int) -> int:
        left = 0
        right = x
```

9. Arranging Coins

You have n coins and you want to build a staircase with these coins. The staircase consists of k rows where the ith row has exactly i coins. The last row of the staircase may be incomplete.
Given the integer n, return the number of complete rows of the staircase you will build.

Example 1:



Input: n = 5
Output: 2
Explanation: Because the 3rd row is incomplete, we return 2.

Code

```
class Solution:
    def arrangeCoins(self, n: int) -> int:
        i=0
```

10. Kth Missing Positive Number

Given an array arr of positive integers sorted in a strictly increasing order, and an integer k.
Return the kth positive integer that is missing from this array.

Example 1:
Input: arr = [2,3,4,7,11], k = 5
Output: 9
Explanation: The missing positive integers are [1,5,6,8,9,10,12,13,...]. The 5th missing positive integer is 9.

Code

```
class Solution:
    def findKthPositive(self, nums: List[int], k: int) -> int:
        n = len(nums)
        res = [False] * 2002
```