==========EASY==========

1.

Write a function that prints the concatenation of the strings of all the even numbers within 0 to 20.

Function Call:

even_string()

Output:

02468101214161820

2.

Write a function that takes an integer as parameter and adds the sum of all the divisors of that number.

Function Call:

div_sum(12)

Output:

28

3.

Write a function that takes an integer and a string from the user and returns

the multilication of the string with the integer.

Function Call:

print(str_mul("Hi", 5))

Output:

HiHiHiHiHi

4.

Write a function that takes an integer and a list of integers, and returns a list containing all the

numbers that are grater or equal to that number.

Function Call #1:

print(listchecker([10,2,-5,5,4,6], 5))

Output:

[10,5,6]

Function Call #2:

print(listchecker([10,2,-5,5,4,6], 15))

Output:

[ ]

5.

Write a function lower_vowel_extractor that takes a string, extracts all the unique lower-case vowels and appends them in a list. Then returns the list. You cannot use isLower() or isUpper() built-in functions.

One lower-case vowel must not occur twice in the returned list.

Function Call 1:

print(lower_vowel_extractor("Hey wait, what is up?"))

Sample Output 1:

["e", "a", "i", "u"]

Explanation 1:

Here the lower_vowel_extractor function takes the string "Hey wait, what is up?" as a parameter. In this string, the lower-case vowels are "e", "a", "i", "a", "i" and "u". The second "a" and "i" are ignored to avoid repetition. Therefore, "e", "a", "i" and "u" are appended in a list and returned.

Function Call 2:

print(lower_vowel_extractor("YEEEEEHAAAWW!!!!"))

Sample Output 2:

[]

Explanation 2:

As there are vowels in the string taken by the function, but none of these are lower-case.

Therefore, an empty list is returned.

6.

Write a function that takes a list and an integer and returns whether the summation of all th elements of the list is grater than the number or not.

Function call #1:

print(comparator([10,15,5,8,13],60))

Output:

Summation is smaller than 60

Function call #2:

print(comparator([10,15,5,8,13],40))

Output:

Summation is grater than 40

========== MEDIUM ==========

7.

#Trace the following code, show tracing table and write the outputs:

```
def methodA(a,b):
  print("This is method A")
  if (a%3==b or a%4>b):
    print(str(a)+str(b))
  else:
    print(str(a-b))
```

```
  methodB(a-b)
  return a+b

def methodB(c):
  print("This is method B")
  if c>0 and c%2==0:
    print(c//2)
  elif c<0 and c%2==0:
    print(c*-1//2)
  else:
    print(c)

for i in range(5, 3, -1):
  print(methodA(i,i-3))
methodB(4.5)
```

8.

#Trace the following code, show tracing table and write the outputs:

```
def method_1(w):
  z=w
  for i in range(3):
    print(z//3)
    z//=3

def method_2(x,y):
  z=y
  for i in x:
    z+= len(x)
```

```
    print(z)
    z//=3
  method_1(len(x)*z)
  return len(x)*z,y

out= method_2([10,20,30],25)
print(out[1])
method_1(12)
```

9.

Write a python function called product_of_list that takes a dictionary as a parameter.

You can pass the dictionary in the function call directly without taking any input from the user.

Inside the function, for each "key: value" pair of the dictionary, you need to take the ASCII values of the two characters from the list in the "value" part of the dictionary. Then multiply those two values. All the multiplied values should be kept in a tuple in sequential order of the dictionary "key: value" pairs. Finally, return the tuple in the function call and print the returned tuple.

Function Call:

```
product_of_list({'C':['g','m'], 'B':['b','z'], 'A':['a','c']})
```

Sample Output:

```
(11227, 11956, 9603)
```

Explanation:

For key 'C', the values are 'g' and 'm'. ASCII values of 'g' = 103 and 'm' = 109. So, the multiplied value is 103*109 = 11227.

For key 'B', the values are 'b' and 'z'. ASCII values of 'b' = 98 and 'z' = 122. So, the multiplied value is 98*122 = 11956.

For key 'A', the values are 'a' and 'c'. ASCII values of 'a' = 97 and 'c' = 99. So, the multiplied value is 97*99 = 9603.

Finally, a tuple is returned to the function call consisting of the multiplied values in the given order, i.e. (11227, 11956, 9603).

10.

Write a Python Function that will take Student assignment names in a single line.

Now create and return a dictionary from the given assignment names that will hold the IDs

in separate keys based on the sections. First 2 digits of the assignment name denotes the section.

Function Call 1:

myFunction(“13_19181820 18_19104354 20_21101457 13_19103372 01_18301021”)

Sample Output1: {13: [19181820, 19103372], 18: [19104354], 20: [21101457], 1: [18301021]}

========== HARD ==========

11.

Write a function named ascii_summation that finds the total ascii values of all the words in a list,

and returns a dictionary with the highest and lowest ascii_values as keys and their corresponding words as values.

Function Call:

print(ascii_summation(["Disorder", "Chaos", "Anarchy", "Crime", "Lawless"]))

Output:

{828: "Disorder", 494:"Chaos"}

12.

Write a python function that takes a string as an argument and searches all the vowels having at least

two occurrences in that string. Then, prepare and RETURN a dictionary where the keys will be the vowels

occuring at least two times and values will be the list of the positions of those characters in the string.

(You are not allowed to use builtin .count() function here)

Function Call:

print(function_name("hello hi how are you"))

Output:

{'e': [1,15], 'o': [4,10,18]}

13.

Write a function that-

a. Takes a string of student names from the user and converts it to a list.

b. Takes the numbers of each student from the user in a for loop and calculates the total marks.

Returns a dictionary with student names as keys and corresponding marks as values

Function Call:

print(marks())

Sample Input:

Jay Ray May

20 15 16 19 10

18 15 15 15 20

20 20 20 20 19

Output:

{"Jay": 80, "Ray": 83, "May": 99}

14.

Assume you work as a cashier in a restaurant. You must keep track of the amount of money your restaurant earns each day and report it to your boss. If a customer purchases something, that amount is added to the daily income book.

Take a customer's order and if a customer enters an "X," the order is complete, and print "Total cost of the items: ". In this way, 3 customers can order at maximum as your restaurant only serves three customers per day; after the service, print the day's total income; if the total money exceeds 500tk, print "Great Work". Otherwise, print "Do Best Next Day."

Note: Write a function called “take_order()” that asks users to input food names, calculate the amount and show it to the customer. When the last customer buys, you need to also calculate the total income of the day and show it at the end. Assume that no customers enter invalid food names.

The Dictionary ‘price_list’ is set for you and use it in the code for your convenience.

price_list={'Juice':30, 'Fried Rice':150, 'Burger':50, 'Pizza':110, 'Fried Chicken':90, 'Noodles':70}

==========

Function Call 01:

take_order()

Sample Input_01:

Please enter the name of the items:

Burger

Juice

Pizza

X

Sample Output_01:

Total cost: 190

Explanation 1 : The customer ordered 3 things: Burger, Juice and a Pizza. So the cost of the three things are added, 50+30+110= 190 and is presented as the output.

==========

Function Call 02:

take_order()

Sample Input_02:

Please enter the name of the items:

Burger

Fried Rice

Pizza

X

Sample Output_02:

Total cost: 310