LEVEL 1:

1. Union of two sets
2. Descending order sort list
3. Remove recurrence - list *
4. Intersection of two sets
5. Length of string
6. Concatenate 2 strings
7. Convert Tuple to list
8. Convert List to Tuple
9. Convert String to int
10. Convert string to float
11. Multiplication Table
12. Factorial with loop
13. Sum of Digits
14. Palindrome
15. Sum of first N natural numbers
16. LCM of two numbers
17. Square Root using Newton Raphson
18. 2nd largest element in list
19. GCD of two numbers
20. Remove duplicate element from list *
21. Reverse String
22. Ascending order sort list
23. Palindrome String
24. Factorial
25. Area of Triangle
26. Circumference of Circle
27. Max of two no. using ternary operator
28. Palindrome using recursion
29. Valid string pass?
30. Area of circle
31. Roots of quadratic eq
32. GCD of three numbers
33. Average of list
34. Perfect Square

35. Find largest prime no. lesser than given number
36. Valid string email?
37. Sum of all even no. in list
38. Sum of all Odd no. in list
39. Divisible by 5 in range
40. Vowel or consonant
41. Smallest of three numbers
42. Prime number
43. Even or odd number
44. Leap Year
45. Largest of three numbers
46. Positive,Negative, Zero
47. LCM (while loop)
48. Pascal Triangle(Nested loop
49. Palindrome String(For)
50. Fibonacci(For)
51. Sum of digits(while)
52. Prime(for)
53. Sum of odd b/w 2 no.s (while)
54. GCD (for)
55. Multiplication Table(Nested)
56. Factorial(for)
57. Count vowels(for)
58. Sum of even no. b/w 2 no.s (while)
59. Palindrome number (while)
60. Reverse String(for)
61. Prime no. range(for)
62. Fibonacci(while)
63. Factorial(while)
64. Multiplication Table(for)
65. Sum of no. from list(for)
66. Print 1-10(for)

1. ***Union of two sets :***

```
arr1=[1,2,3,4,5]
arr2=[3,4,5,6,7]
union=list(set(arr1+arr2))
print("union",union)
```

2. ***Descending order sort list:***

```
list = [64, 34, 25, 12, 22, 11, 90]
n = len(list)
for i in range(n):
    for j in range(0, n-i-1):
        if list[j] < list[j+1]:
            list[j], list[j+1] = list[j+1], list[j]

print("Sorted list in descending order:",list)
```

3. ***Remove recurrence - list :***

4. ***Intersection of two sets***

```
arr1=[1,2,3,4,5]
arr2=[4,5,6,7,8]
intersection=list(set(arr1)& set(arr2))
print("Intersection:",intersection)
```

5. ***Length of string:***

```
string="new horizon"
length=len(string)
print("Length of string is:",length)
```

6. ***Concatenate 2 strings:***

```
str1="hello"
str2="world"
concatenate=str1+ " "+str2
print("Concatenated string:",concatenate)
```

7. ***Convert Tuple to list:***

```
tuple=(1,2,4,5)
list=list(tuple)
print("converted:",list)
```

8. ***Convert List to Tuple:***

```
list=[1,2,3,4]
tuple=tuple(list)
print("converted:",tuple)
```

9. ***Convert String to int :***

```
str = "abc"
try:
    integer = int(str)
    print("Converted integer:", integer)
except ValueError:
    print("can't convert to integer. enter a valid integer
string.")
```

10. ***Convert string to float:***

```
str = "3.14"
float = float(str)
print("Converted float:",float)
```

11. ***Multiplication Table:***

```
num=10
print(f"Multiplication table for {num}")
for i in range(1,11):
  mul=num*i
  print(f"{num} X {i} = {mul}")
```

12. ***Factorial with loop:***

```
num=5
fact=1
counter=num
while(counter>0):
    fact=fact*counter
    counter=counter-1
print(f"The factorial of {num} is: {fact}")
```

13. Sum of Digits:

```
number = 12345
digit_sum = 0
while number > 0:
    digit = number % 10
    digit_sum += digit
    number //= 10
print("Sum of digits:", digit_sum)
```

14. Palindrome Number:

```
num=input("enter number:")
reverse_num=num[::-1]
if(reverse_num==num):
  print(f"{num} is palindrome")
else:
  print(f"{num} is not palindrome")
```

15. Sum of first N natural numbers:

```
N=int(input("Enter a positive integer N: \n"))
sum= 0
print(f"The sum of the first {N} natural numbers is:")
for i in range(1, N + 1):
  sum=sum+i

print(sum)
```

16. LCM of two numbers:

```
num1=24
num2=36
a=num1
b=num2
while b:
  a,b=b,a%b

gcd=a
lcm=(num1*num2)//gcd
print("gcd of", num1, "and", num2, "is", gcd)
print("lcm of", num1, "and", num2, "is", lcm)
```

17. ***Square Root using Newton Raphson:***

```
num=49
sqrt=num/2
tolerance=1e-7
while abs(sqrt*sqrt-num)>tolerance:
  sqrt=0.5*(sqrt+num/sqrt)

print(f"sqrt of {num} is:",sqrt)
```

18. ***2nd largest element in list:***

```
numbers=[10,6,8,2,5]
first=float('-inf')
second=float('-inf')

for num in numbers:
  if num>first:
    second=first
    first=num
  elif num>second:
    second=num

print("second largest number is:",second)
```

19. ***GCD of two numbers:***

```
num1=24
num2=36
a=num1
b=num2
while b:
  a,b=b,a%b
  gcd=a
print("gcd of", num1, "and", num2, "is", gcd)
```

20. Remove duplicate element from list:

21. Reverse String:

```
str="hello"
reverse=str[::-1]
print("reverse string is:",reverse)
```

22. Ascending order sort list:

```
list=[10,2,5,8,3]
n=len(list)
for i in range(n):
  for j in range(0,n-i-1):
    if list[j] > list[j+1]:
      list[j], list[j+1]=list[j+1], list[j]

print("ascending order:",list)
```

23. Palindrome String:

```
str="tap"
reverse=str[::-1]
if reverse==str:
  print("string is palindrome")
else:
  print("string is not palindrome")
```

24. Factorial:

```
num=5
fact=1
for i in range(1,num+1):
  fact=fact*i

print(f"factorial of {num} is:",fact)
```

25. *Area of Triangle:*

```
b=4
h=6
area=0.5*b*h
print("area of triangle is:",area)
```

26. *Circumference of Circle:*

```
r=4
pi=3.14
circumference=2*pi*r
print("Circumference of circe is:",circumference)
```

27. *Max of two no. using ternary operator:*

```
num1 = 15
num2 = 27
max= num1 if num1 > num2 else num2
print(f"The maximum of {num1} and {num2} is:
{max}")
```

28. *Palindrome using recursion:*

```
def is_palindrome(num_str):
    if len(num_str) <= 1:
        return True
    if num_str[0] == num_str[-1]:
        return is_palindrome(num_str[1:-1])
    else:
        return False
test_cases = [12321, 12345, 1221, 123454321]

for number in test_cases:
    num_str = str(number)
    result = f"{number} is a palindrome" if
is_palindrome(num_str) else f"{number} is not a
palindrome"
    print(result)
```

29. *Valid string pass?:*

30. *Area of circle:*

```
r=4
pi=3.14
area=pi*r*r
print("Area of circle is:",area)
```

31. *Roots of quadratic eq:*

```
import math
a, b, c = 1, -3, 2
d = b**2 - 4*a*c
if d >= 0:
   r1 = (-b + math.sqrt(d)) / (2*a)
   r2 = (-b - math.sqrt(d)) / (2*a)
   print(f"Roots are: {r1} and {r2}")
else:
   r1 = complex(-b / (2*a), math.sqrt(abs(d)) / (2*a))
   r2 = complex(-b / (2*a), -math.sqrt(abs(d)) / (2*a))
   print(f"Roots are: {r1} and {r2}")
```

32. *GCD of three numbers:*

```
import math
num1 = 36
num2 = 60
num3 = 84

gcd= math.gcd(math.gcd(num1, num2), num3)

print(f"The GCD of {num1}, {num2}, and {num3} is:
{gcd}")
```

33. *Average of list:*

```
numbers = [23, 56, 78, 90, 123]
total = 0
for num in numbers:
    total += num
average = total / len(numbers)
print(f"The average of the numbers {numbers} is:
{average}")
```

34. Perfect Square:

```
numbers = [16, 25, 36, 49, 50]

for number in numbers:
    is_perfect_square = False

    for i in range(1, number + 1):
        if i * i == number:
            is_perfect_square = True
            break

    if is_perfect_square:
        print(f"{number} is a perfect square")
    else:
        print(f"{number} is not a perfect square")
```

35. Find largest prime no. lesser than given number:

```
number = 50
largest_prime = None

for num in range(number - 1, 1, -1):
    if all(num % i != 0 for i in range(2, int(num**0.5) +
1)):
        largest_prime = num
        break

if largest_prime is not None:
    print(f"The largest prime number less than {number}
is: {largest_prime}")
else:
  print(f"There is no prime number less than
{number}")
```

36. ***Valid string email?:***

```
import re

emails = ["user@example.com",
  "user@example"]

pattern = r'^[\w\.-]+@[a-zA-Z\d\.-]+\.[a-zA-Z]{2,}$'

for email in emails:
  print(f"{email} is {'valid' if re.match(pattern, email)
else 'not valid'}")
```

37. ***Sum of all even no. in list:***

```
numbers=[1,2,3,4,5,6,7]
even_sum=0
for num in numbers:
 if num%2==0:
  even_sum=even_sum+num

print("Sum of even numbers is:",even_sum)
```

38. ***Sum of all Odd no. in list:***

```
numbers=[1,2,3,4,5,6,7]
odd_sum=0
for num in numbers:
 if num%2!=0:
  odd_sum=odd_sum+num

print("Sum of odd numbers is:",odd_sum)
```

39. ***Divisible by 3 or 5:***

```
num=150
if num%5==0:
  print(f"{num} is divisible by 5")
elif num%3==0:
  print(f"{num} is divisible by 3")
else:
    print(f"{num} not divisible by both")
```

40. Vowel or consonant:

```
char = 'x'
if char in 'aeiouAEIOU':
  print(f"{char} is a vowel")
else:
  print(f"{char} is a consonant")
```

41. Smallest of three numbers:

```
num1=5154
num2=423
num3=32566
if num1<num2 and num1<num3:
  print(f"{num1} is smallest")
elif num2<num1 and num2<num3:
  print(f"{num2} is smallest")
else:
  print(f"{num3} is smallest")
```

42. Prime number:

```
numbers = [2, 3, 11, 25, 29, 37]
for number in numbers:
    if number > 1 and all(number % i != 0 for i in range(2, 
int(number**0.5) + 1)):
        print(f"{number} is a prime number")
    else:
        print(f"{number} is not a prime number")
```

43. *Even or odd number:*

```
num=6
if(num%2==0):
  print("number is even")
else:
  print("number is odd")
```

44. *Leap Year:*

```
years = [2000, 2020, 2021, 1900, 2004]
for year in years:
   if (year % 4 == 0 and year % 100 != 0) or (year % 400
== 0):
      print(f"{year} is a leap year")
   else:
      print(f"{year} is not a leap year")
```

45. *Largest of three numbers:*

```
num1=5
num2=4
num3=3
if num1>num2 and num2>num3:
 print(f"{num1} is largest")
elif num2>num1 and num1>num3:
 print(f"{num2} is largest")
else:
 print(f"{num3} is largest")
```

46. *Positive,Negative, Zero:*

```
num=-8
if(num>0):
  print("number is positive")
elif(num<0):
  print("Number is negative")
elif(num==0):
  print("number is zero")
```

47. ***LCM (while loop)***:

```
num1=24
num2=36
a=num1
b=num2
while b:
 a,b=b,a%b
 gcd=a
 lcm=(num1*num2//gcd)
print(f"Lcm of {num1} and {num2} is:{lcm}")
```

48. ***Pascal Triangle(Nested loop):***

```
num_rows = 5

triangle = [[1] * (i + 1) for i in range(num_rows)]
for i in range(2, num_rows):
    for j in range(1, i):
        triangle[i][j] = triangle[i - 1][j - 1] + triangle[i - 1][j]
print(f"Pascal's Triangle with {num_rows} rows:")
for row in triangle:
    print(" ".join(map(str, row)))
```

49. ***Palindrome String(For):***

```
str = "atta"

for i in range(0,12):
  reverse = str[::-1]

if str == reverse:
    print("Palindrome")
else:
    print("Not Palindrome")
```

50.   Fibonacci(For):

```
n = 6
a, b = 0, 1
print(f"The first {n} Fibonacci numbers are:")
for _ in range(n):
   print(a, end=" ")
   a, b = b, a + b
```

51.   Sum of digits(while):

```
number = 12345
sum= 0
while number > 0:
   digit = number % 10
   sum=sum+digit
   number //= 10

print("Sum of digits:", sum)
```

52.   Prime(for):

```
num = 17
prime = num > 1

for i in range(2, int(num**0.5) + 1):
   if num % i == 0:
      prime = False
   break

if prime:
   print(f"{num} is a prime number")
else:
   print(f"{num} is not a prime number")
```

53. ***Sum of odd b/w 2 no.s (while):***

```
num1 = 1
num2 = 10
sum= 0
num = num1 + (num1 % 2)

while num <= num2:
   sum=sum+num
   num=num+ 2

print(f"The sum of odd numbers between {num1} and
{num2} is:", sum)
```

54. ***GCD (for):***

```
num1 = 24
num2 = 36
gcd=1
for i in range(1, min(num1, num2) + 1):
   if num1 % i == 0 and num2 % i == 0:
      gcd=i

print(f"The GCD of {num1} and {num2} is: {gcd}")
```

55. ***Multiplication Table(Nested):***

```
n = 10
for i in range(1, n+1):
   for j in range(1, n+1):
      print(f"{i} X {j}= {i*j}")
```

56. ***Factorial(for):***

```
num = 5
factorial = 1
for i in range(1, num + 1):
   factorial *= i

print(f"The factorial of {num} is: {factorial}")
```

57. *Count vowels(for):*

```
str="trisha"
vowel=0
for char in str:
  if char in 'aeiou':
    vowel=vowel+1

print(vowel)
```

58. *Sum of even no. b/w 2 no.s (while):*

```
num = 5
sum = 0

while num < 10:
   if num % 2 == 0:
      sum += num
   num += 1

print("Sum of even numbers:", sum)
```

59. *Palindrome number (while):*

```
num = 121
original_num = num
reversed_num = 0

while num > 0:
   reversed_num = reversed_num * 10 + num % 10
   num //= 10

if original_num == reversed_num:
   print(f"{original_num} is a palindrome")
else:
   print(f"{original_num} is not a palindrome")
```

60. *Reverse String(for):*

```
str="hello"
for _ in range(1,10):
  reverse=str[::-1]
  str=reverse

print(reverse)
```

61. *Prime no. range(for):*

```
start = 10
end = 50

print(f"Prime numbers between {start} and {end}:")
for num in range(start, end + 1):
   if num > 1:
      is_prime = True
      for i in range(2, int(num**0.5) + 1):
         if num % i == 0:
            is_prime = False
            break
      if is_prime:
         print(num)
```

62. *Fibonacci(while):*

```
num=50
a,b=0,1

print(a,end=" ")
while b<=num:
  print(b,end=" ")
  a,b=b,a+b
```

63. *Factorial(while):*

```
num=5
fact=1
while num>0:
 fact=fact*num
 num=num-1

print(f"factorial is: {fact}")
```

64. Multiplication Table(for):

```
num=5
print(f"multiplication table for {num} is:")
for i in range(1,11):
  mul=num*i
  print(f"{num} X {i} = {mul}")
```

65. Sum of no. from list(for):

```
num=[1,2,3,4,5,6]
sum=0
for num in range(6):
  sum=sum+num

print("sum of all numbers:",sum)
```

66. Print 1-10(for):

```
for i in range(1,11):
  print(i)
```

LEVEL 2:

1. Concatenate 2 lists(Generator function)
2. Length of longest word(Reduce functions)
3. Prime no. (Filter Function)
4. Count vowels(Map Function)
5. Lcm (Built-int Functions)
6. Sum of 2 no. (Recursive Function)
7. Valid Password (Lambda Function)
8. Ascending Order(Built-in Function)
9. Perfect No.(User-defined Function)
10. Area of circle(User-defined Function)
11. No. of Words in string
12. Freq of char in string
13. largest palindrome substring in a given string
14. Valid email address.
15. Index of a given substring
16. Remove all whitespace characters
17. Common characters between two strings.
18. Find second most frequent character
19. Check if string is an anagram of another string.
20. Find first non-repeating character
21. Replace all occurrences of a given word
22. Most frequent word in a sentence
23. Check if given string contains only digits.
24. Remove all the vowels from string
25. Concatenate two strings without '+'
26. Length of the longest word in a sentence.
27. Capitalize the first letter of each word
28. String palindrome?
29. Reverse a string using slicing
30. Count the no.of occurrences of a character
31. Find kth smallest element
32. Difference between two lists
33. Intersection of two lists.
34. Union of two lists

35. Common elements between two lists
36. First n Fibonacci numbers & store them in a list
37. Largest subsequence sum
38. Merge two sorted list
39. Remove all even numbers
40. Find maximum and minimum elements in a list.
41. Shuffle a list.
42. Concatenate two lists
43. Sort in ascending order
44. Frequency of all elements
45. Find index of a element
46. Reverse a given list.
47. Remove duplicates from list
48. Smallest number in a list.
49. Second largest number in list
50. Sum of all elements in list
51. Sort tuple in ascending order.
52. Sum of all elements in tuple.
53. Convert a list of tuples to a list of lists.
54. Check if element exists in a given tuple
55. Convert a tuple to a list.
56. Index of element in tuple.
57. Concatenate two tuples
58. Maximum and minimum elements in a tuple
59. Reverse tuple.
60. Length of a given tuple.

1. *Concatenate 2 lists(Generator function):*

```
def concatenate_lists(list1, list2):
    yield from list1
    yield from list2

list1 = [1, 2, 3]
list2 = [4, 5, 6]

concatenated = concatenate_lists(list1, list2)
```

```
concatenated_list = list(concatenated)
print(concatenated_list)
```

2. ***Length of longest word(Reduce functions):***

```
from functools import reduce

def find_longest_word_length(sentence):
   return reduce(lambda x, y: max(x, len(y)),
sentence.split(), 0)

sentence = "This is an example sentence to find the
longest word length"
result = find_longest_word_length(sentence)
print(f"The length of the longest word is: {result}")
```

3. ***Prime no. (Filter Function):***

```
def prime(num):
   return num > 1 and all(num % i != 0 for i in range(2,
int(num**0.5) + 1))

def check_prime(number):
   return number in filter(prime, range(number))

number = 17
print(f"{number} is {'a prime number.' if
check_prime(number) else 'not a prime number.'}")
```

4. ***Count vowels(Map Function):***

```
str1 = "PRAJIITH"
c= map(str1.lower().count, "aeiou")
print("[A, E, I, O, U]")
print(list(c))
```

5. Lcm (Built-int Functions):

```
import math

num1, num2 = 12, 15
result = math.lcm(num1, num2)
print(f"The LCM of {num1} and {num2} is: {result}")
```

6. Sum of 2 no. (Recursive Function):

```
def recursive_sum(a, b):
    if b == 0:
        return a
    return recursive_sum(a ^ b, (a & b) << 1)

num1, num2 = 5, 3
result = recursive_sum(num1, num2)
print(f"The sum of {num1} and {num2} is: {result}")
```

7. Valid Password (Lambda Function):

```
check= lambda password: \
    len(password) >= 8 and any(c.isupper() for c in
password) \
    and any(c.islower() for c in password) and
any(c.isdigit() for c in password)

password = "Password123"
if check(password):
    print(f"'{password}' is a valid password.")
else:
    print(f"'{password}' is not a valid password.")
```

8. Ascending Order(Built-in Function):

```
numbers = [5, 2, 9, 1, 5, 6]

sorted_numbers = sorted(numbers)

print("Ascending order:", sorted_numbers)
```

9. Perfect No.(User-defined Function):

```
def is_perfect_number(num):
    if num <= 0:
        return False
    return num == sum(i for i in range(1, num) if num % i == 0)

number = 28
if is_perfect_number(number):
    print(f"{number} is a perfect number.")
else:
    print(f"{number} is not a perfect number.")
```

10. Area of circle(User-defined Function):

```
def greet(area):
    pi=3.14
    r=3
    print(f"area is : {pi*r*r}")
greet("area")
```

11. No. of Words in string:

```
str= "vanakkam nanbargale"
words = str.split(" ")
num_words = len(words)
print("Number of words:", num_words)
```

12. Freq of char in string:

```
string = "hello world"

frequency = {char: string.count(char) for char in string}

print("Character frequencies:")
for char, freq in frequency.items():
    print(f"'{char}' : {freq}")
```

13. ***largest palindrome substring in a given string:***

```
input_string = "babad"
n = len(input_string)
max_length = 0
start = 0

for i in range(n):
   for j in range(i, n):
      substring = input_string[i:j+1]
      if substring == substring[::-1] and len(substring) >
max_length:
         max_length = len(substring)
         start = i

largest_palindrome = input_string[start:start +
max_length]
print(f"The largest palindrome substring in
'{input_string}' is '{largest_palindrome}'.")
```

14. ***Valid email address.:***

```
import re

email = "example@example.com"
pattern = r'^[\w\.-]+@[a-zA-Z\d\.-]+\.[a-zA-Z]{2,}$'

if re.match(pattern, email):
   print(f"'{email}' is a valid email address.")
else:
   print(f"'{email}' is not a valid email address.")
```

15. Index of a given substring:

```
main_string = "Hello, world! Welcome to Python."
substring = "world"

index = main_string.find(substring)

if index != -1:
   print(f"The substring '{substring}' is found at index
{index}.")
else:
   print(f"The substring '{substring}' is not found in the
string.")
```

16. Remove all whitespace characters:

```
a="he is a good boy"
p=a.replace(" ",'')
print(p)
```

17. Common characters between two strings:

```
s1 = "hi"
s2 = "hiii"

common= sorted(set(s1) & set(s2) - {' '})

print("The common letters are:", *common)
```

18. Find second most frequent character:

```
string = "aabbbc"

char_counts = {char: string.count(char) for char in
set(string)}

sorted_chars = sorted(char_counts.items(),
key=lambda item: item[1], reverse=True)

if len(sorted_chars) > 1:
    second_most_freq_char = sorted_chars[1][0]
    print(f"2nd most freq char in '{string}' is
'{second_most_freq_char}'.")
else:
    print(f"There is no 2nd most freq charin '{string}'.")
```

19. ***Check if string is an anagram of another string:***

```
str1 = "listen"
str2 = "silent"

anagram = sorted(str1.lower().replace(" ", "")) ==
sorted(str2.lower().replace(" ", ""))

print(f"'{str1}' and '{str2}' are {'anagrams' if anagram
else 'not anagrams'}.")
```

20. ***Find first non-repeating character:***

```
from collections import Counter

def non_repeating_char(s):
    char_counts = Counter(s)
    return next((char for char in s if char_counts[char]
== 1), None)

string = "abacabad"
result = non_repeating_char(string)

if result:
    print(f"The first non-repeating character in
'{string}' is '{result}'.")
else:
    print(f"There is no non-repeating character in
'{string}'.")
```

21. *Replace all occurrences of a given word:*

```
input= "Hello, world! Hello, Python!"
old_word = "Hello"
new_word = "Hi"

result= ' '.join([new_word if word == old_word else word for word in input.split()])

print(f"Original string: '{input}'")
print(f"Modified string: '{result}'")
```

22. *Most frequent word in a sentence:*

```
from collections import Counter

sentence = "This is an example sentence. This sentence is to find the most frequent word."

word_counts = Counter(sentence.split())

most_common_word, frequency = max(word_counts.items(), key=lambda item: item[1])

print(f"The most frequent word is '{most_common_word}' with a frequency of {frequency} time(s).")
```

23. *Check if given string contains only digits:*

```
string = 'trish232'
if string.isdigit():
   print("The string contains only digits.")
else:
   print("The string does not contain only digits.")
```

24. *Remove all the vowels from string:*

```
a="he is a good boy"
vowel=('a','e','i','o','u')
for i in a.lower():
    if i in vowel:
        a=a.replace(i,"")
print(a)
```

25. *Concatenate two strings without '+':*

```
string1 = "Hello, "
string2 = "world!"
concatenated_string = "{}{}".format(string1, string2)
concatenated_string_f = f"{string1}{string2}"
print("Concatenated string (using formatted string):", concatenated_string)
print("Concatenated string (using f-string):", concatenated_string_f)
```

26. *Length of the longest word in a sentence:*

```
sentence = "This is an example sentence to find the longest word length"

max_l= max(len(word) for word in sentence.split())

print(f"The length of the longest word is: {max_l}")
```

27. *Capitalize the first letter of each word*:

```
str = "hi im trisha."

capitalized=str.title()

print("Original string:", str)
print("Capitalized string:", capitalized)
```

28. String palindrome? :

```
str="helleh"
reverse=str[::-1]
if str==reverse:
  print("palindrome")
else:
  print("not palindrome")
```

29. Reverse a string using slicing:

```
str="hello"
reverse=str[::-1]
print(reverse)
```

30. Count the no.of occurrences of a character:

```
count = 0
my_string = "hello"
my_char = 'l'
for i in my_string:
    if i == my_char:
        count += 1
print(count)
```

31. Find kth smallest element :

```
n = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
k = 3
n.sort()
print("smallest kth element is:",n[k - 1])
```

32. Difference between two lists:

```
list1=[1,2,3,4,5,7]
list2=[1,24,3,4,5,6]
set1=set(list1)
set2=set(list2)
difference=set1-set2
list3=list(difference)
print(list3)
```

33. *Intersection of two lists:*

```
list1=[1,2,3,4,5,7]
list2=[1,24,3,4,5,6]
set1=set(list1)
set2=set(list2)
difference=set1-set2
list3=list(difference)
print(list3)
```

34. *Union of two lists:*

```
list1=[6,7,8,9,10,4]
list2=[4,5,6,7,8]
set1=set(list1)
set2=set(list2)
union=set1.union(set2)
list3=list(union)
print("union",list3)
```

35. *Common elements between two lists:*

```
list1=[1,2,3,4,5]
list2=[4,5,6,7,8]
set1=set(list1)
set2=set(list2)
common=set1&set2
list3=list(common)
print("common",list3)
```

36. *First n Fibonacci numbers & store them in a list:*

```
a,b=0,1
fibo_l=[a,b]
for i in range(50):
  a,b=b,a+b
  fibo_l.append(b)
print(fibo_l)
```

37. ***Largest subsequence sum:***

```
nums = [3,-1,4,-1,5,-9,2,6,-5,3,5]
max_sum = current_sum = float('-inf')
for num in nums:
   current_sum = max(num, current_sum + num)
   max_sum = max(max_sum, current_sum)

print("The largest subsequence sum is:", max_sum)
```

38. ***Merge two sorted list:***

```
list1=[1,13,5,7]
list2=[2,4,6,6,8]
merge=sorted(list1+list2)
print(merge)
```

39. ***Remove all even numbers:***

```
list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
list = [x for x in list if x % 2 != 0]
print("List with odd numbers only:", list)
```

40. ***Find maximum and minimum elements in a list:***

```
list=[1,2,3,4,5]
print("Max:",max(list))
print("Min:",min(list))
```

41. ***Shuffle a list:***

```
import random
list=[1,2,3,4,5]
random.shuffle(list)
print(list)
```

42. ***Concatenate two lists:***

```
list1=[2,3,4]
list2=[6,5,4]
concatenate=list1+list2
print("Concatenated",concatenate)
```

43. *Sort in ascending order:*

```
list=[2,1,7,6,4]
n = len(list)
for i in range(n - 1):
 for j in range(0,n-i-1):
   if list[j] > list[j + 1]:
    list[j],list[j+1]=list[j+1],list[j]
print(list)
```

44. *Frequency of all elements:*

```
list = [1, 2, 3, 4, 5, 2, 3, 4, 2, 2, 1, 1, 5]
freq = {item: list.count(item) for item in list}

print("Element freq:")
for key, value in freq.items():
    print(f"{key}: {value}")
```

45. *Find index of a element:*

```
list1 = [8, 5, 6, 1, 2]
index_ele=list1.index(6)
print(f"element {6} is in index :{index_ele}")
```

46. *Reverse a given list:*

```
list=[1,2,3,4]
list.reverse()
print(list)
```

47. *Remove duplicates from list:*

```
l=[1,3,2,1,4,5,4]
list2=list(set(l))
print(list2)
```

48. Smallest number in a list:

```
list = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
smallest = list[0]
for num in list:
   if num < smallest:
      smallest = num

print("The smallest number in the list is:", smallest)
```

49. Second largest number in list:

```
list=[4,2,1,7,6,8]
list.sort()
print(list[-2])
```

50. Sum of all elements in list:

```
list=[1,2,3,4,5,6]
sum=sum(list)
print(sum)
```

51. Sort tuple in ascending order:

```
g_tuple = (3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5)
sorted_l= sorted(g_tuple)
sorted_t =tuple(sorted_l)
print("Original tuple:", tuple)
print("Sorted tuple in ascending order:", sorted_t)
```

52. Sum of all elements in tuple:

```
tuple = (1, 2, 3, 4, 5)
sum = 0
for num in tuple:
   sum += num

print("The sum of all elements in the tuple is:",sum)
```

53. Convert a list of tuples to a list of lists:

```
list_of_tuples = [(1, 2), (3, 4), (5, 6)]
list_of_lists = [list(tup) for tup in list_of_tuples]
print("List of tuples:", list_of_tuples)
print("Converted to list of lists:", list_of_lists)
```

54. Check if element exists in a given tuple:

```
tuple = (1, 2, 3, 4, 5)
element= 3
if element in tuple:
   print(f"The element {element} exists in the tuple.")
else:
   print(f"The element {element} does not exist in the tuple.")
```

55. Convert a tuple to a list:

```
tuple=(1,2,3,4)
list_t=list(tuple)
print(list_t)
```

56. Index of element in tuple:

```
tuple= (1, 2, 3, 4, 5, 6)
index = tuple.index(6)
print(index)
```

57. Concatenate two tuples:

```
tuple1=(1,2,3,4)
tuple2=(5,6,7,8)
concatenate=tuple1+tuple2
print(concatenate)
```

58. Maximum and minimum elements in a tuple:

```
tuple=(1,2,3,4,5,0)
max_t=max(tuple)
min_t=min(tuple)
print(max_t,min_t)
```

59. Reverse tuple:

```
tuple = (1, 2, 3, 4, 5)
tuple_rev = tuple[::-1]
print(tuple_rev)
```

60. Length of a given tuple:

```
tuple=(1,2,3,4,5,6,7)
length=len(tuple)
print(length)
```

LEVEL 3:

1. Convert a dictionary to a list of tuples.
2. Sort dictionary by its values.
3. Frequency of all elements in dictionary
4. Find keys with maxi & min values in dictionary.
5. Find maxi & min values in dictionary.
6. Remove key from a dictionary.
7. Value exists in a dictionary?
8. Concatenate two dictionaries.
9. Key exists in dictionary?
10. Find length of dictionary.
11. Write a list of strings to a file.
12. Copy contents of one file to another.
13. Count no. of characters in a file.
14. Count no. of lines in a file.
15. Count no. of words in a file.
16. Read a file & display its contents.
17. Raise a custom exception.
18. Handle multiple exceptions using single except block
19. Handle a NameError exception.
20. Handle a KeyError exception.
21. Handle a FileNotFoundError exception.
22. Handle a ValueError exception.
23. Handle a TypeError exception.
24. Handle a IndexError exception
25. Handle a ZeroDivisionError exception.

1. *Convert a dictionary to a list of tuples:*

```
dict={'age':20,'date':3}
list_t=list(dict.items())
print(list_t)
```

2. Sort dictionary by its values:

```
my_dict = {'two': 2, 'three': 3, 'four': 4, 'one': 1}
sorted_dict = dict(sorted(my_dict.items(), key=lambda
item: item[1]))

print("Sorted dictionary by values:", sorted_dict)
```

3. Frequency of all elements in dictionary:

```
from collections import Counter
dict = {'a': 10, 'b': 5, 'c': 10, 'd': 8, 'e': 10}
frequency = Counter(dict.values())

for value, count in frequency.items():
   print(f"Value '{value}' occurs {count} time(s) in the
dictionary.")
healthiest
```

4. Find keys with maxi & min values in dictionary:

```
dict = {'a': 10, 'b': 5, 'c': 20, 'd': 8}
max = max(dict, key=dict.get)
min = min(dict, key=dict.get)

print("Key with max value:", max, "and value:",
dict[max])
print("Key with min value:", min, "and value:",
dict[min])
```

5. Find maxi & min values in dictionary:

```
dict={'two':2,'three':3,'four':9}
min=min(dict.values())
max=max(dict.values())
print(min,max)
```

6. *Remove key from a dictionary:*

```
dict = {'a': 1, 'b': 2, 'c': 3}
remove = 'b'
if remove in dict:
    del dict[remove]
    print(f"Key '{remove}' removed successfully.")
else:
    print(f"Key '{remove}' not found in the dictionary.")

print("Updated dictionary:", dict)
```

7. *Value exists in a dictionary?:*

```
dict={'time':9,'color':5}
search=9
if search in dict.values():
  print("yes".format(search))
else:
  print("no".format(search))
```

8. *Concatenate two dictionaries:*

```
dict1={'hi':3,'my':5}
dict2={'name':7,'is':8,'trisha':2}
dict1.update(dict2)
print(dict1)
```

9. *Key exists in dictionary?:*

```
dict={'time':9,'color':5}
search='place'
if search in dict:
  print("yes".format(search))
else:
  print("no".format(search))
```

10. *Find length of dictionary:*

```
dict={'age':20,'date':3}
length=len(dict)
print(length)
```

11. *Write a list of strings to a file:*

```
list_of_strings = [
    "Hello, World!",
    "This is a test.",
    "Writing to a file in Python."
]

file_name = "output.txt"

try:
    with open(file_name, 'w') as file:
        file.writelines('\n'.join(list_of_strings))
    print(f"List of strings has been written to
'{file_name}' successfully.")
except IOError:
    print(f"Error writing to '{file_name}'.")
```

12. *Copy contents of one file to another:*

13. *Count no. of characters in a file.*

14. *Count no. of lines in a file.*

15. *Count no. of words in a file.*

16. *Read a file & display its contents.*

17. *Raise a custom exception:*

```
class CustomException(Exception):
    def ___init___(self, message):
        self.message = message

try:
    num = 12038
    if num <= 100:
        raise CustomException("Number should be greater
than 100")
    else:
        print(f"Entered number {num} is valid")
except CustomException as e:
    print(f"Custom Exception caught: {e.message}")
except ValueError:
    print("Invalid input. Please enter a valid number.")
```

18. Handle multiple exceptions using a single except block:

```
try:
    num1 = 2
    num2 = 3

    result = num1 / num2

    my_list = [1, 2, 3]
    print(my_list[5])

except (ValueError, ZeroDivisionError, IndexError) as
e:
    if isinstance(e, ValueError):
        print("Invalid input or not a number.")
    elif isinstance(e, ZeroDivisionError):
        print("Cannot divide by zero.")
    else:
        print("Index is out of range for the list.")
```

19. Handle a NameError exception:

```
undefined_variable = None
```

```
try:
   print(undefined_variable)
except NameError as e:
   print(f"NameError occurred: {e}")
   print("The variable 'undefined_variable' is not
defined or initialized.")
```

20. ***Handle a KeyError exception:***

```
my_dict = {'name': 'Alice', 'age': 30, 'city': 'New York'}

try:
   print(my_dict['occupation'])
except KeyError as e:
   print(f"KeyError occurred: {e}")
   print("The key 'occupation' does not exist in the
dictionary.")
```

21. ***Handle a FileNotFoundError exception:***

```
file_name = "nonexistent_file.txt"

try:
   with open(file_name, 'r') as file:
      contents = file.read()
   print(f"Contents of '{file_name}':")
   print(contents)
except FileNotFoundError:
   print(f"Error: File '{file_name}' not found.")
except IOError as e:
   print(f"Error: {e}")
```

22. ***Handle a ValueError exception:***

```
try:
   num = 100
   result = 10 / num
   print(f"Result of 10 divided by {num} is: {result}")
```

```
except ValueError:
    print("Error: Invalid input. Please enter a valid
number.")
except ZeroDivisionError:
    print("Error: Cannot divide by zero.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

23. ***Handle a TypeError exception:***

```
try:
    num = 250
    result = 10 / int(num)
    print(f"Result of 10 divided by {num} is: {result}")
except TypeError:
    print("Error: Unsupported operation. Check the types
of operands.")
except ValueError:
    print("Error: Invalid input. Please enter a valid
number.")
except ZeroDivisionError:
    print("Error: Cannot divide by zero.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

24. ***Handle a IndexError exception:***

```
try:
    my_list = [1, 2, 3]
    index = 245
    value = my_list[index]
    print(f"Value at index {index}: {value}")
except IndexError:
    print(f"Error: Index out of range. Please enter a valid
index.")
except ValueError:
    print(f"Error: Invalid input. Please enter a valid integer.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

25. Handle a ZeroDivisionError exception:

```
try:
    num1 = 0
    num2 = 2

    result = num1 / num2

    print(f"Result of {num1} divided by {num2} is: {result}")
except ZeroDivisionError:
    print("Error: Division by zero is not allowed. Please enter a non-zero denominator.")
except ValueError:
    print("Error: Invalid input. Please enter valid integers.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```