ALIGARH MUSLIM UNIVERSITY



DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF SCIENCE

**Computer Lab work**

**CAMS-3P01: Laboratory Course III**

**M.C.A 3Rd SEMESTER**

### SUBMITTED TO

### MR. IMSHAD AHMAD KHAN

### MR. MOHD FARAZ

### MR. MD NAVED IQBAL

### MR. TIPU SULTAN

### DR. PARVEJ AALAM

### MR. ZUBAIR AHMAD HASHMI

**SUBMITTED BY**

**MOHD FARAZ ALAM**

**23CAMSA129**

**GJ9695**

**WEEK 1**

1. Write a program to find the product of two user-supplied integers and if the product is equal to or lower than 5000, then return the sum of the two numbers.

**Program:**

def product\_and\_sum(a, b):

    product = a \* b

    if product <= 5000:

        return a + b

    return product

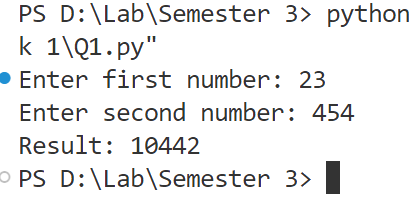
num1 = int(input("Enter first number: "))

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

result = product\_and\_sum(num1, num2)

print("Result:", result)

**Output:**

****

1. Write a program to print the sum of the first 10 numbers.

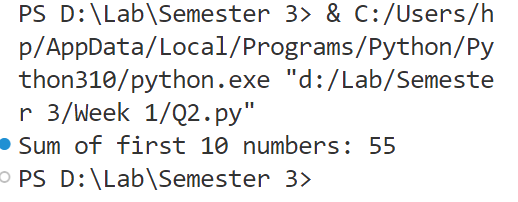
**Program:**

def sum\_of\_first\_10\_numbers():

    return sum(range(1, 11))

print("Sum of first 10 numbers:", sum\_of\_first\_10\_numbers())

**Output:**



1. Write a program to iterate through a supplied list of 20 numbers and print only those numbers which are divisible by 5.

**Program:**

def print\_divisible\_by\_5(numbers):

    for num in numbers:

        if num % 5 == 0:

            print(num)

numbers = []

print("Please enter 20 numbers:")

for \_ in range(20):

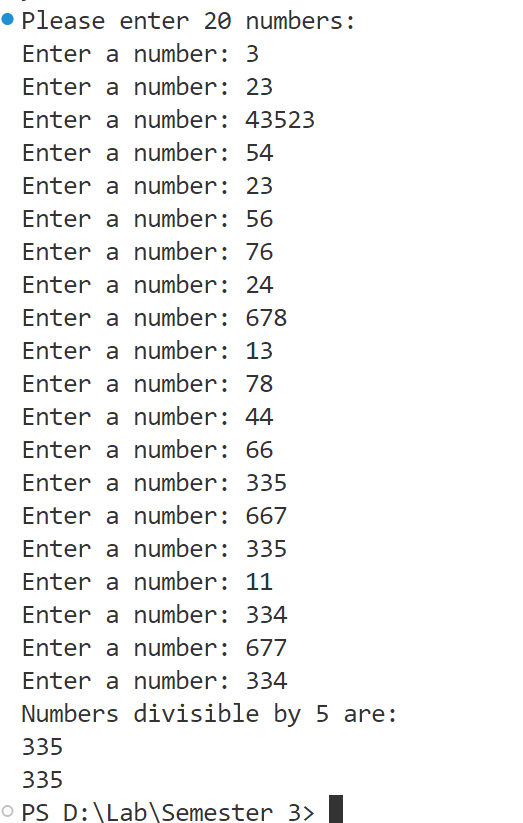
    number = int(input("Enter a number: "))

    numbers.append(number)

print("Numbers divisible by 5 are:")

print\_divisible\_by\_5(numbers)

**Output:**



1. Write a program to check if the given number is a palindrome.

**Program:**

def is\_palindrome(number):

    return str(number) == str(number)[::-1]

num = int(input("Enter a number: "))

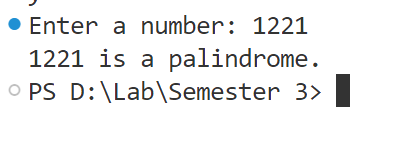
if is\_palindrome(num):

    print(f"{num} is a palindrome.")

else:

    print(f"{num} is not a palindrome.")

**Output:**



1. Write a program to calculate the cube of all numbers from 1 to a given number.

**Program:**

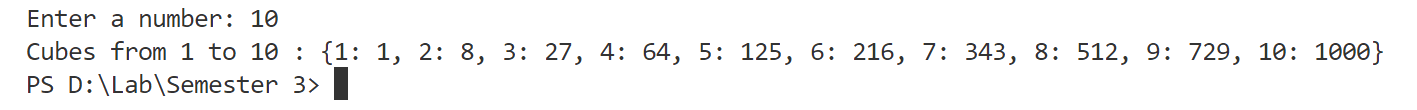
def cubes\_up\_to\_n(n):

    return {i: i\*\*3 for i in range(1, n+1)}

n = int(input("Enter a number: "))

print("Cubes from 1 to", n, ":", cubes\_up\_to\_n(n))

**Output:**



**WEEK 2**

1. Write a program to extract each digit from an integer in reverse order.

**Program:**

def extract\_digits\_reverse(num):

    while num > 0:

        digit = num % 10

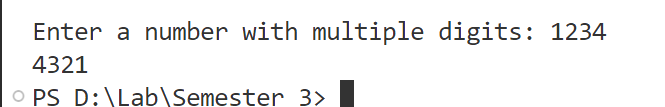
        print(digit, end="")

        num //= 10

num = int(input("Enter a number with multiple digits: "))

extract\_digits\_reverse(num)

**Output:**



1. Write a program to count the total number of digits in a number using a while loop.

**Program:**

def count\_digits(num):

    count = 0

    while num != 0:

        num //= 10

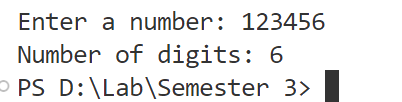
        count += 1

    return count

num = int(input("Enter a number: "))

print("Number of digits:", count\_digits(num))

**Output:**



1. Write a program to display all prime numbers within a range.

**Program:**

def is\_prime(n):

    if n <= 1:

        return False

    for i in range(2, n):

        if n % i == 0:

            return False

    return True

lower = int(input("Enter the lower limit: "))

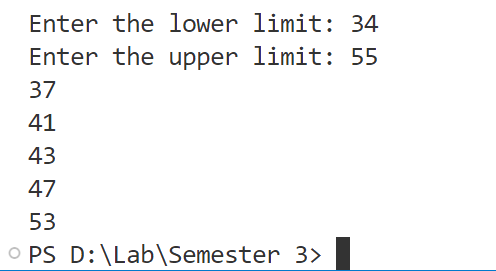
upper = int(input("Enter the upper limit: "))

for num in range(lower, upper + 1):

    if is\_prime(num):

        print(num)

**Output:**



1. Write a program to use a loop to find the factorial of a given number.

**Program:**

def factorial(n):

    if n == 0:

        return 1

    fact = 1

    for i in range(1, n + 1):

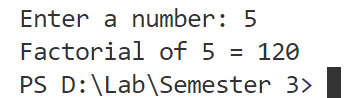
        fact \*= i

    return fact

num = int(input("Enter a number: "))

print(f"Factorial of {num} = {factorial(num)}")

**Output:**



1. Write a program to find the sum of the digits of a supplied integer.

**Program:**

def sum\_of\_digits(num):

    sum =0

    while num > 0:

        digit = num % 10

        sum = sum +digit

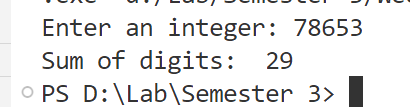
        num  = num// 10

    return sum

num = int(input("Enter an integer: "))

print("Sum of digits: ", sum\_of\_digits(num))

**Output:**



**WEEK 3**

1. Write a program to print the following pattern using the for loop:

5 4 3 2 1

4 3 2 1

3 2 1

2 1

1

**Program:**

rows = int(input("Enter Row: "))

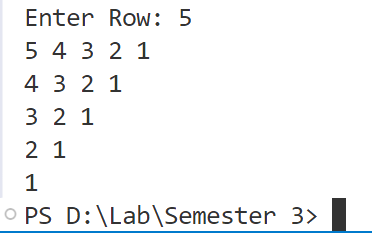
for i in range(rows, 0,-1):

    for j in range(i, 0,-1):

        print(j, end=" ")

    print()

**Output:**



2. Write a program to print the following star pattern using the for loop:

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

**Program:**

rows = int(input('Enter Row: '))

for i in range(1, rows + 1):

    for j in range(i):

        print("\*", end=" ")

    print()

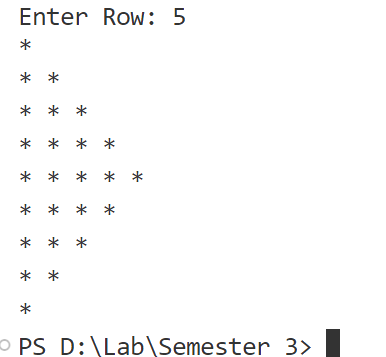
for i in range(rows - 1, 0, -1):

    for j in range(i):

        print("\*", end=" ")

    print()

**Output:**



3. Write a program to print characters from a string which are present at even index numbers.

**Program:**

def print\_even\_index\_characters(string):

    result = ""

    for i in range(len(string)):

        if i % 2 == 0:

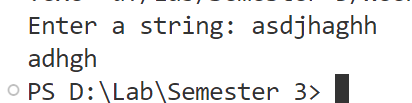
            result += string[i]

    print(result)

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

print\_even\_index\_characters(input\_string)

**Output:**



4. Write a program to accept a string from the user and display characters that are present at even index numbers.

**Program:**

def print\_even\_index\_characters\_from\_user\_input():

    string = input("Enter a string: ")

    result = ""

    for i in range(len(string)):

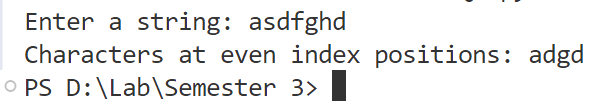
        if i % 2 == 0:

            result += string[i]

    print("Characters at even index positions:", result)

print\_even\_index\_characters\_from\_user\_input()

**Output:**



1. Write a program to remove characters from a string starting from the nth position to the last and return a new string. Example: remove\_chars("aligarh", 3) should output ali.

**Program:**

def remove\_chars(string, n):

    if n > len(string):

        return ""

    return string[:n]

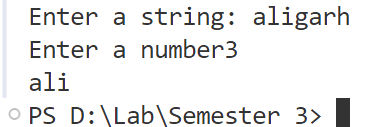
str =  input("Enter a string: ")

num =  int(input("Enter a number"))

new\_string = remove\_chars(str, num)

print(new\_string)

**Output:**



**Week 4**

1. Write a program to create a function cal\_sum\_sub() that accepts two variables and calculates addition and subtraction. Also, it must return both addition and subtraction in a single return call.

**Program**:

def cal\_sum\_sub(a, b):

    addition = a + b

    subtraction = a - b

    return addition, subtraction

a =  int(input("Enter 1st number: "))

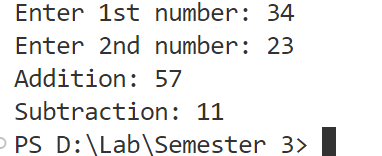
b =  int(input("Enter 2nd number: "))

result = cal\_sum\_sub(a, b)

print("Addition:", result[0])

print("Subtraction:", result[1])

**Output:**



1. Write a function to return True if the first and last number of a given list are the same. If the numbers are different, return False.

**Program:**

def is\_first\_last\_same(numbers):

    return numbers[0] == numbers[-1]

numbers = []

a =  input("Enter number (for quit enter 'q'): ")

while a != 'q':

    try:

        numbers.append(int(a))

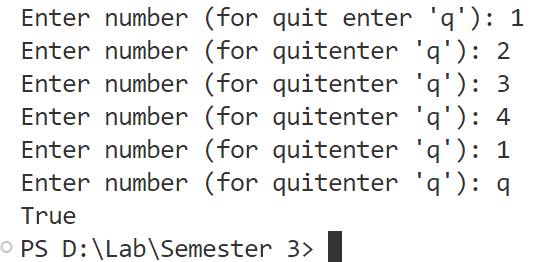
    except:

        print("enter valid input")

    a =  input("Enter number (for quitenter 'q'): ")

print(is\_first\_last\_same(numbers))

**Output:**



1. Given a list of numbers, write a program to turn every item of the list into its square.

Program:

def square\_list(lst):

    return [x\*\*2 for x in lst]

numbers = []

a =  input("Enter number (for quit enter 'q'): ")

while a != 'q':

    try:

        numbers.append(int(a))

    except:

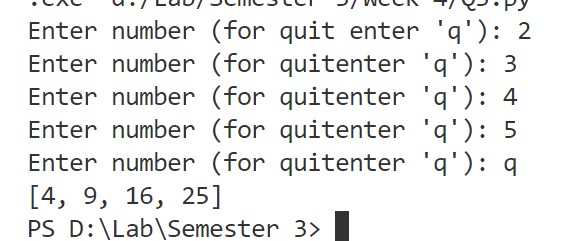
        print("enter valid input")

    a =  input("Enter number (for quitenter 'q'): ")

squared\_numbers = square\_list(numbers)

print(squared\_numbers)

Output:



1. Given two Python lists, write a program to iterate both lists simultaneously and display items from list 1 in original order and items from list 2 in reverse order.

Program:

def iterate\_lists(lst1, lst2):

    for x, y in zip(lst1, lst2[::-1]):

        print(f"List 1 item: {x}, List 2 item: {y}")

def input\_fun(a):

    l =[]

    a =  input(f"Enter value {a} for list (for quit enter 'q'): ")

    while a != 'q':

        try:

            l.append(int(a))

        except:

            pass

        a =  input("Enter number (for quitenter 'q'): ")

    return l;

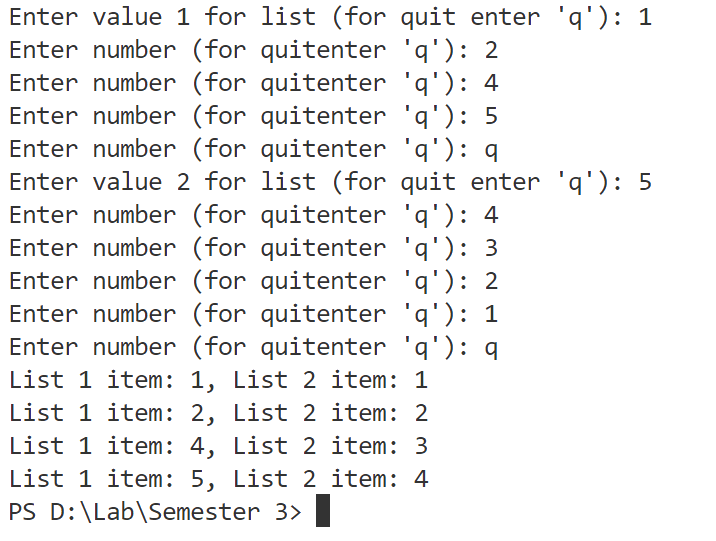
list1 = input\_fun(1)

print("Enter value for 2nd list")

list2  =input\_fun(2)

iterate\_lists(list1, list2)

**Output:**



1. Write a program to count the number of occurrences of item 50 in the tuple tp1 = (50, 10, 60, 70, 50).

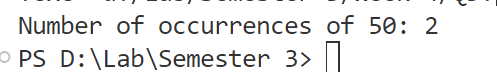
Program:

tp1 = (50, 10, 60, 70, 50)

count\_50 = tp1.count(50)

print("Number of occurrences of 50:", count\_50)

Output:



**Week 5**

1. Write a program to generate a 6-digit random secure OTP.

**Program:**

import random

import math

def generate\_otp():

    digits = "0123456789"

    OTP = ""

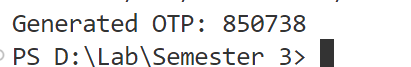
    for i in range(6) :

        OTP += digits[math.floor(random.random() \* 10)]

    return OTP

print("Generated OTP:", generate\_otp())

**Output:**



1. Write a program to pick a random character from a user-supplied string.

**Program:**

import random

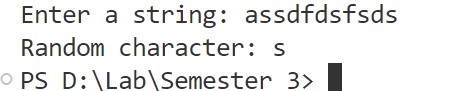
def pick\_random\_char(user\_string):

    return random.choice(user\_string)

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

print("Random character:", pick\_random\_char(user\_string))

**Output:**



1. Write a program to generate a random password that meets the following conditions:

a. Password length must be 10 characters long.

b. It must contain at least 2 uppercase letters, 1 digit, and 1 special symbol.

**Program:**

import random

import string

def generate\_password():

    upper\_case = random.choices(string.ascii\_uppercase, k=2)

    digit = random.choice(string.digits)

    remaining\_chars = random.choices(string.ascii\_letters + string.digits + string.punctuation, k=6)

    password\_list = upper\_case + [digit] + remaining\_chars

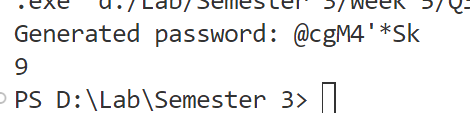
    random.shuffle(password\_list)

    return ''.join(password\_list)

print("Generated password:", generate\_password())

print(len(generate\_password()))

Output:



1. Given two lists of numbers, write a program to create a new list containing odd numbers from the first list and even numbers from the second list.

Program:

def merge\_odd\_even(lst1, lst2):

    odd\_numbers = [x for x in lst1 if x % 2 != 0]

    even\_numbers = [x for x in lst2 if x % 2 == 0]

    return odd\_numbers + even\_numbers

# Example usage

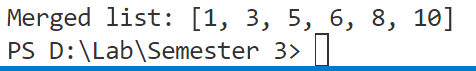
list1 = [1, 2, 3, 4, 5]

list2 = [6, 7, 8, 9, 10]

result = merge\_odd\_even(list1, list2)

print("Merged list:", result)

Output:



1. Write a program to create a numpy array and return an array of odd rows and even columns from the numpy array.

Program:

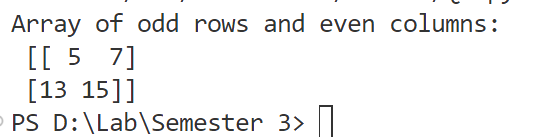
import numpy as np

arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12], [13, 14, 15, 16]])

result = arr[1::2, ::2]

print("Array of odd rows and even columns:\n", result)

**Output:**



6. Write a program to create a numpy array and sort it as per the following cases:

a. Case 1: Sort the array by the second row.

b. Case 2: Sort the array by the second column.

**Program:**

import numpy as np

# a) Sort array by second row

def sort\_by\_second\_row(arr):

    return arr[:, arr[1, :].argsort()]

# b) Sort array by second column

def sort\_by\_second\_column(arr):

    return arr[arr[:, 1].argsort()]

arr = np.array([[4, 2, 3], [9, 1, 7], [8, 5, 6]])

# Case 1

sorted\_by\_row = sort\_by\_second\_row(arr)

print("Array sorted by second row:\n", sorted\_by\_row)

# Case 2

sorted\_by\_column = sort\_by\_second\_column(arr)

print("Array sorted by second column:\n", sorted\_by\_column)

**Output:**

