


12 June 2024

TOPICS COVERED-> Python Functions, lambda function, recursive functions all categories of Function arguments, list comprehension & dictionary comprehension

1. Write a Python function that takes a string as input and returns the reverse of the string.

```
def reverse_string(input_string):  
    return input_string[::-1]  
reverse_string("jashan")  
print(reverse_string)
```

 <function reverse_string at 0x79e1eaaa9cf0>


2. Create a list of numbers. Write a function that finds and returns the maximum value in the list without using the built-in max() function.

```
number=[43,95,67,99]  
def find(number):  
    max_num=number[0]  
    for i in number[1:]:  
        if i>max_num:  
            max_num=i  
    return max_num  
print(find(number))
```

 99


3. Define a function that accepts a list of integers and returns a new list containing only the even numbers from the original list

```
num=[2,3,14,35,66]  
def list(val):  
    mylist1 = [ i for i in num if i % 2 == 0]  
    print(mylist1)  
list(num)
```

 [2, 14, 66]


4. Implement a function that takes a list of strings and returns a new list with the strings sorted by their lengths in ascending order.

```
str=["Divya", "Avi","Manav","Kriti"]  
def value(str):  
    return sorted(str)  
print(value(str))
```

 ['Avi', 'Divya', 'Kriti', 'Manav']

5. Create a dictionary with student names as keys and their corresponding ages as values. Write a function to find and print the names of students who are above a certain age.

```
def student_detail(**g):  
    for key, value in g.items():  
        if(value>15):  
            print(f"{key}: {value}")  
  
student_detail(divya=12, riya=19, punam=14)
```

 riya: 19

6. Develop a Python function that calculates the sum of squares for a given range of numbers.

```
def sum_of_square(start,end):
    total=0
    for i in range(start,end+1):
        total+=i*i
    return total
start=1
end=5
print(sum_of_square(start,end))
```

↗ 55

7. Use a lambda function to filter a list of integers and return a new list containing only the numbers greater than 10.

```
number=[12,3,44,5,36]
filter_numbers=(filter(lambda x: x>10, number))
print(filter_numbers)
```

↗ <filter object at 0x79e1eaa5ada0>

8. Write a recursive function that calculates the Fibonacci sequence for a given term.

```
def fibo(n):
    if n<=1:
        return n
    else:
        return fibo(n-1) + fibo(n-2)
value=6
print(fibo(value))
```

↗ 8

9. You have a list of names. Use list comprehension to create a new list that contains only the names that start with the letter "A"

```
names=["Divya","Vishu","Aradhiaya","Deep","Manav","Avi"]
value=[name for name in names if name.startswith('A')]
print(value)
```

↗ ['Aradhiaya', 'Avi']

10. Given a list of temperatures in Fahrenheit, use list comprehension to convert each temperature to Celsius and create a new list with these values

```
fahr_temperature=[32,45,67,89,100,211]
celsius_temp=[(temp-32)*5/9 for temp in fahr_temperature]
print(celsius_temp)
```

↗ [0.0, 7.222222222222222, 19.444444444444443, 31.666666666666668, 37.77777777777778, 99.44444444444444]

11. You are processing a list of numbers. Use list comprehension to create a new list that contains the squares of all the numbers in the original list.

```
numbers=[1,2,3,4,5,6]
value=[i**i for i in numbers]
print(value)
```

↗ [1, 4, 27, 256, 3125, 46656]

12. You have a list of words. Use list comprehension to create a new list that contains only the words that have more than 5 characters.

```
list=["Divya", "Manav", "Riya", "Komalpreet", "Jashandeep", "Harmandeep"]
val=[i for i in list if len(i)>5]
print(val)
```

↗ ['Komalpreet', 'Jashandeep', 'Harmandeep']

13. Given a list of employees and their salaries, use dictionary comprehension to create a dictionary where the keys are the employee names and the values are their salaries increased by 10%.

```
dict=[("Divya",50000), ("Manav",56000),("Riya",57000)]  
update={name : salary * 1.10 for (name , salary) in dict}  
print(update)
```

```
↵ { 'Divya': 55000.000000000001, 'Manav': 61600.000000000001, 'Riya': 62700.000000000001 }
```