In [248…

```
#Q1. You are writing code for a company. The requirement of the company is that you create a python
#function that will check whether the password entered by the user is correct or not. The function should
#take the password as input and return the string “Valid Password” if the entered password follows the
#below-given password guidelines else it should return “Invalid Password”.
#Note: 1. The Password should contain at least two uppercase letters and at least two lowercase letters.
#      2. The Password should contain at least a number and three special characters.
#      3. The length of the password should be 10 characters long.
```

In [1]:

```
Pass = input("")

upper_letters = []
lower_letters = []
num = []
spl_char = []

for i in Pass:
    if i == i.upper():
        upper_letters.append(i)
    elif i == i.lower():
        lower_letters.append(i)

for i in Pass:
    for j in i:
        if j == '1' or j == '2' or j == '3' or j == '4' or j == '5' or j == '6' or j == '7' or j == '8' or j == '9' o
            num.append(j)

for i in Pass:
    for j in i:
        if j == '!' or j == '@' or j == '#' or j == '$' or j == '%' or j == '^' or j == '&' or j == '*' or j == '~'
            spl_char.append(j)

print(upper_letters, "minimum 2 upper_letter")
print(lower_letters, "minimum 2 lower_letters")
print(num, "minimum 1 number")
print(spl_char, "minimum 3 spl_char")

if len(upper_letters)>=2 and len(lower_letters)>=2 and len(num)>=1 and len(spl_char)>=3 and len(Pass) == 10:
    print("Valid Password")
else:
    print("Invalid Password")
```

```
HArs@#$456
['H', 'A', '@', '#', '$', '4', '5', '6'] minimum 2 upper_letter
['r', 's'] minimum 2 lower_letters
['4', '5', '6'] minimum 1 number
['@', '#', '$'] minimum 3 spl_char
Valid Password
```

In [ ]:

In [1]:
```
#Q2. Solve the below-given questions using at least one of the following:
# 1. Lambda functioJ
# 2. Filter functioJ
# 3. Zap functioJ
# 4. List ComprehensioI
```

In [2]:
```
#Filter odd numbers from the given list.
l = [1,2,3,4,5,6,7,8,9,10]

list(filter(lambda x:x%3==0, l))
```

Out[2]:
```
[3, 6, 9]
```

In [ ]:

In [3]:
```
#Check if a given number is even.
```

In [4]:
```
x = [6]
```

In [5]:
```
list(print("Even Number") for i in x if i%2==0)
```

```
Even Number
```

Out[5]:
```
[None]
```

In [ ]:

In [6]:
```
#Find the squares of numbers from 1 to 10.

l1 = list(range(1,11))

def sq(x):
```

```
    return x**x

list(map(sq, l1))
```

Out[6]: [1, 4, 27, 256, 3125, 46656, 823543, 16777216, 387420489, 10000000000]

In [ ]:

In [7]:
```
#Find the cube root of numbers from 1 to 10.

l2 = list(range(1,11))

def cube_root(x):
    return x**(1/3)

print(list(map(cube_root, l2)))
```

[1.0, 1.2599210498948732, 1.4422495703074083, 1.5874010519681994, 1.7099759466766968, 1.8171205928321397, 1.912931182772389, 2.0, 2.080083823051904, 2.154434690031884]

In [ ]:

In [ ]:

In [ ]:

In [8]:
```
#Check if the string starts with a particular letter.
```

In [1]:
```
def abc(x):
    if x[0] == 'a' or x[0] == 'b' or x[0] == 'c' or x[0] == 'd' or x[0] == 'e' or x[0] == 'f' or x[0] == 'g' or x[0]
        print("yes, it is starting with a particular letter")
    else:
        print("no, the string is starting with a numeric value")
```

In [2]:
```
string = ["56nfhnf"]
```

In [3]:
```
new1 = list(map(abc, string))
```

no, the string is starting with a numeric value

In [4]:
```
obc = ["gbjc546dfsg"]
```

In [5]:
```
new2 = list(map(abc, obc))
```

yes, it is starting with a particular letter

In [ ]:
```

```

In [14]:
```
#Check if the string is numeric.
```

In [7]:
```
def str_num(k):
    for x in k:
        if x == 'a' or x == 'b' or x == 'c' or x == 'd' or x == 'e' or x == 'f' or x == 'g' or x == 'h' or x == 'i' 
            print("no, string is not numeric")
            break
```

In [8]:
```
x = ["45154gh36547"]
```

In [9]:
```
check_str_num = list(map(str_num, x))
```

no, string is not numeric

In [10]:
```
y = ["56462fndf4754"]
```

In [11]:
```
check_str_num = list(map(str_num, y))
```

no, string is not numeric

In [12]:
```
z = ["6546457"]
```

In [17]:
```
check_str_num = list(map(str_num, z)) #accept the argument without any comment or error.......... it means it is num
```

In [18]:
```
a = ["54*64@"]
```

In [19]:
```
check_str_num = list(map(str_num, a))
```

no, string is not numeric

In [20]:
```
#HERE, IF THE STRING HAS ANY ALPHA CHARACTER,THE OUPUT IS "NO, STRING IS NOT MUMERIC" and WHEN THE STRING IS NUMERIC,
```

In [ ]:

In [ ]:

In [ ]: