13.ROMAN TO INTEGER

QUESTION: Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

- I can be placed before V (5) and X (10) to make 4 and 9.
- X can be placed before L (50) and C (100) to make 40 and 90.
- C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Example 1:

Input: s = "III"

Output: 3

Explanation: III = 3.

Example 2:

Input: s = "LVIII"

Output: 58

Explanation: L = 50, V= 5, III = 3.

Example 3:

Input: s = "MCMXCIV"

Output: 1994

Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.

PROGRAM:

```
class Solution:
    def romanToInt(self, s: str) -> int:
        roman={"I":1,"V":5,"X":10,"L":50,"C":100,"D":500,"M":1000}
        number=0
        for i in range(len(s)-1):
            if roman[s[i]]<roman[s[i+1]]:
                number-=roman[s[i]]
            else:
                number+=roman[s[i]]
        return number+roman[s[-1]]
```

14 LARGEST COMMON PREFIX

Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Example 1:

Input: strs = ["flower","flow","flight"]

Output: "fl"

Example 2:

Input: strs = ["dog","racecar","car"]

Output: ""

Explanation: There is no common prefix among the input strings.

Constraints:

- 1 <= strs.length <= 200

SOLUTION:

```
class Solution:
    def longestCommonPrefix(self, strs: List[str]) -> str:
        ans=""
        strs=sorted(strs)
        first=strs[0]
        last=strs[-1]
        for i in range(min(len(first),len(last))):
            if (first[i]!=last[i]):
                return ans
            ans=ans+first[i]
        return ans
```

VALID PARANTHESIS

Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Example 1:

Input: s = "()"

Output: true

Example 2:

Input: s = "()[]{}"

Output: true

Example 3:

Input: s = "(]"

Output: false

Example 4:

Input: s = "([])"

Output: true

Program:

```
class Solution:
    def isValid(self, s: str) -> bool:
        stack=[]
        closeToOpen={"]":"[",")":"(","}":"{"}
        for i in s:
            if i in closeToOpen:
                if stack and stack[-1]==closeToOpen[i]:
                    stack.pop()
                else:
                    return False
            else:
                stack.append(i)
        return True if not stack else False
```

PROGRAM:

Input: 2

Output: 2 2 1 1 $2 1 $

Input: 3

Output: 3 3 3 2 2 2 1 1 1 $3 3 2 2 1 1 $3 2 1 $

Program:

```
Def pattern(n):
for i in range(n,0,-1):
    for j in range(n,0,-1):
        for k in range(i,0,-1):
            print(j,end=" ")
    print("$",end='')
pattern(n)
```

program:

Create the multiplication table of a given number N and return the table as an array.

Example 1:

Input:

N = 9

Output:

9 18 27 36 45 54 63 72 81 90

Explanation:

The table of 9 is the output whose 1st

term is 9 and the 10th term is 90.

Answer:

```
array=[]
for i in range(10):
    array.append(N*(i+1))
return array
```

proble