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

In [4]:
```
def find_max(lst):
    print(max(lst))
find_max([1,2,323])
```

```
323
```

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

```
-Sample List : (8, 2, 3, 0, 7)
-Expected Output : 20
```

In [7]:
```
def sum_num(lst):
    print(sum(lst))
9sum_num([1,2,3,4,4,3])
```

```
17
```

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

```
-Sample List : (8, 2, 3, -1, 7)
-Expected Output : -336
```

In [18]:
```
def multi(lst):
    b=1
    for a in lst:
        b*=a
    return b
multi([8,2,3])
```

Out[18]: 48

4. Write a Python program to reverse a string.

```
-Sample String : "1234abcd"
-Expected Output : "dcba4321"
```

In [23]:
```
def rever(input_string):
    return input_string[::-1]
rever('123awebjhbasjofh')
```

Out[23]: 'hfojsabhjbewa321'

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 [7]:
```
def factorial(input_string):
    result = 1
    for a in range(1,input_string + 1):
        result *= a
    return result
factorial(5)
```

Out[7]: 120

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

In [8]:
```
def is_in_range(number, start, end):
    return start <= number <= end
is_in_range(4,56,2)
```

Out[8]: False

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 [11]:
```
def count_case_letters(s):
    upper_count = sum(1 for char in s if char.isupper())
    lower_count = sum(1 for char in s if char.islower())
    print("No. of Upper case characters :", upper_count)
    print("No. of Lower case Characters :", lower_count)

sample_string = 'The quick Brow Fox'
count_case_letters(sample_string)
```

```
No. of Upper case characters : 3
No. of Lower case Characters : 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 [12]:
```
def get_unique_elements(lst):
    return list(set(lst))
sample_list = [1, 2, 3, 3, 3, 3, 4, 5]
unique_list = get_unique_elements(sample_list)
print("Unique List:", unique_list)
```

```
Unique List: [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 [4]:
```
def prime(input_string):
    b = 1
    for a in input_string:
        if a % 3 == 0:
            b = a
    print(f"This number is prime: {b}")
prime([1,2,4,2,3])
```

```
This number is prime: 3
```

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 [12]:
```
def even(lst):
    abc=[]
    for a in lst:
        if a % 2 == 0:
            abc.append(a)
    return abc
even([1, 2, 3, 4, 5, 6, 7, 8])
```

Out[12]:
```
[2, 4, 6, 8]
```

In [ ]:
```
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 [15]:
```
def is_perfect_number(n):
    if n < 1:
        return False
    sum = 0
    for i in range(1, n):
        if n % i == 0:
```

```
            sum += i
    return sum == n
is_perfect_number(6)
```

Out[15]: True

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 [18]:
```
def palin(a):
    if a == a[::-1]:
        return True
    else:
        return False
palin('madam')
```

Out[18]: True

In [ ]: