**PYTHON PRACTICAL (DAY-1)**

**1]LSD & MSD:**

number = input("Enter a number: ")

lsd = number[-1]

msd = number[0]

lsd = int(lsd)

msd = int(msd)

print(f"Least Significant Digit (LSD): {lsd}")

print(f"Most Significant Digit (MSD): {msd}")

**2]SQUARE & CUBE:**

n=int(input('enter the number:'))

x=n\*\*2

y=n\*\*3

print(int(x))

print(int(y))

**3]UPPER STRING & SPACE COUNT B/W STRING:**

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

uppercase\_string = user\_string.upper()

space\_count = user\_string.count(' ')

print(f"Uppercase String: {uppercase\_string}")

print(f"Number of Spaces: {space\_count}")

**4]LEAP YEAR PRINT NEXT YEAR WITH DAY MONTH YEAR:**

# Step 1: Input the year from the user

year = int(input("Enter the anniversary year: "))

# Step 2: Determine if the year is a leap year

def is\_leap\_year(year):

if year % 4 == 0:

if year % 100 == 0:

if year % 400 == 0:

return True

else:

return False

else:

return True

else:

return False

# Step 3: Check if the year is a leap year

if is\_leap\_year(year):

next\_anniversary = year + 1

print(f"The year {year} is a leap year.")

print(f"The next anniversary year is {next\_anniversary}.")

else:

previous\_anniversary = year - 1

print(f"The year {year} is not a leap year.")

print(f"The previous anniversary year is {previous\_anniversary}.")

**or**

def is\_leap\_year(year):

"""Check if a year is a leap year."""

if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):

return True

else:

return False

year = int(input("Enter the year of the anniversary: "))

if is\_leap\_year(year):

print(f"{year} is a leap year.")

next\_anniversary = year + 1

print(f"The next anniversary year is {next\_anniversary}.")

else:

print(f"{year} is not a leap year.")

previous\_anniversary = year - 1

print(f"The previous anniversary year is {previous\_anniversary}.")

**5]Perform operation on the tuple:**

ple1 = tuple(input("Enter elements of the first tuple separated by space: ").split())

tuple2 = tuple(input("Enter elements of the second tuple separated by space: ").split())

concatenated\_tuple = tuple1 + tuple2

element = input("Enter the element to find the index and count: ")

try:

index\_of\_element = concatenated\_tuple.index(element)

except ValueError:

index\_of\_element = None

count\_of\_element = concatenated\_tuple.count(element)

print(f"Concatenated Tuple: {concatenated\_tuple}")

if index\_of\_element is not None:

print(f"The index of the element '{element}' is: {index\_of\_element}")

else:

print(f"The element '{element}' is not found in the tuple.")

print(f"The count of the element '{element}' is: {count\_of\_element}")

**6]Printing all Non-Prime Numbers b/w A&B:**

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

b = int(input("Enter the ending number (b): "))

def is\_prime(n):

if n <= 1:

return False

for i in range(2, int(n\*\*0.5) + 1):

if n % i == 0:

return False

return True

non\_prime\_numbers = []

for num in range(a, b + 1):

if not is\_prime(num):

non\_prime\_numbers.append(num)

print("Non-prime numbers between", a, "and", b, "are:", non\_prime\_numbers)

**7]To Sort a list according to the length of elements:**

# Function to sort a list by the length of its elements

def sort\_by\_length(words):

return sorted(words, key=len)

# Taking input from the user

input\_list = input("Enter a list of words separated by spaces: ").split()

# Sorting the list

sorted\_list = sort\_by\_length(input\_list)

# Printing the sorted list

print("Sorted list by length of elements:", sorted\_list)

**or**

input\_list = input("Enter a list of words separated by spaces: ").split()

sorted\_list = sorted(input\_list, key=len)

print("Sorted list by length of elements:", sorted\_list)

**INPUT:** Enter a list of words separated by spaces: 1,23,45,0,78

**OUTPUT:** Sorted list by length of elements: ['1', '0', '23', '45', '78']

**8]To Sort words in alphabetical order:**

words = input("Enter a list of words separated by spaces: ").split()

sorted\_words = sorted(words)

print("Sorted list of words in alphabetical order:", sorted\_words)

**INPUT:** Enter a list of words separated by spaces: red green violet zebra

**OUTPUT:** Sorted list of words in alphabetical order: ['green', 'red', 'violet', 'zebra']

**9]To read a character until a’\*’ is encountered. Also, count the number of uppercase lowercase numbers entered by the user:**

def count\_characters():

uppercase\_count = 0

lowercase\_count = 0

number\_count = 0

print("Enter characters one by one. Enter '\*' to stop:")

while True:

ch = input()

if ch == '\*':

break

elif ch.isupper():

uppercase\_count += 1

elif ch.islower():

lowercase\_count += 1

elif ch.isdigit():

number\_count += 1

print("Number of uppercase letters:", uppercase\_count)

print("Number of lowercase letters:", lowercase\_count)

print("Number of digits:", number\_count)

count\_characters()

**INPUT:** Enter characters one by one. Enter '\*' to stop:

A

v

B

x

p

R

Y

**\***

**OUTPUT:**

Number of uppercase letters: 4

Number of lowercase letters: 3

Number of digits: 0

**10]To remove duplicate item in the list:**

def remove\_duplicates(numbers):

return list(set(numbers))

def remove():

try:

user\_input = input("Enter a list of numbers separated by spaces: ")

numbers = list(map(int, user\_input.split()))

unique\_numbers = remove\_duplicates(numbers)

print("List after removing duplicates:", unique\_numbers)

except ValueError:

print("Please enter a valid list of numbers.")

remove()

**INPUT:** Enter a list of numbers separated by spaces: 1 2 3 4 1 2 3 6 5 7

**OUTPUT:** List after removing duplicates: [1, 2, 3, 4, 5, 6, 7]

**11]Program to find the sum of N numbers entered by the user:**

n=int(input("Enter the value:"))

sum=0

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

sum=sum+i

print(sum)

**INPUT**: Enter the value=5

**OUTPUT**: Sum=15

**12]To print the number of vowels and consonants in the given string:**

def count\_vowels\_and\_consonants(input\_string):

vowels = "aeiouAEIOU"

vowel\_count = 0

consonant\_count = 0

for char in input\_string:

if char in vowels:

vowel\_count += 1

elif char.isalpha():

consonant\_count += 1

return vowel\_count, consonant\_count

input\_string=input('enter the string:')

vowels, consonants = count\_vowels\_and\_consonants(input\_string)

print(f"Number of vowels: {vowels}")

print(f"Number of consonants: {consonants}")

**INPUT:**

enter the string: Hello World!

**OUTPUT:**

Number of vowels: 3

Number of consonants: 7

**13]To find whether two strings have same index and returns the number of matches:**

string1 = input("Enter the first string: ")

string2 = input("Enter the second string: ")

if len(string1) != len(string2):

print("The strings have different lengths.")

else:

match\_count = 0

for i in range(len(string1)):

if string1[i] == string2[i]:

match\_count += 1

print(f"Number of matches at the same index: {match\_count}")

**INPUT:**

Enter the first string: sreeja

Enter the second string: sreeja

**OUTPUT:**

Number of matches at the same index: 6

**14]Write a program to count the words starting “T” in the string.**

def count\_words\_starting\_with\_t(input\_string):

words = input\_string.split()

count = 0

for word in words:

if word.lower().startswith('t'):

count += 1

return count

input\_string = input('enter the string:')

count = count\_words\_starting\_with\_t(input\_string)

print(f"Number of words starting with 'T': {count}")

**INPUT:**

enter the string:THE TIGER IS AT THE TREE

**OUTPUT:**

Number of words starting with 'T': 4

**15] Write a program to convert to the case of the given string.**

def convert\_case(input\_string):

uppercase\_string = input\_string.upper()

lowercase\_string = input\_string.lower()

titlecase\_string = input\_string.title()

return uppercase\_string, lowercase\_string, titlecase\_string

input\_string = input('enter the string:')

uppercase, lowercase, titlecase = convert\_case(input\_string)

print(f"Uppercase: {uppercase}")

print(f"Lowercase: {lowercase}")

print(f"Titlecase: {titlecase}")

**INPUT:**

enter the string: "Hello, World! This is a Test String."

**OUTPUT:**

Uppercase: "HELLO, WORLD! THIS IS A TEST STRING."

Lowercase: "hello, world! this is a test string."

Titlecase: "Hello, World! This Is A Test String."

**16] Write a program to convert the Decimal to Binary**

def decimal\_to\_binary(decimal\_number):

if decimal\_number == 0:

return "0"

binary\_number = ""

while decimal\_number > 0:

remainder = decimal\_number % 2

binary\_number = str(remainder) + binary\_number

decimal\_number = decimal\_number // 2

return binary\_number

decimal\_number = 42

binary\_representation = decimal\_to\_binary(decimal\_number)

print(f"Binary representation of {decimal\_number} is {binary\_representation}")

**INPUT:** Enter the number: 42

**OUTPUT:** Binary representation of 42 squares is 101010

**17] Python Program to create a list of all numbers in a range that are perfect**

import math

def is\_perfect\_square(n):

"""Check if a number is a perfect square."""

root = math.isqrt(n)

return root \* root == n

def perfect\_squares\_in\_range(start, end):

"""Return a list of perfect squares in the given range [start, end]."""

perfect\_squares = []

for number in range(start, end + 1):

if is\_perfect\_square(number):

perfect\_squares.append(number)

return perfect\_squares

start = int(input("Enter the start of the range: "))

end = int(input("Enter the end of the range: "))

perfect\_squares = perfect\_squares\_in\_range(start, end)

print(f"Perfect squares in the range {start} to {end}: {perfect\_squares}")

**INPUT:**

Enter the start of the range: 1

Enter the end of the range: 100

**OUTPUT:**

Perfect squares in the range 1 to 100: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

**18] Write a program to generate Pythagorean Triplets for the given limit.**

import math

def generate\_pythagorean\_triplets(limit):

triplets = []

for a in range(1, limit + 1):

for b in range(a, limit + 1):

c\_square = a\*\*2 + b\*\*2

c = int(math.sqrt(c\_square))

if c \* c == c\_square and c <= limit:

triplets.append((a, b, c))

return triplets

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

pythagorean\_triplets = generate\_pythagorean\_triplets(limit)

print(f"Pythagorean triplets up to {limit}: {pythagorean\_triplets}")

**INPUT:**

Enter the limit: 9

**OUTPUT:**

Pythagorean triplets up to 9: [(3, 4, 5)]