1. **Palindrome Number**

Solution-class Solution:

def isPalindrome(self, x):

if str(x) == str(x)[::-1]:

return True

else:

return FalseGraphical user interface, application

Description automatically generated

2) A screenshot of a computer

Description automatically generated

3) **Search Insert Position**

**Solution-class Solution:**

**def searchInsert(self, nums: List[int], target: int) -> int:**

**if target in nums:**

**return nums.index(target)**

**if target < nums[0]:**

**return 0**

**else:**

**nums.append(target)**

**nums.sort()**

**return nums.index(target)**

A screenshot of a computer

Description automatically generated

4) **Multiply Strings**

Solution=class Solution:

def multiply(self, num1, num2):

s1 = int(num1)

s2 = int(num2)

s = s1\*s2

return str(s)

A screenshot of a computer

Description automatically generated

5) **Sqrt(x)**

Solution-import numpy as np

class Solution:

def mySqrt(self, x: int) -> int:\

x1=int(np.sqrt(x))

return x1

Graphical user interface, text, application

Description automatically generated

6) **Single**

Solution-class Solution:

def singleNumber(self, nums: List[int]) -> int:

list=[]

for i in nums:

if i in list:

list.remove(i)

else:

list.append(i)

return list[0]

Graphical user interface, text, application

Description automatically generated