1. Write a Python function to find the maximum of three numbers.

```
In [1]: def max_num(a,b,c):
return max(a,b,c)
```

```
In [3]: max_num(12,14,16)
```

```
Out [3]:16
```

2. Write a Python function to sum all the numbers in a list.

Sample List: (8, 2, 3, 0, 7)

Expected Output: 20

```
In [35]: def sum_1st (1st):
return sum(1st)
```

```
In [37]: sum_1st([8,2,3,0,7])
```

```
Out [37]: 20
```

3. Write a Python function to multiply all the numbers in a list.

Sample List: (8, 2, 3, -1, 7)

Expected Output: -336

```
In [40]: def mul_1st(1st):
result = 1
for i in 1st:
result *= i
return result
```

```
In [42]: mul_1st([8,2,3,-1,7])
```

```
Out [42]: -336
```

4. Write a Python program to reverse a string.

Sample String: "1234abcd"

Expected Output: "dcba4321"

```
In [45]: def rev_str(string):
return string[::-1]
```

```
In [47]: rev_str("1234abcd")
```

```
Out [47]: 'dcba4321'
```

5. Write a Python function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument

```
In [1]: def fact (n):
i=1
for x in range(1,n+1):
i*=x print(i)
```

```
In [3]: fact (3)
```

```
6
```

6. Write a Python function to check whether a number falls within a given range

```
In [7]: def is_in_range(num):
lower = 1
upper = 20
if lower <= num <= upper:
return True
else:
```

```
False
```

```
In [9]: is_in_range(9)
```

```
Out [9]: True
```

7. Write a Python function that accepts a string and counts the number of upper and lower case letters.

Sample String: 'The quick Brow Fox'

Expected Output: No. of Upper case characters: 3 || No. of Lower case Characters: 12

```
In [5]: def string (string):
uppercase_count = 0
lowercase_count = 0
for i in range(len(string)):
if string[i].isupper():
uppercase_count += 1
elif string[i].islower():
lowercase_count += 1
else:
pass
return f"
Uppercase letters: {uppercase_count}  Lowercase letters: {lowercase_count}
```

```
In [9]: print(string("The quick Brow Fox"))
```

```
Uppercase letters:  3  Lowercase letters :  12
```

8. Write a Python function that takes a list and returns a new list with distinct elements from the first list.

Sample List: [1,2,3,3,3,3,4,5]

Unique List: [1, 2, 3, 4, 5]

```
In [15]: def 1st (1st):

unique_1st=set (1st)
print(list (unique_1st))
```

```
In [17]: 1st([1,2,3,3,3,3,4,5])
```

```
[1, 2, 3, 4, 5]
```

9. Write a Python function that takes a number as a parameter and checks whether the number is prime or not.

Note: A prime number (or a prime) is a natural number greater than 1 and that has no positive divisors other than 1 and itself.

```
In [104]: def check (num):
if num <= 1:
return "Enter number greater than 1"
for i in range (2,num):
if num % i == 0:
return f"""It is not a prime number : {num}" return f"""It is a prime number: {num}"""'
```

```
In [110]: check (5)
```

```
Out[110]: 'It is a prime number : 5
```

10. Write a Python program to print the even numbers from a given list.

Sample List: [1, 2, 3, 4, 5, 6, 7, 8, 9]

Expected Result : [2, 4, 6, 8]

```
In [15]: def 1st ():
1st=[1,2,3,4,5,6,7,8,9]
```

```
1st1=[]
for i in range(2,len (1st),2):
1st1.append(i)
return 1st1
```

```
In [17]: 1st()
```

```
Out [17]: [2, 4, 6, 8]
```

11. Write a Python function to check whether a number is "Perfect" or not. According to Wikipedia: In number theory, a perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself (also known as its aliquot sum). Equivalently, a perfect number is a number that is half the sum of all of its positive divisors (including itself).

Example: The first perfect number is 6, because 1, 2, and 3 are its proper positive divisors, and 1+2+3 = 6. Equivalently, the number 6 is equal to half the sum of all its positive divisors: (1 + 2 +3+6)/2 = 6. The next perfect number is 28 = 1 + 2 + 4 + 7 + 14. This is followed by the perfect numbers 496 and 8128.

```
In [1]: def check_perfect_num(num):
            divider = [i for i in range(1,num) if num%i==0]
            if sum(divider) == num:
                  print(f"{num} is a perfect number"
            else:
                  print(f"{num} is not a perfect number"
```

12. Write a Python function that checks whether a passed string is a palindrome or not.

Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run.

```
In [40]: def check (string):
if string == string[-1]:
```

```
else:
return f"""{string} is palindrome'

return f"""{string} is not palindrome
```

```
In [42]: check("mam")
```

```
Out[42]: 'mam is palindrome'
```

```
In []:
"""
```