**Name :- Surwade Trisharan Rajesh.**

**RollNo:- 48.**

**Practical Batch no:-04**

**1. Write a Python program to demonstrate i. class variable and instance variable ii. method resolution order**

#class variable and a instance variable

class Car:

wheels=4

def \_\_init\_\_(self):

self.mil = 10

self.make = "BMW"

c1=Car()

c2=Car()

print(c1.mil,c1.make,c1.wheels)

print(c2.mil,c2.make,c1.wheels)

#MRO Method Resolution Order

class A:

def \_\_init\_\_(self):

print('In constructor of class A')

def feature1(self):

print('Feature 1 Working')

class B(A):

def feature3(self):

print('Feature 3 Working')

a1=A()

print('----')

b1=B()

**2. Write a Python program to demonstrate Outer class and Inner class**

class Student: #outer class

def \_\_init\_\_(self,name,rollno):

self.name = name

self.rollno = rollno

self.lap=self.Laptop()

def show(self):

print(self.name,self.rollno)

self.lap.show()

class Laptop: #inner Class

def \_\_init\_\_(self):

self.brand = 'Dell'

self.cpu = 'RYZEN 5'

self.ram = 16

def show(self):

print(self.brand,self.cpu,self.ram)

s1=Student('Trisharan',48)

s2=Student('Himanshu',24)

s1.show()

s2.show()

**3. Write a Python program that matches a string that has an 'a' followed by**

**A) zero or more b's. B) one or more b's.**

**C) zero or one 'b' D) three 'b'. E) two to three 'b'.**

import re

def match\_pattern(input\_string, pattern):

if re.match(pattern, input\_string):

return True

else:

return False

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

pattern\_A = r'ab\*'

pattern\_B = r'ab+'

pattern\_C = r'ab?'

pattern\_D = r'ab{3}'

pattern\_E = r'ab{2,3}'

patterns = [# List of patterns and corresponding descriptions

(pattern\_A, "A) Zero or more 'b's"),

(pattern\_B, "B) One or more 'b's"),

(pattern\_C, "C) Zero or one 'b's"),

(pattern\_D, "D) Three 'b's"),

(pattern\_E, "E) Two to three 'b's")

]

for pattern, description in patterns:

if match\_pattern(input\_string, pattern):

print(f"The string matches pattern {description}.")

else:

print(f"The string does not match pattern {description}.")

**4. Write a Python program that matches a string that has an 'a' followed by anything ending in 'b'.**

import re

def match\_pattern(input\_string):

# Define a regular expression pattern to match the string

pattern = r'a.\*b$'

# Use re.match() to check if the string matches the pattern

if re.match(pattern, input\_string):

return True

else:

return False

# Get input string from the user

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

# Check if the input string matches the pattern

if match\_pattern(input\_string):

print("The string matches the pattern: 'a' followed by anything ending in 'b'.")

else:

print("The string does not match the pattern.")

**5. Write a Python program to match a string that contains only upper and lowercase letters, numbers, and underscores.**

import re

def is\_valid\_string(input\_string):

# Define a regular expression pattern to match the string

pattern = r'^[a-zA-Z0-9\_]+$'

# Use re.match() to check if the string matches the pattern

if re.match(pattern, input\_string):

return True

else:

return False

# Get input string from the user

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

# Check if the input string is valid

if is\_valid\_string(input\_string):

print("The string contains only upper and lowercase letters, numbers, and underscores.")

else:

print("The string contains characters other than upper and lowercase letters, numbers, and underscores.")

**6. Write a program that accepts a string from the user and writes it to a text file. Thereafter, the same program reads the text file and displays it on**

**the screen.**

def write(text):

# Open the file in write mode

with open("text\_file.txt", "w") as file:

# Write the text to the file

file.write(text)

def read():

# Open the file in read mode

with open("text\_file.txt", "r") as file:

# Read the contents of the file

text = file.read()

return text

# Get input string from the user

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

# Write the input string to the text file

write(input\_text)

# Read the text from the file

file\_text = read()

# Display the text on the screen

print("Text read from file:")

print(file\_text)

**7. Write python GUI code using tkinter to convert degree celicus to Farenheight. The formula is farenenheight = (celcius \* 9/5) + 32.**

**Lamda**

import tkinter as tk

from tkinter import messagebox

def convert():

try:

# Get the Celsius temperature from the entry widget

celsius = float(entry\_celsius.get())

# Convert Celsius to Fahrenheit using lambda function

fahrenheit = lambda c: (c \* 9/5) + 32

result = fahrenheit(celsius)

# Display the result in the label widget

label\_result.config(text=f"{celsius}°C = {result}°F")

except ValueError:

# Display error message if input is not a valid number

messagebox.showerror("Error", "Please enter a valid number for Celsius.")

# Create the main application window

root = tk.Tk()

root.title("Celsius to Fahrenheit Converter")

root.geometry("360x200")

# Create label and entry widgets for Celsius input

label\_celsius = tk.Label(root, text="Enter Celsius temperature:")

label\_celsius.pack()

entry\_celsius = tk.Entry(root)

entry\_celsius.pack()

# Create button to trigger conversion

button\_convert = tk.Button(root, text="Convert", command=convert)

button\_convert.pack()

# Create label widget to display result

label\_result = tk.Label(root, text="")

label\_result.pack()

# Run the Tkinter event loop

root.mainloop()

**8. Write a Python program to demonstrate how to catch multiple specific exceptions**

try:

# Try to perform some operations that may raise specific exceptions

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

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

result = num1 / num2

print("Result:", result)

# This operation will raise a ValueError if the input is not an integer

index = int(input("Enter an index to access a list: "))

my\_list = [1, 2, 3]

print("Value at index", index, ":", my\_list[index])

except ZeroDivisionError:

# Catch division by zero error

print("Error: Division by zero!")

except ValueError:

# Catch value error (e.g., when converting input to an integer)

print("Error: Invalid input. Please enter a valid integer.")

except IndexError:

# Catch index error (e.g., when accessing a list with an invalid index)

print("Error: Index out of range. Please enter a valid index.")

except Exception as e:

# Catch any other exceptions

print("An error occurred:", e)

**9. Write python code to demonstrate various constructs available for control structures in python (for and while).**

# For loop example

print("For loop example:")

for i in range(1, 6): # Loop from 1 to 5

print(i, end=" ") # Print numbers separated by space

print("\n")

# While loop example

print("While loop example:")

count = 1

while count <= 5: # Loop until count is less than or equal to 5

print(count, end=" ") # Print count value

count += 1 # Increment count by 1

print("\n")

# Using break statement in a loop

print("Using break statement in a loop:")

for i in range(1, 11): # Loop from 1 to 10

print(i, end=" ")

if i == 5: # If i reaches 5, exit the loop

break

print("\n")

# Using continue statement in a loop

print("Using continue statement in a loop:")

for i in range(1, 11): # Loop from 1 to 10

if i % 2 == 0: # If i is even, skip to the next iteration

continue

print(i, end=" ")

print("\n")

# Nested loop example

print("Nested loop example:")

for i in range(1, 4): # Outer loop

for j in range(1, 4): # Inner loop

print(f"({i}, {j})", end=" ") # Print coordinate pairs

print() # Print new line after each inner loop

print("\n")

# Using else with loops

print("Using else with loops:")

for i in range(1, 6): # Loop from 1 to 5

print(i, end=" ")

else:

print("Loop completed without using break.")

print("\n")

# While loop with else

print("While loop with else:")

count = 1

while count <= 5:

print(count, end=" ")

count += 1

else:

print("Loop completed without using break.")

**10. Write a python code to implement bubble sort and insertion sort**

**Bubble sort:-**

def bubble\_sort(list1):

n=len(list1)

for i in range(n):

for j in range(0,n-i-1):

if list1[j]>list1[j+1]:

list1[j], list1[j+1] = list1[j+1],list1[j]

numlist = list(map(int, input("Enter numbers separated by space: ").split()))

sorted\_list=bubble\_sort(numlist)

print("The sorted list is:")

for i in range(len(numlist)):

print(numlist[i], end=" ")

**#Selection Sort.**

def selection\_sort(list2):

n = len(list2)

for i in range(n):

min\_index = i

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

if list2[j] < list2[min\_index]:

min\_index = j

list2[i], list2[min\_index] = list2[min\_index], list2[i]

return list2

# Take user input

numlist = list(map(int, input("Enter numbers separated by space: ").split()))

# Call the selection\_sort function

sorted\_list = selection\_sort(numlist)

print("The sorted list is:")

for num in sorted\_list:

print(num, end=" ")

**11. Write a python code to implement binary search.**

class BinarySearch:

def \_\_init\_\_(self, A):

self.sa = sorted(A)

def bin\_search(self, x):

front = 1

last = len(self.sa) - 1

while front <= last:

middle = (front + last) // 2

if x > self.sa[middle]:

front = middle + 1

elif x < self.sa[middle]:

last = middle - 1

else:

return middle

return -1

def main():

n = int(input("Enter the number of elements: "))

A = [int(input(f"Enter element {i + 1}: ")) for i in range(n)]

bsearch = BinarySearch(A)

print("The Sorted Element is:", bsearch.sa)

x = int(input("Enter the element to search: "))

result = bsearch.bin\_search(x)

if result != -1:

print(f"Element {x} founded at index {result}.")

else:

print(f"Element {x} not found in the array.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**12. Write python code to demonstrate the use of :**

**A. Anonymous function lambda B. Mapping functions over Sequences**

**C. Functional programing using filter D. Functional programing using reduce**

**A. Anonymous function lambda**

#lambda

# Using lambda to define an anonymous function

addition = lambda x, y: x + y

# Taking user input for two numbers

num1 = float(input("Enter the first number: "))

num2 = float(input("Enter the second number: "))

# Using the lambda function to add the numbers

result = addition(num1, num2)

print("Sum of the numbers:", result)

**B. Mapping functions over Sequences**

#map function

# Using map function to apply a function to each element of a list

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

# Mapping a lambda function to square each element of the list

squared\_numbers = list(map(lambda x: x \*\* 2, numbers))

print("Original list:", numbers)

print("Squared list:", squared\_numbers)

**C. Functional programing using filter.**

#Filter

numbers = list(map(int,input("Enter the numbers in seperated by space").split()))

even\_numbers = list(filter(lambda x: x % 2 == 0,numbers))

print("Original list:", numbers)

print("Even numbers:", even\_numbers)

**D. Functional programing using reduce.**

#functools

from functools import reduce

# Function to calculate the product of two numbers

multiply = lambda x, y: x \* y

# Taking user input for the list of numbers

numbers = list(map(int, input("Enter a list of numbers separated by spaces: ").split()))

# Using reduce to find the product of all numbers in the list

product = reduce(multiply, numbers)

print("List of numbers:", numbers)

print("Product of all numbers:", product)

**13. A program to calculate average marks of n students using a function where n is entered by the user. (Use List)**

def calculate\_average\_marks():

n = int(input("Enter the number of students: "))

marks = []

for i in range(n):

mark = float(input(f"Enter the marks for student {i + 1}: "))

marks.append(mark) # Add the mark to the list

total\_marks = sum(marks)

average = total\_marks / n

print(f"The average marks of {n} students is {average}")

# Call the function

calculate\_average\_marks()

**14. Write a program with a user defined function to count the number of times a character (passed as argument) occurs in the given string.**

def count\_char(string, char):

count = 0

for character in string:

if character == char:

count += 1

return count

string = input("Enter a string: ")

char = input("Enter a character: ")

count = count\_char(string, char)

print(f"The character '{char}' occurs {count} time(s) in the string.")

**15. Write a program to input your friends’ names and their Phone Numbers and store them in the dictionary as the key-value pair. Perform the following operations on the dictionary:**

**a) Display the name and phone number of all your friends**

**b) Add a new key-value pair in this dictionary and display the modified dictionary**

**c) Delete a particular friend from the dictionary**

**d) Modify the phone number of an existing friend**

**e) Check if a friend is present in the dictionary or not**

**f) Display the dictionary in sorted order of names**

def display\_friends(phone\_book):

print("Friends List:")

for name, phone\_number in phone\_book.items():

print(f"Name: {name}, Phone Number: {phone\_number}")

def add\_friend(phone\_book, name, phone\_number):

phone\_book[name] = phone\_number

print("Friend added successfully!")

def delete\_friend(phone\_book, name):

if name in phone\_book:

del phone\_book[name]

print("Friend deleted successfully!")

else:

print("Friend not found in the list.")

def modify\_phone\_number(phone\_book, name, new\_phone\_number):

if name in phone\_book:

phone\_book[name] = new\_phone\_number

print("Phone number modified successfully!")

else:

print("Friend not found in the list.")

def check\_friend(phone\_book, name):

if name in phone\_book:

print(f"{name} is present in the dictionary.")

else:

print(f"{name} is not present in the dictionary.")

def display\_sorted(phone\_book):

sorted\_phone\_book = sorted(phone\_book.items(), key=lambda x: x[0])

print("Sorted Friends List:")

for name, phone\_number in sorted\_phone\_book:

print(f"Name: {name}, Phone Number: {phone\_number}")

def main():

phone\_book = {}

while True:

print("\n1. Display all friends")

print("2. Add a friend")

print("3. Delete a friend")

print("4. Modify phone number")

print("5. Check if friend is present")

print("6. Display sorted friends list")

print("7. Exit")

choice = int(input("Enter your choice: "))

if choice == 1:

display\_friends(phone\_book)

elif choice == 2:

name = input("Enter friend's name: ")

phone\_number = input("Enter friend's phone number: ")

add\_friend(phone\_book, name, phone\_number)

elif choice == 3:

name = input("Enter friend's name to delete: ")

delete\_friend(phone\_book, name)

elif choice == 4:

name = input("Enter friend's name to modify phone number: ")

new\_phone\_number = input("Enter new phone number: ")

modify\_phone\_number(phone\_book, name, new\_phone\_number)

elif choice == 5:

name = input("Enter friend's name to check: ")

check\_friend(phone\_book, name)

elif choice == 6:

display\_sorted(phone\_book)

elif choice == 7:

print("Exiting program...")

break

else:

print("Invalid choice. Please enter a valid option.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**16. Write a program to create a module, use it in other program by renaming it. Demonstrate different ways to use a module in your program.**

# main.py

# Method 1: Import the module directly

#import calculator

# Method 2: Import the module and rename it

import calculator as calc

# Method 3: Import specific functions from the module

from calculator import add as addition, subtract as subtraction, multiply as multiplication, divide as division

# Get user input for numbers

x = float(input("Enter the first number: "))

y = float(input("Enter the second number: "))

# Using the module directly

print("\nUsing the calculator module directly:")

print("Addition:", calculator.add(x, y))

print("Subtraction:", calculator.subtract(x, y))

print("Multiplication:", calculator.multiply(x, y))

print("Division:", calculator.divide(x, y))

# Using the module with a renamed alias

print("\nUsing the calculator module with a renamed alias:")

print("Addition:", calc.add(x, y))

print("Subtraction:", calc.subtract(x, y))

print("Multiplication:", calc.multiply(x, y))

print("Division:", calc.divide(x, y))

# Using specific functions from the module

print("\nUsing specific functions from the calculator module:")

print("Addition:", addition(x, y))

print("Subtraction:", subtraction(x, y))

print("Multiplication:", multiplication(x, y))

print("Division:", division(x, y))

**17. Write a program using a user defined function to check if a string is a palindrome or not. (A string is called palindrome if it reads same backwards as forward. For example, Kanak is a palindrome.)**

def is\_palindrome(s):

#Function to check if a string is a palindrome or not.

# Convert the string to lowercase to handle case-insensitive palindromes

s = s.lower()

# Remove spaces from the string

s = s.replace(" ", "")

# Compare the string with its reverse

return s == s[::-1]

# Get user input

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

# Check if the input string is a palindrome

if is\_palindrome(user\_input):

print(f"{user\_input} is a palindrome string.")

else:

print(f"{user\_input} is not a palindrome string.")