9/9/2023

Abhishek Roka

10621019

Python Practical File

Full Time Diploma in Computer Engineering 5th Semester

INDEX

[Calculate the multiplication and sum of two numbers 4](#_Toc145198781)

[Print the sum of the current number and the previous number 5](#_Toc145198782)

[Print characters from a string that are present at an even index number 6](#_Toc145198783)

[Remove first n characters from a string 7](#_Toc145198784)

[Check if the first and last number of a list is the same 8](#_Toc145198785)

[Display numbers divisible by 5 from a list 9](#_Toc145198786)

[Return the count of a given substring from a string 10](#_Toc145198787)

[Print the following pattern 11](#_Toc145198788)

[Check Palindrome Number 12](#_Toc145198789)

[Create a new list from a two list using the following condition 13](#_Toc145198790)

[Write a Program to extract each digit from an integer in the reverse order. 14](#_Toc145198791)

[Calculate income tax for the given income by adhering to the below rules 15](#_Toc145198792)

[Print multiplication table form 1 to 10 17](#_Toc145198793)

[Print downward Half-Pyramid Pattern with Star (asterisk) 19](#_Toc145198794)

[Write a function called exponent(base, exp) that returns an int value of base raises to the power of exp. 20](#_Toc145198795)

[Write a program to create a function that takes two arguments, name and age, and print their value. 21](#_Toc145198796)

[Write a program to create function func1() to accept a variable length of arguments and print their value. 22](#_Toc145198797)

[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 23](#_Toc145198798)

[Write a program to create a function show\_employee() using the following conditions. 24](#_Toc145198799)

[Create an inner function to calculate the addition in the following way 25](#_Toc145198800)

[Write a program to create a recursive function to calculate the sum of numbers from 0 to 10. 26](#_Toc145198801)

[Assign a different name to function and call it through the new name 27](#_Toc145198802)

[Generate a Python list of all the even numbers between 4 to 30 28](#_Toc145198803)

[Find the largest item from a given list 29](#_Toc145198804)

[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. 30](#_Toc145198805)

[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. 32](#_Toc145198806)

[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. 33](#_Toc145198807)

[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. 35](#_Toc145198808)

[Write a program that defines a function sanitize\_list() to remove all duplicate entries from the list that it receives. 36](#_Toc145198809)

[Write a program to receive three integers from keyboard and get their sum and product calculated through a user-defined function cal\_sum\_prod(). 37](#_Toc145198810)

[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(). 38](#_Toc145198811)

[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. 40](#_Toc145198812)

[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). 41](#_Toc145198813)

[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. 42](#_Toc145198814)

[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. 44](#_Toc145198815)

[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. 45](#_Toc145198816)

[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. 47](#_Toc145198817)

[Write a program that defines two functions called create\_sent1() and create\_sent2(). 48](#_Toc145198818)

**Practical 1**

Calculate the multiplication and sum of two numbers

**Code**

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

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

mul = num1\*num2

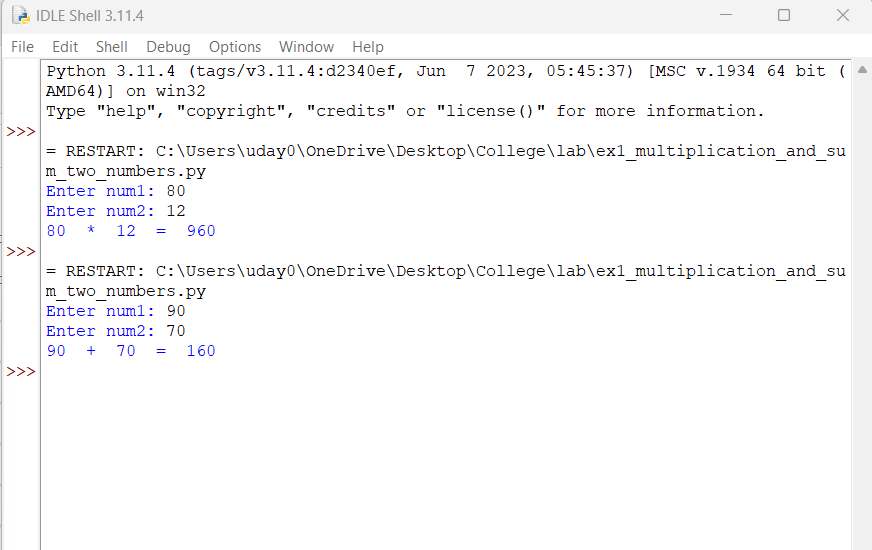
if mul <= 1000:

print(num1, " \* ", num2, " = ", mul)

else:

print(num1, " + ", num2, " = ", num1+num2)

**Output:**



**Practical 2**

Print the sum of the current number and the previous number

**Code:**

prev\_num = 0

rangelen = range(10)

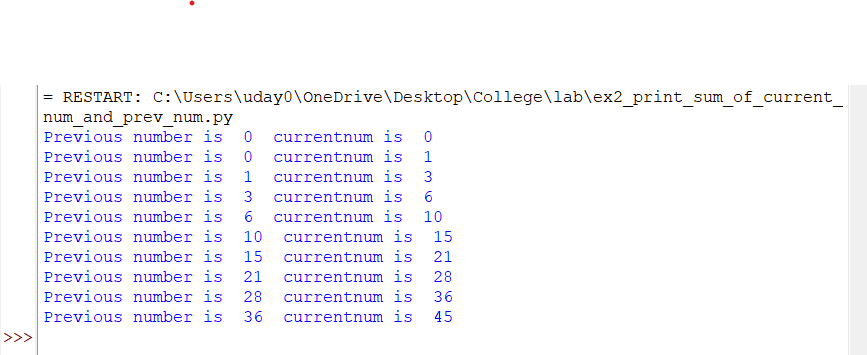
currentnum=0

for i in rangelen:

currentnum = prev\_num + i

print("Previous number is ", prev\_num, " currentnum is ", currentnum)

prev\_num = currentnum

**Output:**

**Practical 3**

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

**Code:**

user\_string = input("Enter string: ")

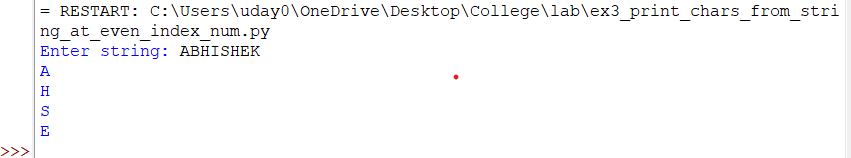
length = len(user\_string)

range\_len = range(0,length,2)

for i in range\_len:

print(user\_string[i])

**Output:**



**Practical 4**

Remove first n characters from a string

**Code:**

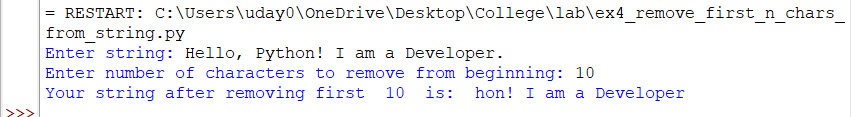
user\_string = input("Enter string: ")

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:**



**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

**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:**

### 

### 

**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:**

### 

**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:**

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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))

### Output:

### 

**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:

### 

**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:

### 

**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:

### 

**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: ")

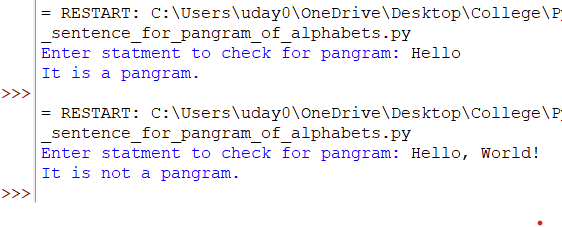
### if ispangram(statement):

### print("It is a pangram.")

### else:

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

### Output:



**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:

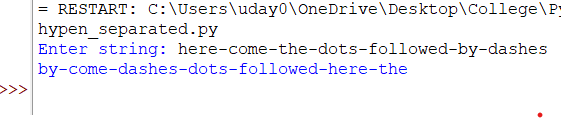
### def convert(sentence):

### return "-".join(sorted(sentence.split("-")))

### user\_input = input("Enter string: ")

### print(convert(user\_input))

### Output:



**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 

**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:

### 