tutort-academy3

May 22, 2022

1 tutort-academy3

Use the "Run" button to execute the code.

[1]:
```
!pip install jovian --upgrade --quiet
```

[2]:
```
import jovian
```

[16]:
```
# Execute this to save new versions of the notebook
jovian.commit(project="tutort-academy3")
```

```
<IPython.core.display.Javascript object>
```

```
[jovian] Updating notebook "kishkath/tutort-academy3" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/kishkath/tutort-academy3
```

[16]:
```
'https://jovian.ai/kishkath/tutort-academy3'
```

1.0.1 1. Happy Number

[4]:
```
class Solution:
    def isHappy(self, n: int) -> bool:

        #res = 0
        num = n

        arr = []

        while (num!=1):
            res = 0
            while num:
                res+= (num%10)**2
                print(res)
                num = num//10
            num = res

            if (num in arr) and (num!=0):
                return False
```

```
            else:
                arr.append(num)
        if (num==1):
            return True
        return False


## Try to get the result of sums,
## In the next step, we need to check if loop is running again, so we append it␣
 ↪to array,
## If found, return False else True.
```

1.0.2 2. Power of Two

[5]:
```
class Solution:
    def isPowerOfTwo(self, n: int) -> bool:

        binary = str(bin(n))[2:]
        if (n==1 or n==2):
            return True
        else:
            if binary[0]=='1':
                if (binary.count('0')==len(binary[1:])):
                    return True
        return False
```

1.0.3 3. Valid Anagram

[6]:
```
class Solution:
    def isAnagram(self, s: str, t: str) -> bool:

        if len(s)!=len(t):
            return False

        S,T = {},{}

        for i in range(len(s)):
            S[s[i]] = 1+S.get(s[i],0)
            T[t[i]] = 1+T.get(t[i],0)
        #print(S,T)

        for i in S:
            if S[i]!=T.get(i,0):
                #print(S[i],T.get(i,0))
```

```
                # print(S[i],T.get(i,0))
                return False
            #print(S[i],T.get(i,0))
        return True
```

1.0.4 4. Ugly Number

[7]:
```
class Solution:
    def isUgly(self, n: int) -> bool:

        if (n<=0):
            return False
        elif (n==1):
            return True

        while n>1:
            if n%2==0:
                n = n//2
            elif n%3==0:
                n = n//3
            elif n%5==0:
                n = n//5
            else:
                return False
        return True
```

1.0.5 5. Move Zeroes

[9]:
```
class Solution:
    def moveZeroes(self, nums) -> None:

        pointer1 = 0
        pointer2 = 0
        for i in range(len(nums)):
            if nums[i]!=0:
                nums[pointer1] = nums[i]
                pointer1+=1
            else:
                pointer2 += 1
        for j in range(pointer1,len(nums)):
            nums[j] = 0
        return nums
```

1.0.6 6. Reverse Vowels of a string

[10]:
```
class Solution:
    def reverseVowels(self, s: str) -> str:

        start = 0
        end = len(s)-1
        vowels = ['a','e','i','o','u','A','E','I','O','U']
        s = list(s)
        while (start<=end):
            if (s[start]) not in vowels:
                  start+=1
                  continue
            if (s[end]) not in vowels:
                  end-=1
                  continue
            if (s[start]) in vowels:
                  s[start],s[end] = s[end],s[start]
                  start+=1
                  end -= 1

        return "".join(map(str,s))
```

1.0.7 7. Third Maximum Number

[12]:
```
class Solution:
    def thirdMax(self, nums) -> int:


        first_maxi = - (2**(32))
        second_maxi = - (2**(32))
        third_maxi =  - (2**(32))

        if len(set(nums))<3:
            return max(nums)

        for curr_max in nums:

            if ((curr_max) > first_maxi):
                third_maxi = second_maxi
                second_maxi = first_maxi
                first_maxi = curr_max
                print('First',first_maxi,second_maxi,third_maxi)

            elif (curr_max!=first_maxi) and (curr_max > second_maxi):
```

```
                third_maxi = second_maxi
                second_maxi = curr_max
                print('second',first_maxi,second_maxi,third_maxi)

            elif (curr_max!=first_maxi) and (curr_max!=second_maxi) and␣
 ↪(curr_max > third_maxi):
                third_maxi = curr_max
                print('third',first_maxi,second_maxi,third_maxi)


        return third_maxi

## IF length of non-duplicates of list is less than, return maximum of list.

## If current element is greater than first_maximum,
##### Then, current_element will be come first_maximum, first_maximum will be␣
 ↪assigned to second_maximum and second_maximum will be assigned to␣
 ↪third_maximum.

### ELIF:
## If current element is greater than second_maximUm,
#### Then, current element will be come second maximum, and second_maximum will␣
 ↪be assigned to currnt elemnt

### ELIF:
## If current elemnt is greater than third_maximum,
#### THen, current elemnet will be third maximum. As the current element is␣
 ↪less than first and second maximuums, no need to change them.
```

1.0.8 8. Find the Difference

[13]:
```
class Solution:
    def findTheDifference(self, s: str, t: str) -> str:

        for i in t:
            if (t.count(i)!=s.count(i)):
                return i

## For every iteration check the count of elements in t, and return the element␣
 ↪which dont have
## Same count in both t and s.
```

1.0.9 9. Add Digits

[14]:
```python
class Solution:
    def addDigits(self, num: int) -> int:


        while (num>9):

            res = 0
            while (num):
                res += num%10
                num = num//10
            num = res

        return num
```

1.0.10 10. Sum of Digits of String After Convert

[15]:
```python
class Solution:
    def getLucky(self, s: str, k: int) -> int:

        res = 0

        dicts = {}
        count = 1

        for i in range(97,123):
            dicts[chr(i)] = count
            count+=1

        num = ""
        for i in s:
            num += str(dicts[i])

        num = int(num)
        while (k!=0):
            sums = 0
            while (num):
                sums += int(num%10)
                num = num//10
            k-=1
            num = sums
        return num
```

[ ]: