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

```
1   def max3(n1, n2, n3):
2     if n1==n2 and n2==n3:
3       print(n1, n2, n3, "all three are equal")
4     elif n1>n2 and n1>n3:
5       print(n1, "is greatest")
6     elif n2>n1  and n2>n3:
7       print(n2, "is greatest")
8     elif n3>n1 and n3>n2:
9       print(n3, "is greatest")
10    else:
11      print("any two are equal")
12
13  no1=float(input("Enter no1: "))
14  no2=float(input("Enter no2: "))
15  no3=float(input("Enter no3: "))
16  max3(no1, no2, no3)
```

```
Enter no1: 2
Enter no2: 6
Enter no3: 7
7.0 is greatest
```

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

```
1   '''
2   Sample List : (8, 2, 3, 0, 7)
3   Expected Output : 20
4   '''
5   def listsum(larg):
6     sum=0
7     for ele in larg:
8       sum=sum+ele
9     return sum
10
11  sample=[8, 2, 3, 0, 7]
12  print("sum: ", listsum(sample))
```

```
sum:  20
```

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

```
1 '''
2 Sample List : (8, 2, 3, -1, 7)
3 Expected Output : -336
4 '''
5 def listmul(larg):
6   product=1
```

```
7    for ele in larg:
8      product=product*ele
9    return product
10
11 mulsample=[8, 2, 3, -1, 7]
12 print("Product: ", listmul(mulsample))
```

```
    Product:  -336
```

4. Write a Python program to reverse a string.

```
1 '''
2 Sample String : "1234abcd"
3 Expected Output : "dcba4321"
4 '''
5 def strreverse(strarg):
6   return strarg[::-1]
7
8 inputstr="1234abcd"
9 print(strreverse(inputstr))
```

```
    dcba4321
```

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

```
1 def factorial(num):
2   fact=1
3   for i in range(num, 0, -1):
4     fact=fact*i
5   return fact
6
7 inputno=int(input("Enter input number: "))
8 print("Factorial of ", inputno, "is", factorial(inputno))
```

```
    Enter input number: 5
    Factorial of  5 is 120
```

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

```
1 def no_in_range(start, end, chk):
2   return chk in range(start, end+1)
3
4 starting=int(input("Enter start of range: "))
5 ending=int(input("Enter end of range: "))
6 chkno=int(input("Enter number to check in range: "))
7 print(no_in_range(starting, ending, chkno))
```

```
    Enter start of range: 1
    Enter end of range: 99
    Enter number to check in range: 99
    True
```

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

```
1 '''
2 Sample String : 'The quick Brow Fox'
3 Expected Output :
4 No. of Upper case characters : 3
5 No. of Lower case Characters : 12
6 '''
7 def count_upper_lower(strarg):
8   uppercnt, lowercnt=0, 0
9   for ch in strarg:
10     if ch.isupper():
11       uppercnt+=1
12     if ch.islower():
13       lowercnt+=1
14   return uppercnt, lowercnt
15
16 samplestr='The quick Brow Fox'
17 ucnt, lcnt=count_upper_lower(samplestr)
18 print("No. of Upper case characters :", ucnt)
19 print("No. of Lower case characters :", lcnt)
```

```
    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.

```
1 '''
2 Sample List : [1,2,3,3,3,3,4,5]
3 Unique List : [1, 2, 3, 4, 5]
4 '''
5 def unique_ele_list(listarg):
6   reslst=[]
7   for item in listarg:
8     if item not in reslst:
9       reslst.append(item)
10   return reslst
11
12 samplelst=[1,2,3,3,3,3,4,5]
13 print("Sample List :", samplelst)
14 print("Unique List :", unique_ele_list(samplelst))
```

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

```
 1 def isprime(num):
 2   flag=True
 3   for i in range(2, num//2):
 4     if num%i==0:
 5       flag=False
 6       break
 7   return flag
 8
 9 no=int(input("Enter number to check prime:"))
10 print("Prime check for", no, ":", isprime(no))
```

```
    Enter number to check prime:23
    Prime check for 23 : True
```

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

```
 1 '''
 2 Sample List : [1, 2, 3, 4, 5, 6, 7, 8, 9]
 3 Expected Result : [2, 4, 6, 8]
 4 '''
 5 def getevens(listarg):
 6   reslst=[]
 7   for ele in listarg:
 8     if ele%2==0:
 9       reslst.append(ele)
10   return reslst
11
12 samplelst=[1, 2, 3, 4, 5, 6, 7, 8, 9]
13 print(getevens(samplelst))
```

```
    [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).

```
 1 '''
 2 Example : The first perfect number is 6, because 1, 2, and 3 are its proper
 3 positive divisors, and 1 + 2 + 3 = 6. Equivalently, the number 6 is equal to half
 4 the sum of all its positive divisors: ( 1 + 2 + 3 + 6 ) / 2 = 6. The next perfect
 5 number is 28 = 1 + 2 + 4 + 7 + 14. This is followed by the perfect numbers 496
 6 and 8128.
 7 '''
 8 def isperfectno(num):
 9   sum=0
10   for i in range(1, (num//2)+1):
11     if num%i==0:
12       sum=sum+i
13   return sum==num
14
```

```
15 num=int(input("Enter number to check perfect number: "))
16 print("Perfect number check for ", num, "is", isperfectno(num))
```

```
    Enter number to check perfect number: 6
    Perfect number check for  6 is 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.

```
1 def ispallindrome(strarg):
2   return strarg==strarg[::-1]
3
4 inputstr=input("Enter a string to check pallindrome: ")
5 print("Pallindrome check for", inputstr, " :", ispallindrome(inputstr))
```
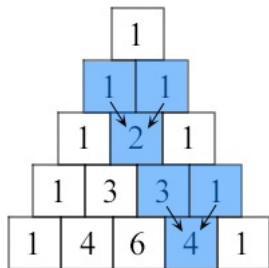
```
    Enter a string to check pallindrome: madam
    Pallindrome check for madam  : True
```

13. Write a Python function that prints out the first n rows of Pascal's triangle.

Note : Pascal's triangle is an arithmetic and geometric figure first imagined by Blaise Pascal.

Sample Pascal's triangle :

Each number is the two numbers above it added together

```
 1 def pascalt(n):
 2   for i in range(0, n+1):
 3     temp=row=11**i
 4     print(" "*(n-i), end="")
 5     while temp>0:
 6       d=temp%10
 7       temp=temp//10
 8       print(d, " ", end="")
 9     print()
10
11 pascalt(4)
```

```
       1
      1  1
     1  2  1
```

```
   1  3  3  1
   1  4  6  4  1
```

14. Write a Python function to check whether a string is a pangram or not. Note : Pangrams are words or sentences containing every letter of
    the alphabet at least once.

```python
 1 '''
 2 For example : "The quick brown fox jumps over the lazy dog"
 3 '''
 4
 5 def ispangram(strarg):
 6   strlow=strarg.lower()
 7   flag=True
 8   for i in range(ord('a'), ord('z')+1):
 9     if chr(i) not in strlow:
10       flag=False
11       break
12   return flag
13
14 samplestr="The quick brown fox jumps over the lazy dog"
15 print("Sample string:", samplestr)
16 print("Pangram check for Sample string :", ispangram(samplestr))
```

```
    Sample string: The quick brown fox jumps over the lazy dog
    Pangram check for Sample string : True
```

15. Write a Python program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated
    sequence after sorting them alphabetically.

```python
 1 '''
 2 Sample Items : green-red-yellow-black-white
 3 Expected Result : black-green-red-white-yellow
 4 '''
 5 def sorthyphensep(strarg):
 6   wlst=strarg.split("-")
 7   wlst.sort()
 8   resstr="-".join(wlst)
 9   return resstr
10
11 inpstr="green-red-yellow-black-white"
12 print(sorthyphensep(inpstr))
```

```
    black-green-red-white-yellow
```

16. Write a Python function to create and print a list where the values are the squares of numbers between 1 and 30 (both included).

```python
 1 def sqlstgen():
 2   res=[]
 3   for i in range(1, 31):
 4     res.append(i*i)
```

```
5   return res
6
7 print(sqlstgen())
```

```
    [1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900]
```

17. Write a Python program to create a chain of function decorators (bold, italic, underline etc.).

```
 1 def bold(strarg):
 2   bold = '\033[1m'
 3   end = '\033[0m'
 4   return bold+strarg+end
 5
 6 def italic(strarg):
 7   italics = '\033[3m'
 8   end = '\033[0m'
 9   return italics+strarg+end
10
11 def underline(strarg):
12   underline = '\033[4m'
13   end = '\033[0m'
14   return underline+strarg+end
15
16 def formatter(strarg, fmting):
17   if fmting=="bold":
18     strarg=bold(strarg)
19   if fmting=="italics":
20     strarg=italic(strarg)
21   if fmting=="underline":
22     strarg=underline(strarg)
23   return strarg
24
25 inputstr="this is testing string for formatting"
26 print("Bold:", formatter(inputstr, "bold"))
27 print("Italic:", formatter(inputstr, "italics"))
28 print("Underlined:", formatter(inputstr, "underline"))
```

```
    Bold: this is testing string for formatting
    Italic: this is testing string for formatting
    Underlined: this is testing string for formatting
```

18. Write a Python program to execute a string containing Python code.

```
 1 def pyrunner(code):
 2   exec(code)
 3
 4 snippet='''a, b=1, 2
 5 print("a:", a, "b:", b)
 6 print("sum of a & b:", a+b)'''
 7 print(snippet, "\n\nAbove Code is executed below: \n")
 8 pyrunner(snippet)
```

```
a, b=1, 2
print("a:", a, "b:", b)
print("sum of a & b:", a+b)

Above Code is executed below:

a: 1 b: 2
sum of a & b: 3
```

19. Write a Python program to access a function inside a function.

```
 1 def menu(a):
 2   print("From menu")
 3   def add(b):
 4     print("From sum")
 5     print("Sum: ", a+b)
 6   return add
 7
 8 no1=int(input("Enter a number: "))
 9 no2=int(input("Enter another number: "))
10 menu(no1)(no2)
```

```
Enter a number: 1
Enter another number: 4
From menu
From sum
Sum:  5
```

20. Write a Python program to detect the number of local variables declared in a function.

```
1 '''
2 Sample Output:
3 3
4 '''
5 def locals(argstr):
6   locstr=argstr
7   return locals.__code__.co_nlocals
8 inpstr="welcome to cdac mumbai"
9 print(locals(inpstr))
```

```
    2
```

```
1
```