9/9/2023

Python Practical File

Full Time Diploma in Computer Engineering
5th Semester

Abhishek Roka
10621019

INDEX

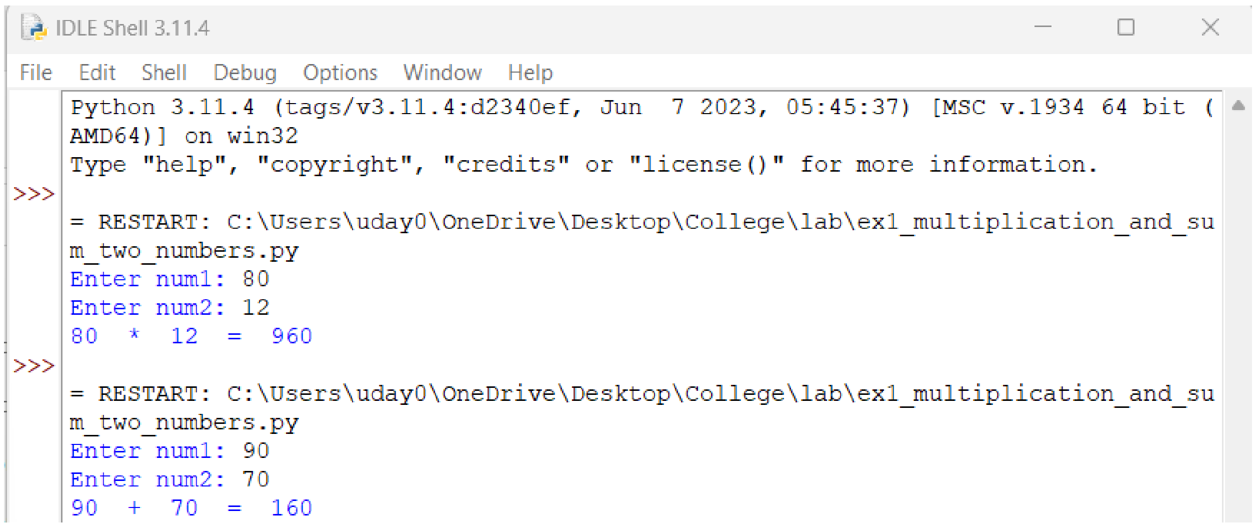
Practical 1

Calculate the multiplication and sum of two numbers

Code

```
num1 = int(input("Enter num1: "))
num2= int(input("Enter num2: "))

mul = num1*num2
if mul <= 1000:
   print(num1, " * ", num2, " = ", mul)
else:
   print(num1, " + ", num2, " = ", num1+num2)
```

Output:



```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex1_multiplication_and_su
m_two_numbers.py
Enter num1: 80
Enter num2: 12
80  *  12  =  960
>>>
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex1_multiplication_and_su
m_two_numbers.py
Enter num1: 90
Enter num2: 70
90  +  70  =  160
```

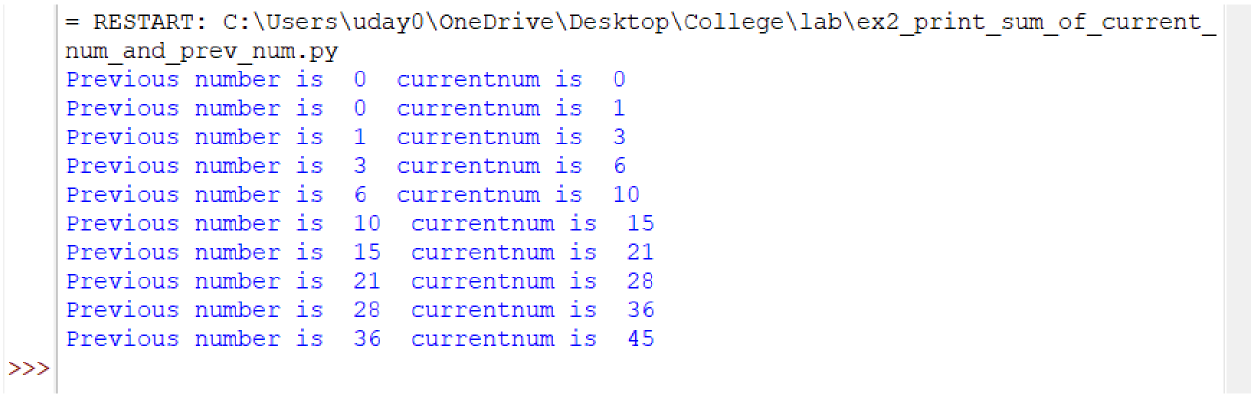
Practical 2

Print the sum of the current number and the previous number

Code:

```
prev_num = 0
rangelen = range(10)
currentnum=0
for i in rangelen:
    currentnum = prev_num + i
    print("Previous number is ", prev_num, " currentnum is ", currentnum)
    prev_num = currentnum
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex2_print_sum_of_current_
num_and_prev_num.py
Previous number is  0  currentnum is  0
Previous number is  0  currentnum is  1
Previous number is  1  currentnum is  3
Previous number is  3  currentnum is  6
Previous number is  6  currentnum is  10
Previous number is  10  currentnum is  15
Previous number is  15  currentnum is  21
Previous number is  21  currentnum is  28
Previous number is  28  currentnum is  36
Previous number is  36  currentnum is  45
>>>
```

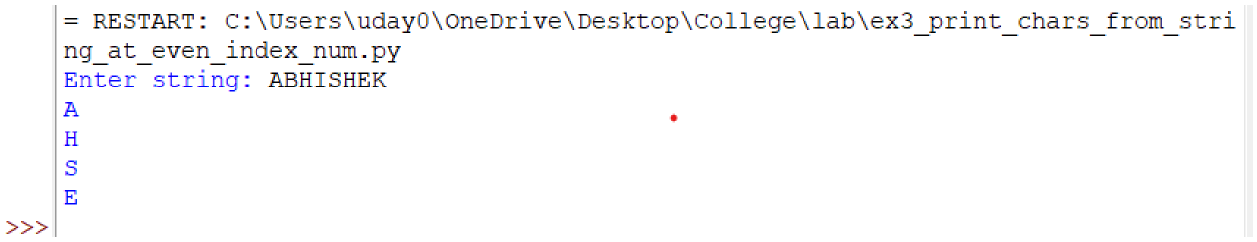
Practical 3

Print characters from a string that are present at an even index number

Code:

```
user_string = input("Enter string: ")

length = len(user_string)

range_len = range(0,length,2)

for i in range_len:

    print(user_string[i])
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex3_print_chars_from_stri
ng_at_even_index_num.py
Enter string: ABHISHEK
A
H
S
E
>>>
```

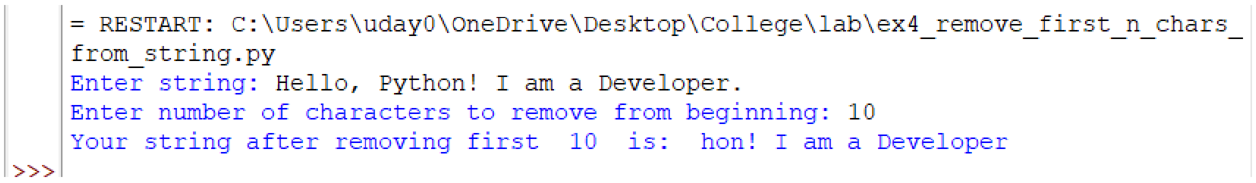
Practical 4

Remove first n characters from a string

Code:

```
user_string = input("Enter string: ")

num_of_chars = int(input("Enter number of characters to remove from beginning: "))

new_string = user_string[num_of_chars:-1]

print("Your string after removing first ", num_of_chars, " is: ", new_string)
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex4_remove_first_n_chars_
from_string.py
Enter string: Hello, Python! I am a Developer.
Enter number of characters to remove from beginning: 10
Your string after removing first  10  is:  hon! I am a Developer
>>>
```

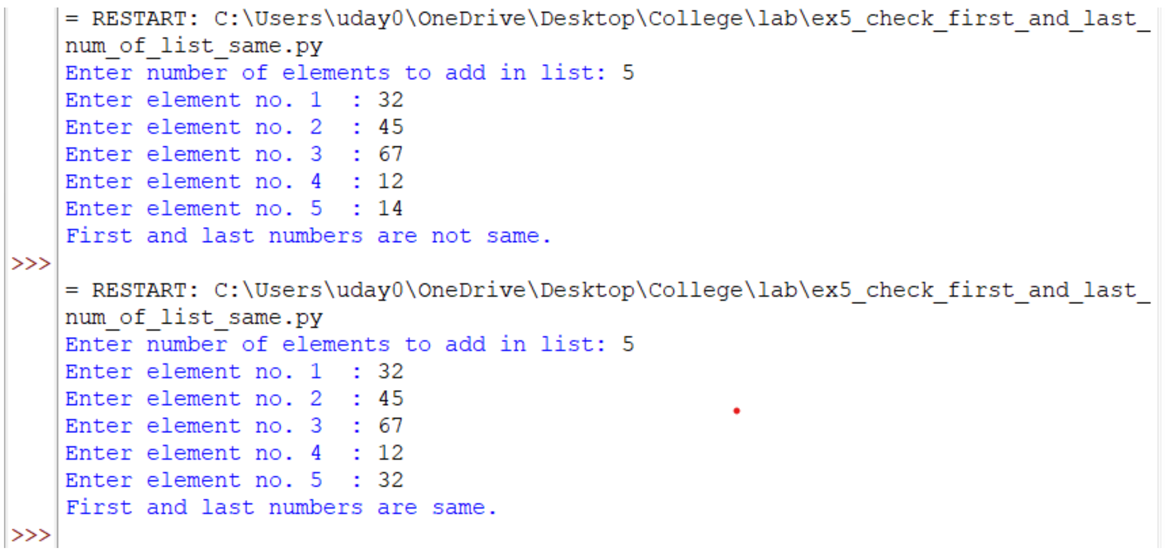
Practical 5

Check if the first and last number of a list is the same

Code:

```
num = int(input("Enter number of elements to add in list: "))
rangeLen = range(num)
user_list = []
for i in rangeLen:
    print("Enter element no.", i+1, " : ", end="")
    user_num = int(input())
    user_list.append(user_num)
if user_list[0] == user_list[-1]:
    print("First and last numbers are same.")
else:
    print("First and last numbers are not same.")
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex5_check_first_and_last_
num_of_list_same.py
Enter number of elements to add in list: 5
Enter element no. 1  : 32
Enter element no. 2  : 45
Enter element no. 3  : 67
Enter element no. 4  : 12
Enter element no. 5  : 14
First and last numbers are not same.
>>>
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex5_check_first_and_last_
num_of_list_same.py
Enter number of elements to add in list: 5
Enter element no. 1  : 32
Enter element no. 2  : 45
Enter element no. 3  : 67
Enter element no. 4  : 12
Enter element no. 5  : 32
First and last numbers are same.
>>>
```

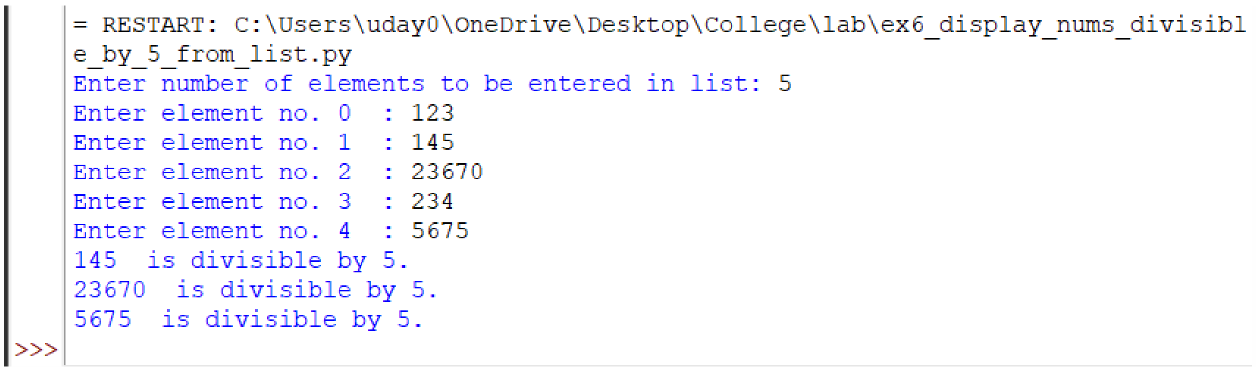
Practiaal 6

Display numbers divisible by 5 from a list

Code:

```
user_list = []
length = int(input("Enter number of elements to be entered in list: "))
rangeLen = range(length)
for i in rangeLen:
    print("Enter element no.", i, " : ", end="")
    num = int(input())
    user_list.append(num)


for i in user_list:
    if i%5==0:
        print(i," is divisible by 5.")
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex6_display_nums_divisibl
e_by_5_from_list.py
Enter number of elements to be entered in list: 5
Enter element no. 0  : 123
Enter element no. 1  : 145
Enter element no. 2  : 23670
Enter element no. 3  : 234
Enter element no. 4  : 5675
145  is divisible by 5.
23670  is divisible by 5.
5675  is divisible by 5.
>>>
```

Practiacal 7

Return the count of a given substring from a string

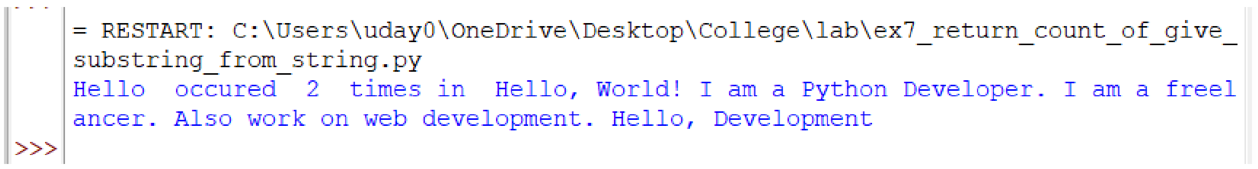
Code:

string_to_examin = "Hello, World! I am a Python Developer. I am a freelancer. Also work on web development. Hello, Development"

to_find_string = "Hello"

print(to_find_string, " occured ", string_to_examin.count(to_find_string), " times in ", string_to_examin)

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex7_return_count_of_give_
substring_from_string.py
Hello  occured  2  times in  Hello, World! I am a Python Developer. I am a freel
ancer. Also work on web development. Hello, Development
>>>
```

Practical 8

Print the following pattern

1

2 2

3 3 3

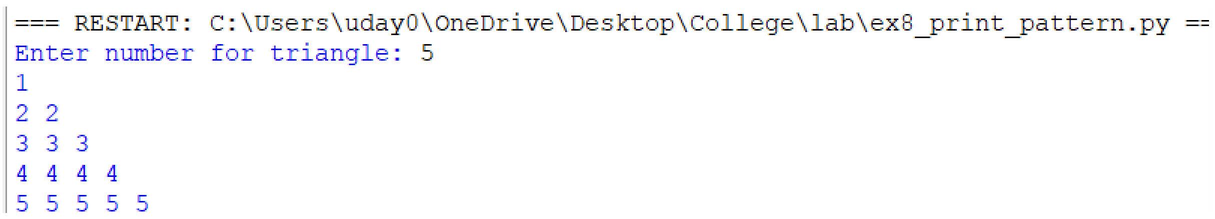
4 4 4 4

5 5 5 5 5

Code:

```
num= int(input("Enter number for triangle: "))
heightRangeLen = range(num)
for i in heightRangeLen:
    lengthRangelen = range(i+1)
    for j in lengthRangelen:
        print(i+1, end=" ")
    print()
```

Output:



```
=== RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex8_print_pattern.py ==
Enter number for triangle: 5
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

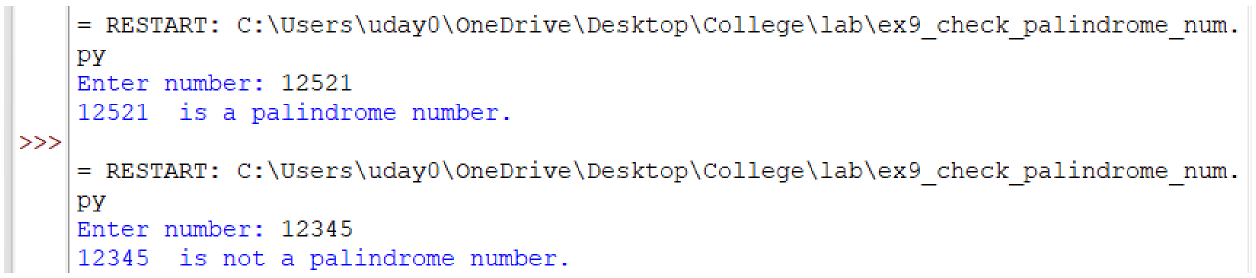
Practical 9

Check Palindrome Number

Code:

```
user_num = input("Enter number: ")


if user_num == user_num[::-1]:
    print(user_num, " is a palindrome number.")
else:
    print(user_num," is not a palindrome number.")
```

Output:



```
   = RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex9_check_palindrome_num.
   py
   Enter number: 12521
   12521  is a palindrome number.
>>>
   = RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex9_check_palindrome_num.
   py
   Enter number: 12345
   12345  is not a palindrome number.
```

Practical 10

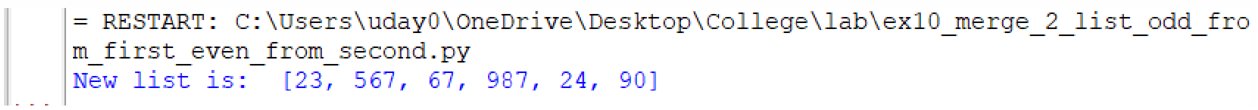
Create a new list from a two list using the following condition

- new list should contain odd numbers from the first list and even numbers from the second list.

Code:

```
list1= [23,44,567,67,88,987]

list2 = [24, 45, 89, 90, 33]

new_list = []

for i in list1:

    if i%2==1:

        new_list.append(i)


for i in list2:

    if i%2==0:

        new_list.append(i)


print("New list is: ", new_list)
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex10_merge_2_list_odd_fro
m_first_even_from_second.py
New list is:  [23, 567, 67, 987, 24, 90]
```

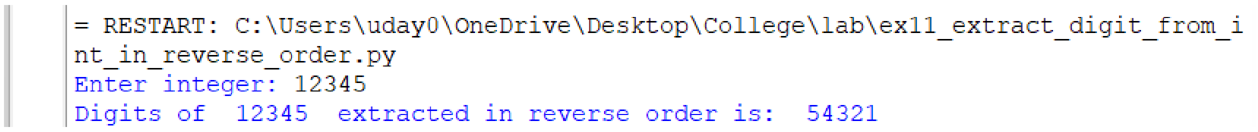
Practical 11

Write a Program to extract each digit from an integer in the reverse order.

Code:

```
num = int(input("Enter integer: "))
new_num=0
store_orginal_num = num
while num>0:
    new_num = new_num*10 + num%10
    num = num // 10
print("Digits of ", store_orginal_num, " extracted in reverse order is: ", new_num)
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex11_extract_digit_from_i
nt_in_reverse_order.py
Enter integer: 12345
Digits of  12345  extracted in reverse order is:  54321
```

Practical 12

Calculate income tax for the given income by adhering to the below rules

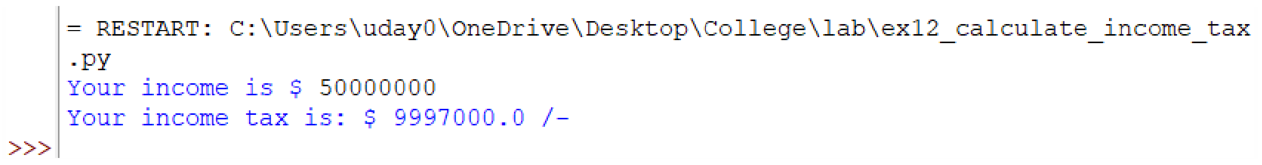
Taxable Income	Rate (in %)
First $10,000	0
Next $10,000	10
The remaining	20

Code:

```
"""

Taxable income          Rate

10,000                  0

next 10,000              10

remaining               20

"""


income_tax = 0

your_income = float(input("Your income is $ "))

income_division = []

if your_income<10000:

    income_tax=0

elif your_income < 20000:

    income_division.append(10000)

    income_division.append(your_income-10000)
```

```
else:
    income_division.append(10000)
    income_division.append(10000)
    income_division.append(your_income-20000)

length = len(income_division)
rangeLen = range(length)
for i in rangeLen:
    if i == 1:
        income_tax += income_division[i]*0.1
    elif i > 1:
        income_tax += income_division[i]*0.2
print("Your income tax is: $", income_tax, "/-")
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex12_calculate_income_tax
.py
Your income is $ 50000000
Your income tax is: $ 9997000.0 /-
>>>
```

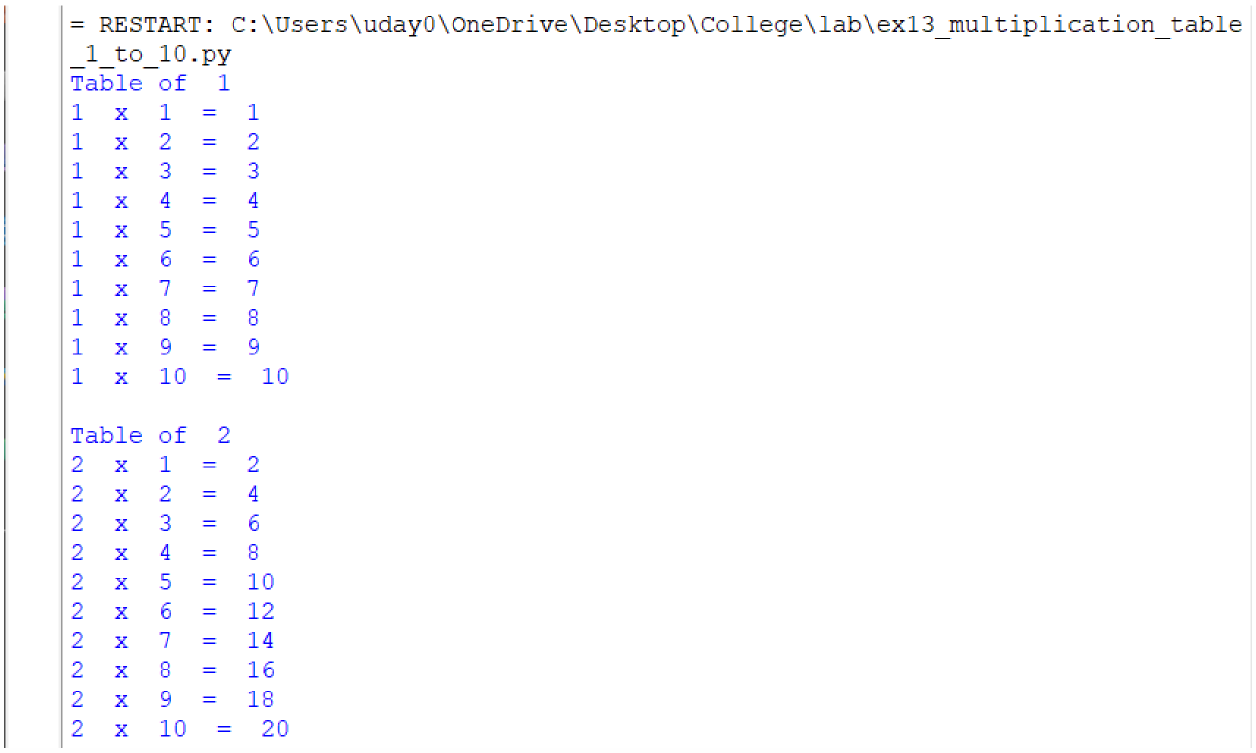
Practical 13

Print multiplication table form 1 to 10

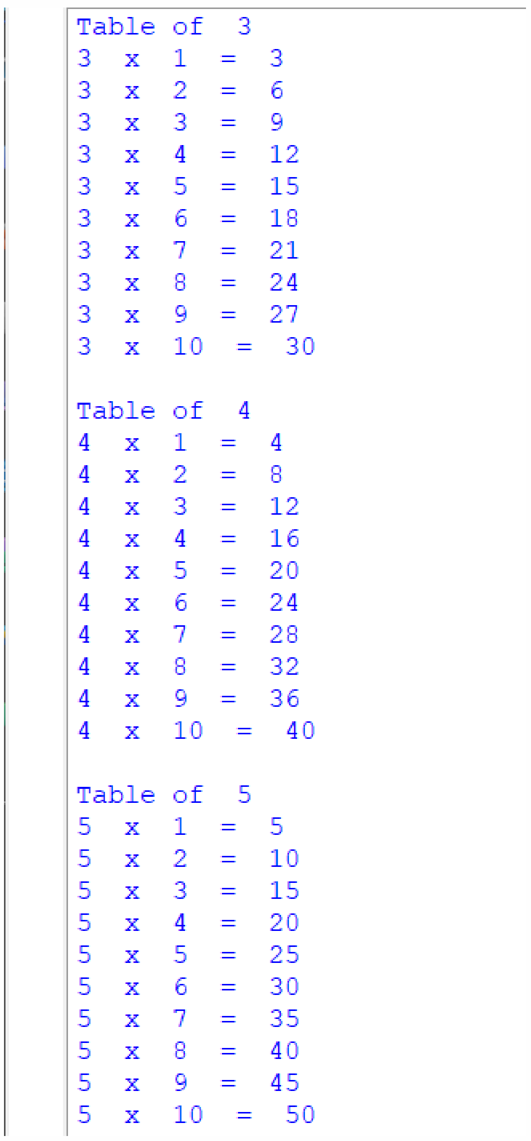
Code:

```
table_of = range(1,11)
table_numbers = range(1,11)
for i in table_of:
    print("Table of ", i)
    for j in table_numbers:
        print(i, " x ", j, " = ", i*j)
    print()
```

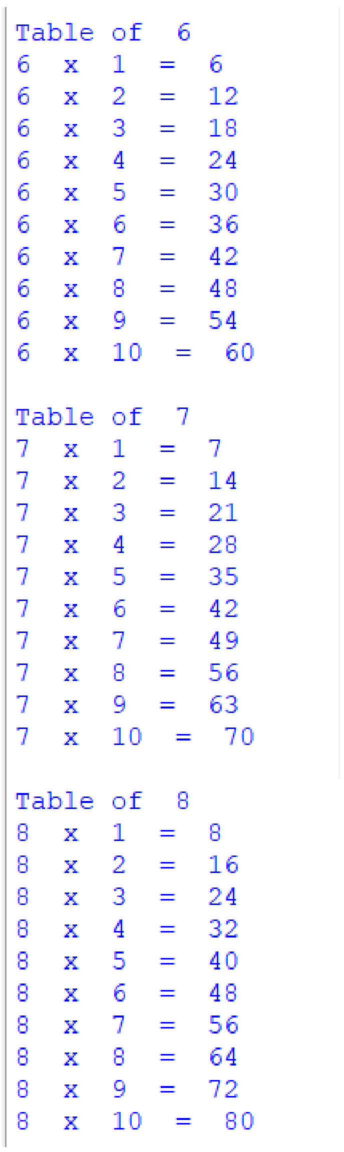
Output:



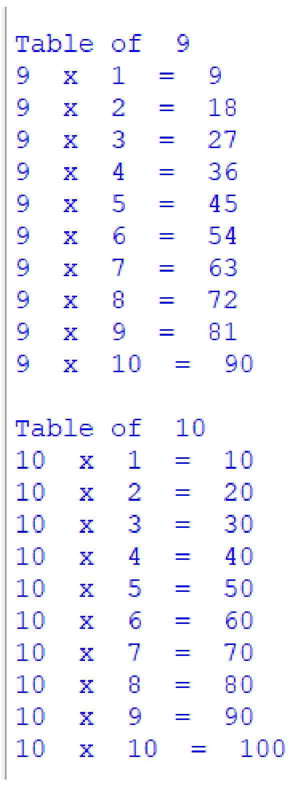
```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex13_multiplication_table
_1_to_10.py
Table of  1
1  x  1  =  1
1  x  2  =  2
1  x  3  =  3
1  x  4  =  4
1  x  5  =  5
1  x  6  =  6
1  x  7  =  7
1  x  8  =  8
1  x  9  =  9
1  x  10  =  10

Table of  2
2  x  1  =  2
2  x  2  =  4
2  x  3  =  6
2  x  4  =  8
2  x  5  =  10
2  x  6  =  12
2  x  7  =  14
2  x  8  =  16
2  x  9  =  18
2  x  10  =  20
```



```
Table of  3
3  x  1  =  3
3  x  2  =  6
3  x  3  =  9
3  x  4  =  12
3  x  5  =  15
3  x  6  =  18
3  x  7  =  21
3  x  8  =  24
3  x  9  =  27
3  x  10  =  30

Table of  4
4  x  1  =  4
4  x  2  =  8
4  x  3  =  12
4  x  4  =  16
4  x  5  =  20
4  x  6  =  24
4  x  7  =  28
4  x  8  =  32
4  x  9  =  36
4  x  10  =  40

Table of  5
5  x  1  =  5
5  x  2  =  10
5  x  3  =  15
5  x  4  =  20
5  x  5  =  25
5  x  6  =  30
5  x  7  =  35
5  x  8  =  40
5  x  9  =  45
5  x  10  =  50
```



```
Table of  6
6  x  1  =  6
6  x  2  =  12
6  x  3  =  18
6  x  4  =  24
6  x  5  =  30
6  x  6  =  36
6  x  7  =  42
6  x  8  =  48
6  x  9  =  54
6  x  10  =  60

Table of  7
7  x  1  =  7
7  x  2  =  14
7  x  3  =  21
7  x  4  =  28
7  x  5  =  35
7  x  6  =  42
7  x  7  =  49
7  x  8  =  56
7  x  9  =  63
7  x  10  =  70

Table of  8
8  x  1  =  8
8  x  2  =  16
8  x  3  =  24
8  x  4  =  32
8  x  5  =  40
8  x  6  =  48
8  x  7  =  56
8  x  8  =  64
8  x  9  =  72
8  x  10  =  80
```



```
Table of  9
9  x  1  =  9
9  x  2  =  18
9  x  3  =  27
9  x  4  =  36
9  x  5  =  45
9  x  6  =  54
9  x  7  =  63
9  x  8  =  72
9  x  9  =  81
9  x  10  =  90

Table of  10
10  x  1  =  10
10  x  2  =  20
10  x  3  =  30
10  x  4  =  40
10  x  5  =  50
10  x  6  =  60
10  x  7  =  70
10  x  8  =  80
10  x  9  =  90
10  x  10  =  100
```

Practical 14

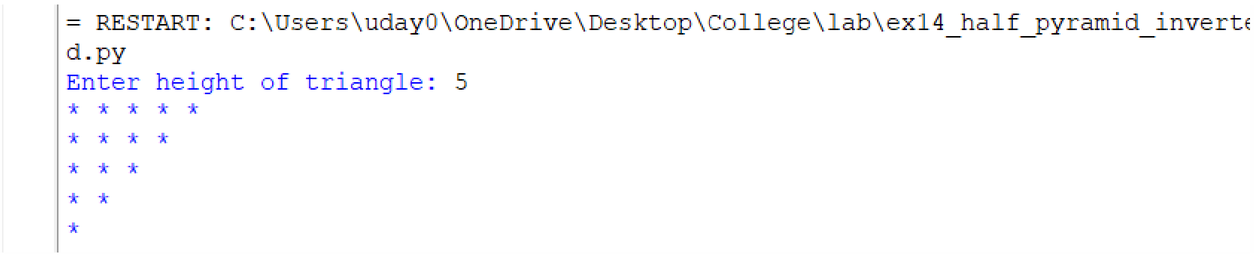
Print downward Half-Pyramid Pattern with Star (asterisk)

```
* * * * *

* * * *

* * *

* *

*
```

Code:

```
length_of_triangle_height = int(input("Enter height of triangle: "))
order_of_height = range(length_of_triangle_height,0,-1)
for i in order_of_height:
    elements_range = range(i)
    for j in elements_range:
        print("*",end=" ")
    print()
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex14_half_pyramid_inverte
d.py
Enter height of triangle: 5
* * * * *
* * * *
* * *
* *
*
```

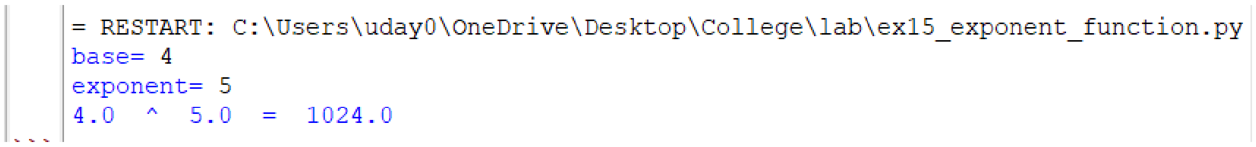
Practical 15

Write a function called exponent(base, exp) **that returns an int value of base raises to the power of exp.**

Code:

```
def exponent(base, exp):
   return base ** exp
base = float(input("base= "))
exp = float(input("exponent= "))
print(base, " ^ ", exp, " = ", exponent(base,exp))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\lab\ex15_exponent_function.py
base= 4
exponent= 5
4.0  ^  5.0  =  1024.0
```

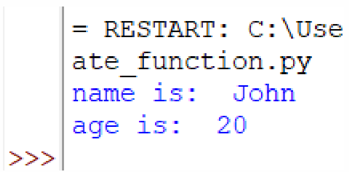
Practical – 16

Write a program to create a function that takes two arguments, name and age, and print their value.

Code:

```
def function(name, age):
    print("name is: ", name)
    print("age is: ", age)


function("John","20")
```

Output:



```
= RESTART: C:\Use
ate_function.py
name is:  John
age is:  20
>>>
```

Practical 17

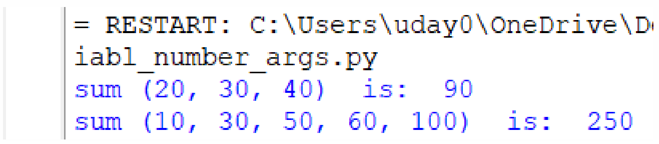
Write a program to create function func1() to accept a variable length of arguments and print their value.

Note: Create a function in such a way that we can pass any number of arguments to this function, and the function should process them and display each argument's value.

Code:

```
def func1(*args):

   print("sum",args, " is: ", sum(args))



func1(20,30,40)

func1(10,30,50,60,100)
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\D
iabl_number_args.py
sum (20, 30, 40)  is:  90
sum (10, 30, 50, 60, 100)  is:  250
```

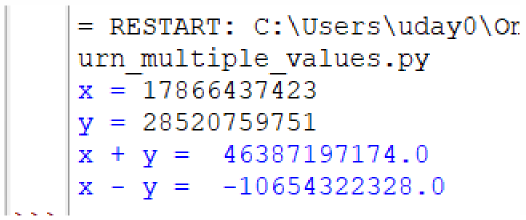
Practical-18

Write a program to create function calculation() such that it can accept two variables and calculate addition and subtraction. Also, it must return both addition and subtraction in a single return call

Code:

```
def calculation(num1,num2):
    return num1+num2, num1-num2


x = float(input("x = "))
y = float(input("y = "))
result = calculation(x,y)
print("x + y = ", result[0])
print("x - y = ", result[1])
```

Output:



```
= RESTART: C:\Users\uday0\Or
urn_multiple_values.py
x = 17866437423
y = 28520759751
x + y =  46387197174.0
x - y =  -10654322328.0
```

Practical 19

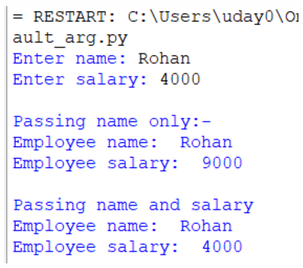
Write a program to create a function show_employee() using the following conditions.

- It should accept the employee's name and salary and display both.
- If the salary is missing in the function call then assign default value 9000 to salary

Code:

```
def show_employee(name, salary=9000):
    print("Employee name: ", name)
    print("Employee salary: ", salary)
name = input("Enter name: ")
salary = int(input("Enter salary: "))
print("\nPassing name only:-")
show_employee(name)
print("\nPassing name and salary")
show_employee(name, salary)
```

Output:



```
= RESTART: C:\Users\uday0\O
ault_arg.py
Enter name: Rohan
Enter salary: 4000

Passing name only:-
Employee name:  Rohan
Employee salary:  9000

Passing name and salary
Employee name:  Rohan
Employee salary:  4000
```

Practical – 20

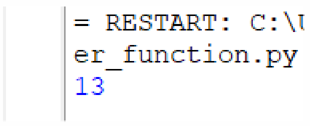
Create an inner function to calculate the addition in the following way

- Create an outer function that will accept two parameters, `a` and `b`
- Create an inner function inside an outer function that will calculate the addition of `a` and `b`
- At last, an outer function will add 5 into addition and return it

Code:

```
def calculation(a,b):
    result = 0
    def addition():
        return a+b
    result += addition()
    result += 5
    return result
print(calculation(3,5))
```

Output:



```
= RESTART: C:\
er_function.py
13
```

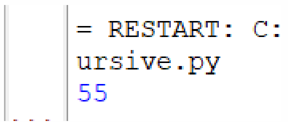
Practical – 21

Write a program to create a recursive function to calculate the sum of numbers from 0 to 10.

Code:

```
def sumNumbers(n):
    if n==0:
        return 0
    else:
        return n+sumNumbers(n-1)
print(sumNumbers(10))
```

Output:



```
= RESTART: C:
ursive.py
55
```

Practical – 22

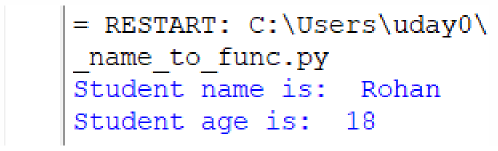
Assign a different name to function and call it through the new name

Below is the function display_student(name, age). Assign a new name show_tudent(name, age) to it and call it using the new name.

Code:

```
def display_student(name, age):

    print("Student name is: ", name)

    print("Student age is: ", age)



show_student = display_student

show_student("Rohan",18)
```

Output:



```
= RESTART: C:\Users\uday0\
_name_to_func.py
Student name is:  Rohan
Student age is:  18
```

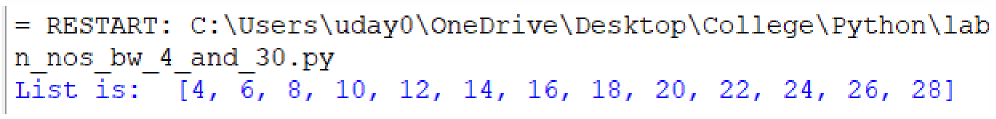
Practical – 23

Generate a Python list of all the even numbers between 4 to 30

Code:

```
def even_numbers_4_to_30():
    return list(range(4,30,2))


print("List is: ", even_numbers_4_to_30())
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Python\lab
n_nos_bw_4_and_30.py
List is:  [4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28]
```

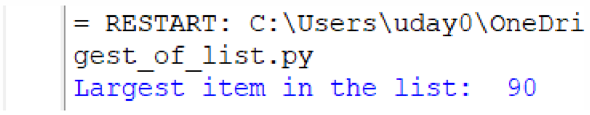
Practical – 24

Find the largest item from a given list

Code:

```
myList = [4,6,5,8,80,90,10,3,2]

print("Largest item in the list: ", max(myList))
```

Output:



```
= RESTART: C:\Users\uday0\OneDri
gest_of_list.py
Largest item in the list:  90
```

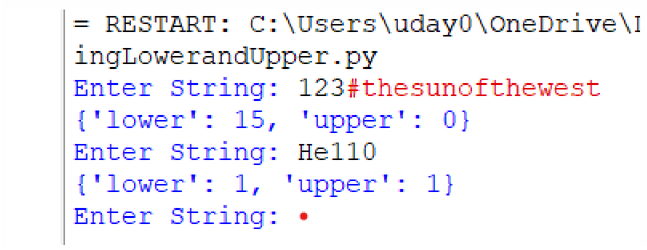
Practical-25

Write a program that defines a function count_lower_upper() that accepts a string and calculates the number of uppercase and lowercase alphabets in it. It should return these values as a dictionary. Call this function for some sample strings.

Code:

```
def count_lower_upper(userstring):
    record = {"lower":0, "upper":0}
    characterRange = list(range(65,91)) + list(range(97,123))
    for i in userstring:
        if ord(i) in characterRange:
            if i.islower():
                record["lower"]+=1
            elif i.isupper():
                record["upper"]+=1
    return record


while True:
    user = input("Enter String: ")
    print(count_lower_upper(user))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\I
ingLowerandUpper.py
Enter String: 123#thesunofthewest
{'lower': 15, 'upper': 0}
Enter String: He110
{'lower': 1, 'upper': 1}
Enter String: 
```

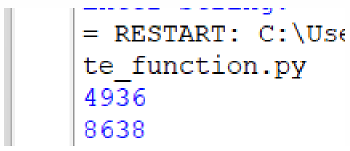
Practical – 26

Write a program that defines a function compute() that calculates the value of n + nn + nnn + nnnn, where n is digit received by the function. Test the function for digits 4 and 7.

Code:

```
def compute(n):

    return n * (1+11+111+1111)

print(compute(4))

print(compute(7))
```

Output:



```
= RESTART: C:\Use
te_function.py
4936
8638
```

Practical – 27

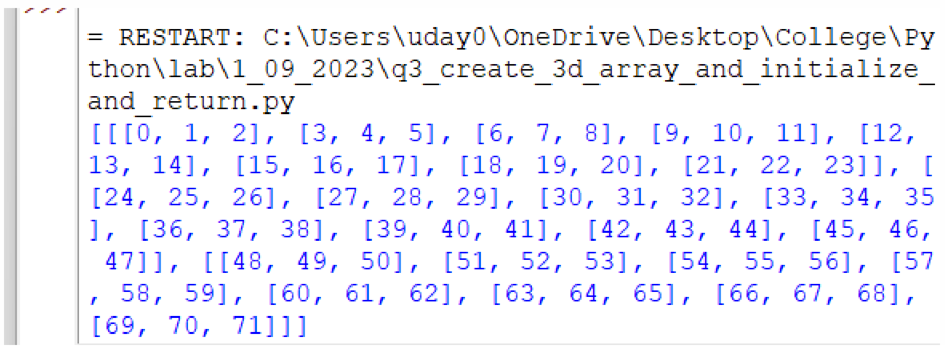
Write a program that defines a function create_array() to create and return a 3D array whose dimensions are passed to the function. Also initialize each element of this array to a value passed to the function.

Code:

```
def create_array(dims,values):
    l=0
    array = []
    if len(values)==dims[0]*dims[1]*dims[2]:
        for i in range(dims[0]):
            subarray=[]
            for j in range(dims[1]):
                subsubarray=[]
                for k in range(dims[2]):
                    subsubarray.append(values[l])
                    l+=1
                subarray.append(subsubarray)
            array.append(subarray)
    return array
```

```
aList = []
for i in range(3*8*3):
    aList.append(i)
print(create_array((3,8,3), aList))
```

<u>Output:</u>



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Py
thon\lab\1_09_2023\q3_create_3d_array_and_initialize_
and_return.py
[[[0, 1, 2], [3, 4, 5], [6, 7, 8], [9, 10, 11], [12,
13, 14], [15, 16, 17], [18, 19, 20], [21, 22, 23]], [
[24, 25, 26], [27, 28, 29], [30, 31, 32], [33, 34, 35
], [36, 37, 38], [39, 40, 41], [42, 43, 44], [45, 46,
 47]], [[48, 49, 50], [51, 52, 53], [54, 55, 56], [57
, 58, 59], [60, 61, 62], [63, 64, 65], [66, 67, 68],
[69, 70, 71]]]
```

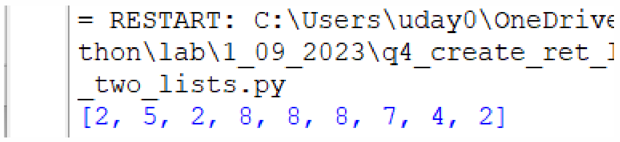
Practical – 28

Write a program that defines a function create_list() to create and return a list which is an intersection of two lists passed to it.

Code:

```
def create_array(list1,list2):
    list3=[]
    for i in list1:
        if i in list2:
            list3.append(i)
    return list3
print(create_array([3, 2, 5, 2, 8, 8, 8, 7, 4, 2],[8, 1, 7, 7, 1, 8, 5, 4, 2, 6]))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive
thon\lab\1_09_2023\q4_create_ret_l
_two_lists.py
[2, 5, 2, 8, 8, 8, 7, 4, 2]
```

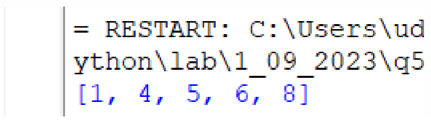
Practical – 29

Write a program that defines a function sanitize_list() to remove all duplicate entries from the list that it receives.

Code:

```
def sanitize_list(givenList):
    newList = []
    for i in set(givenList):
        newList.append(i)
    return newList
print(sanitize_list([8, 8, 6, 4, 1, 6, 1, 5, 8, 5]))
```

Output:



```
= RESTART: C:\Users\ud
ython\lab\1_09_2023\q5
[1, 4, 5, 6, 8]
```

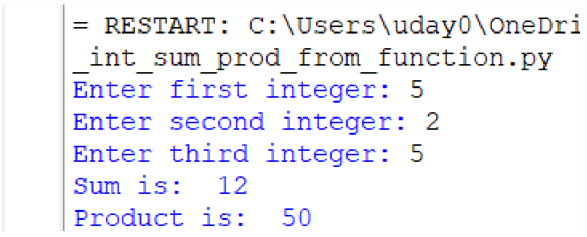
Practical – 30

Write a program to receive three integers from keyboard and get their sum and product calculated through a user-defined function cal_sum_prod().

Code:

```
def cal_sum_prod(num1,num2,num3):

    return num1+num2+num3, num1*num2*num3



n1 = int(input("Enter first integer: "))

n2 = int(input("Enter second integer: "))

n3 = int(input("Enter third integer: "))

result = cal_sum_prod(n1,n2,n3)



print("Sum is: ", result[0])

print("Product is: ", result[1])
```

Output:



```
= RESTART: C:\Users\uday0\OneDri
_int_sum_prod_from_function.py
Enter first integer: 5
Enter second integer: 2
Enter third integer: 5
Sum is:  12
Product is:  50
```

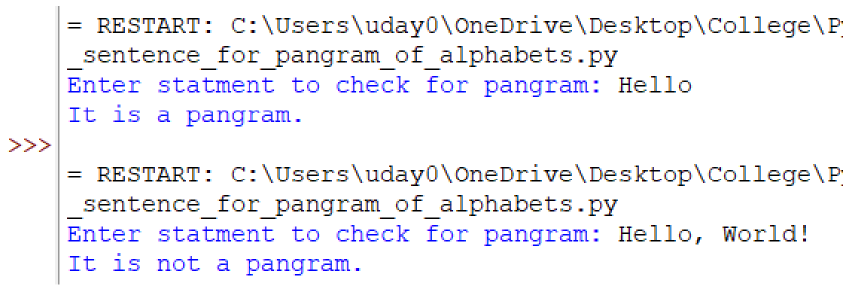
Practical – 31

Pangram is a sentence that uses every letter of the alphabet. Write a program that checks whether a given string is pangram or not, through a user-defined function ispangram().

Code:

```
def ispangram(sentence):

    alphabets = [chr(i) for i in range(65,91)] + [chr(i) for i in range(97,123)] #list of
all alphabets lowercase and uppercase

    for i in set(sentence):

        if i not in alphabets:

            return False

    return True

statement = input("Enter statment to check for pangram: ")

if ispangram(statement):

    print("It is a pangram.")

else:

    print("It is not a pangram.")
```

Output:



```
 = RESTART: C:\Users\uday0\OneDrive\Desktop\College\P
 _sentence_for_pangram_of_alphabets.py
 Enter statment to check for pangram: Hello
 It is a pangram.
>>>
 = RESTART: C:\Users\uday0\OneDrive\Desktop\College\P
 _sentence_for_pangram_of_alphabets.py
 Enter statment to check for pangram: Hello, World!
 It is not a pangram.
```

Practical – 32

Write a Python program that accepts a hyphen-separated sequence of words as input and calls a function convert() which converts it into a hyphen-separated sequence after sorting them alphabetically. For example, if the input string is

here-come-the-dots-followed-by-dashes

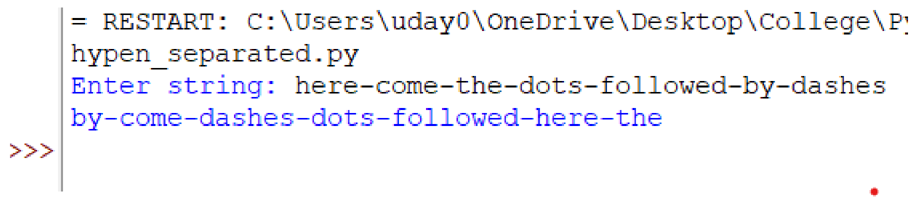
Then, the output must be:

by-come-dashes-dots-followed-here-the

Code:

```
def convert(sentence):
    return "-".join(sorted(sentence.split("-")))
user_input = input("Enter string: ")
print(convert(user_input))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\P
hypen_separated.py
Enter string: here-come-the-dots-followed-by-dashes
by-come-dashes-dots-followed-here-the
>>>
```

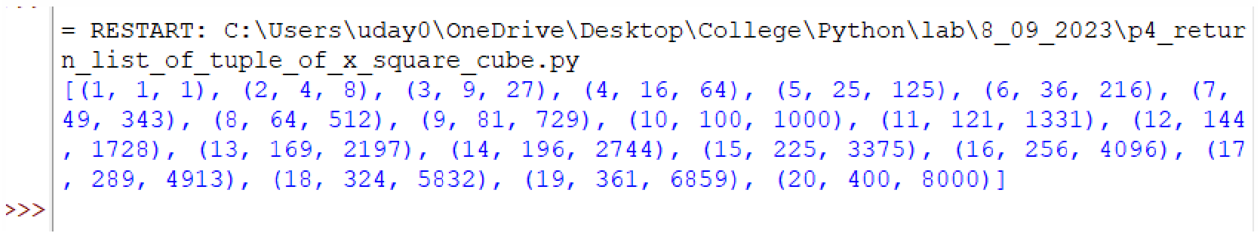
Practical – 33

Write a python function to create and return a list containing tuples of the form (x, x2, x3) for all x between 1 and 20 (both included).

Code:

```
def x_square_cube():
    returnList = []
    for x in range(1,21):
        returnList.append((x,x**2,x**3))


    return returnList
print(x_square_cube())
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Python\lab\8_09_2023\p4_retur
n_list_of_tuple_of_x_square_cube.py
[(1, 1, 1), (2, 4, 8), (3, 9, 27), (4, 16, 64), (5, 25, 125), (6, 36, 216), (7,
49, 343), (8, 64, 512), (9, 81, 729), (10, 100, 1000), (11, 121, 1331), (12, 144
, 1728), (13, 169, 2197), (14, 196, 2744), (15, 225, 3375), (16, 256, 4096), (17
, 289, 4913), (18, 324, 5832), (19, 361, 6859), (20, 400, 8000)]
>>>
```

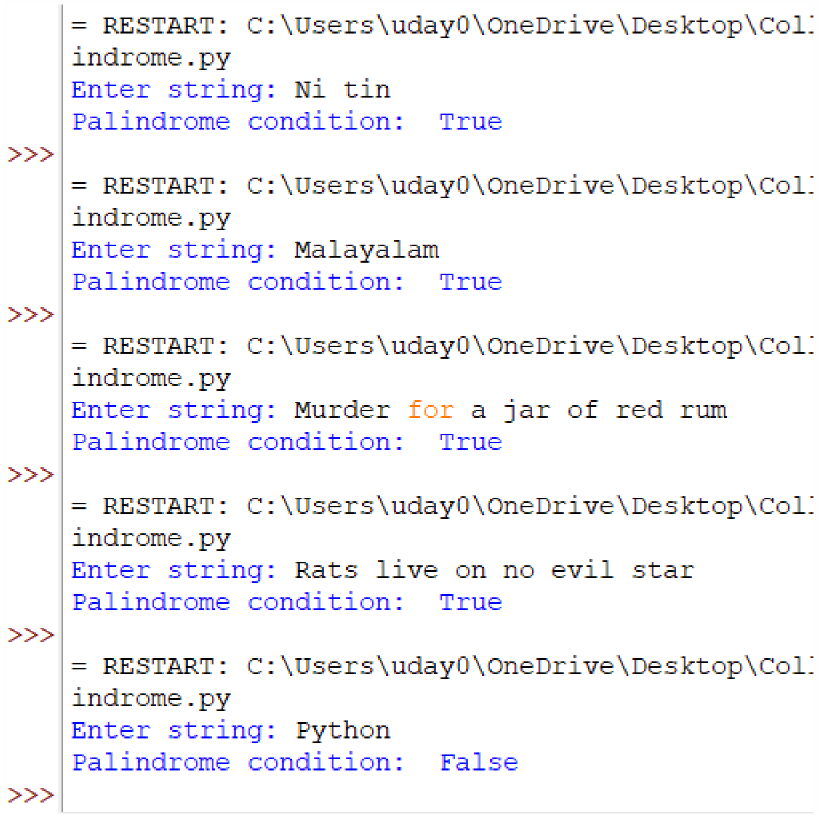
Practical – 34

Write a program that defines a function ispalindrome() which checks whether a given string is a palindrome or not. Ignore spaces and case mismatch while checking for palindrome.

Code:

```
def ispalindrome(arg_string):
    arg_string = arg_string.replace(" ","").lower()
    if arg_string == arg_string[::-1]:
        return True
    return False
user_string = input("Enter string: ")
print("Palindrome condition: ",ispalindrome(user_string))
```

Output:



```
    = RESTART: C:\Users\uday0\OneDrive\Desktop\Col
    indrome.py
    Enter string: Ni tin
    Palindrome condition:  True
>>>
    = RESTART: C:\Users\uday0\OneDrive\Desktop\Col
    indrome.py
    Enter string: Malayalam
    Palindrome condition:  True
>>>
    = RESTART: C:\Users\uday0\OneDrive\Desktop\Col
    indrome.py
    Enter string: Murder for a jar of red rum
    Palindrome condition:  True
>>>
    = RESTART: C:\Users\uday0\OneDrive\Desktop\Col
    indrome.py
    Enter string: Rats live on no evil star
    Palindrome condition:  True
>>>
    = RESTART: C:\Users\uday0\OneDrive\Desktop\Col
    indrome.py
    Enter string: Python
    Palindrome condition:  False
>>>
```

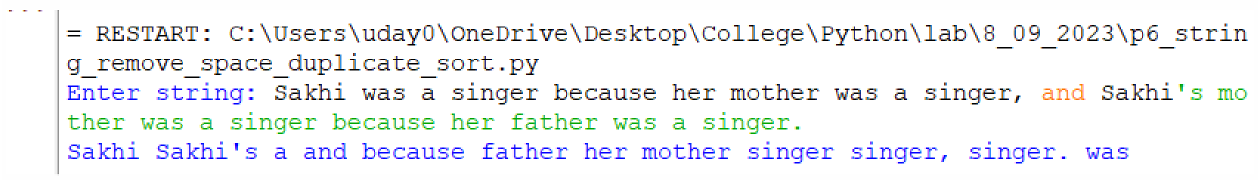
Practical – 35

Write a program that defines a function convert() that receives a string containing a sequence of whitespace separated words and returns a string after removing all duplicate words and sorting them alphanumerically.

Code:

```
def convert(arg_string):
    x = sorted(set(arg_string.split(" ")))
    return " ".join(x)


sentence = input("Enter string: ")
print(convert(sentence))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Python\lab\8_09_2023\p6_strin
g_remove_space_duplicate_sort.py
Enter string: Sakhi was a singer because her mother was a singer, and Sakhi's mo
ther was a singer because her father was a singer.
Sakhi Sakhi's a and because father her mother singer singer, singer. was
```

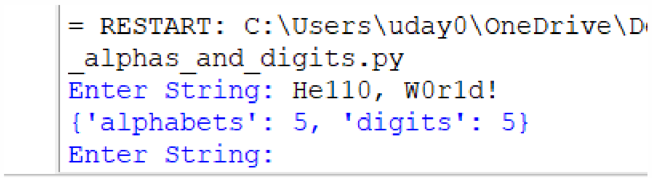
Practical – 36

Write a program that defines a function count_alphabets_digits() that accepts a string and calculates the number of alphabets and digits in it. It should return these values as a dictionary. Call this function for some sample strings.

Code:

```
def count_alphabets_digits(userstring):
    record = {"alphabets":0, "digits":0}
    characterRange = list(range(65,91)) + list(range(97,123)) + list(range(48,58))
    for i in userstring:
        if ord(i) in characterRange:
            if i.isalpha():
                record["alphabets"]+=1
            elif i.isdigit():
                record["digits"]+=1
    return record


while True:
    user = input("Enter String: ")
    print(count_alphabets_digits(user))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\D
_alphas_and_digits.py
Enter String: Hel10, W0r1d!
{'alphabets': 5, 'digits': 5}
Enter String:
```

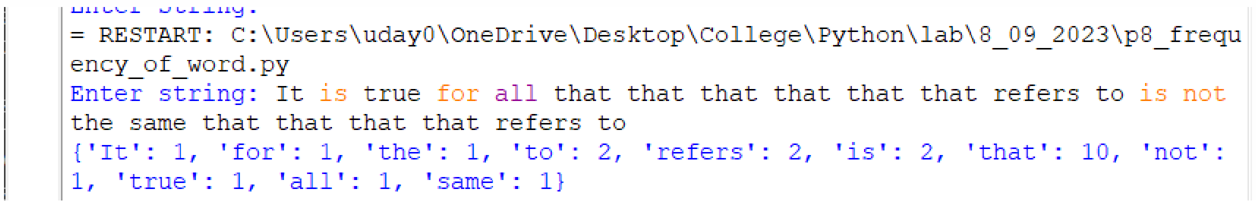
Practical – 37

Write a program that defines a function called frequency() which computes the frequency of words present in a string passed on it. The frequencies should be returned in sorted order by words in the string.

Code:

```
def frequency(arg_string):
    returnDict = {}
    arg_string = arg_string.split(" ")
    for i in set(arg_string):
        returnDict[i] = 0
    for i in arg_string:
        returnDict[i] +=1
    return returnDict
user_string = input("Enter string: ")
print(frequency(user_string))
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Python\lab\8_09_2023\p8_frequ
ency_of_word.py
Enter string: It is true for all that that that that that that refers to is not
the same that that that that refers to
{'It': 1, 'for': 1, 'the': 1, 'to': 2, 'refers': 2, 'is': 2, 'that': 10, 'not':
1, 'true': 1, 'all': 1, 'same': 1}
```

Practical – 38

Write a program that defines two functions called create_sent1() and create_sent2().

Both receive following 3 lists:

subjects = [‘He’,’She’]

verb = [‘loves’,’hates’]

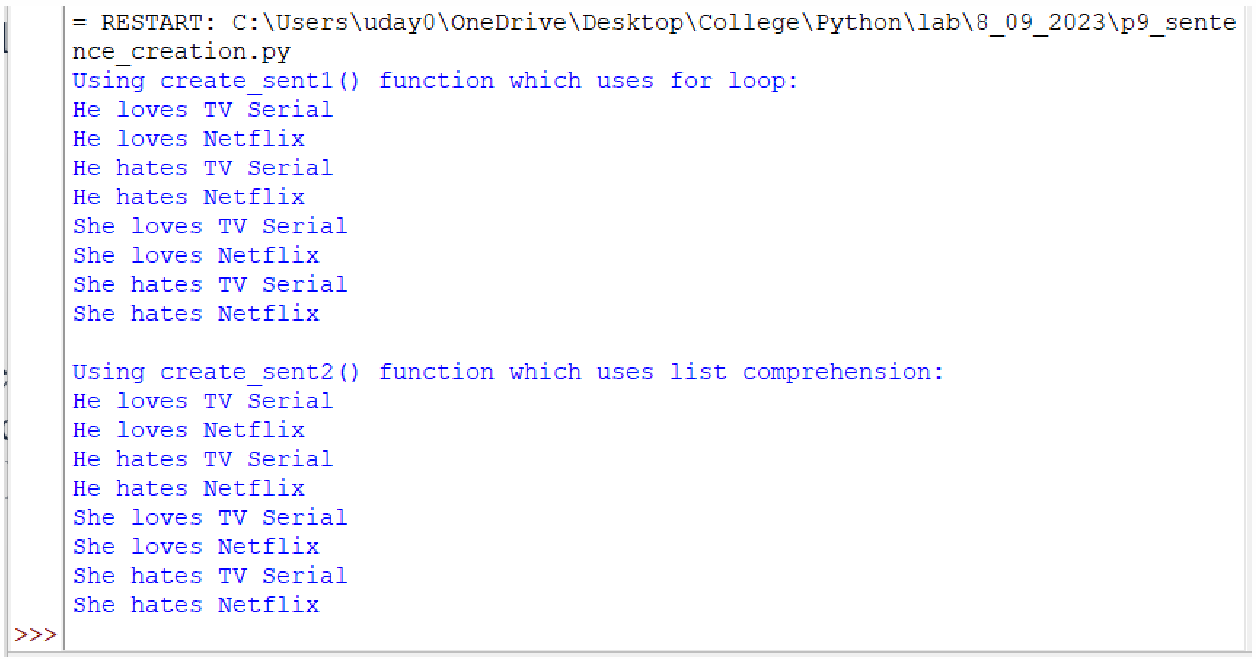
objects = [‘TV Serials’,’Netflix’]

Both functions should form sentences by picking elements from the lists and returns them. Use for loops in create_sent1() and list comprehension in create_sent2().

Code:

```
def create_sent1(sub,verb,obj):

    # using for loop

    sentences = []

    for i in sub:

        for j in verb:

            for k in obj:

                sentences.append(" ".join((i,j,k)))

    return sentences


def create_sent2(sub,verb,obj):
    # Using list comprehension
    return [" ".join((i,j,k)) for i in sub for j in verb for k in obj]

subject = ['He','She']
verb = ['loves', 'hates']
obj = ['TV Serial','Netflix']

print("Using create_sent1() function which uses for loop:")
for i in create_sent1(subject,verb,obj):
    print(i)

print("\nUsing create_sent2() function which uses list comprehension:")
for i in create_sent2(subject,verb,obj):
    print(i)
```

Output:



```
= RESTART: C:\Users\uday0\OneDrive\Desktop\College\Python\lab\8_09_2023\p9_sente
nce_creation.py
Using create_sent1() function which uses for loop:
He loves TV Serial
He loves Netflix
He hates TV Serial
He hates Netflix
She loves TV Serial
She loves Netflix
She hates TV Serial
She hates Netflix

Using create_sent2() function which uses list comprehension:
He loves TV Serial
He loves Netflix
He hates TV Serial
He hates Netflix
She loves TV Serial
She loves Netflix
She hates TV Serial
She hates Netflix
>>>
```