INDEX

Sr. No.	Practical Name	Teacher Sign
1		
2		
3		
4		
5		
6		
7		

Program 1(a): Write a Python Program to Calculate the Area of a Triangle.

Solution:

```
base = float(input("Enter the base of the triangle: "))
height = float(input("Enter the height of the triangle: "))
area = 0.5 * base * height
print(f"The area of the triangle is: {area}")
```

Output 1(a):

```
Enter the base of the triangle: 3
Enter the height of the triangle: 4
The area of the triangle is: 6.0
```

Program 1(b): Write a Python Program to Swap Two Variables.

Solution:

```
a = input("Enter the first variable (a): ")
b = input("Enter the second variable (b): ")
a, b = b, a
print(f"After swapping: a = {a}, b = {b}")
```

Output1(b):

```
Enter the first variable (a): 7
Enter the second variable (b): 8
After swapping: a = 8, b = 7
```

Program 1(c): Write a Python Program to Convert Celsius to Fahrenheit.

Solution:

```
celsius = float(input("Enter temperature in Celsius: "))
fahrenheit = (celsius * 9/5) + 32
print(f"{celsius} degrees Celsius is equal to {fahrenheit} degrees Fahrenheit.")
```

Output:

```
Enter temperature in Celsius: 34
34.0 degrees Celsius is equal to 93.2 degrees Fahrenheit.
```

Program 2(a):Write a Python Program to Check if a Number is Odd or Even.

Solution:

```
number = int(input("Enter a number: "))
if number % 2 == 0:
    print(f"{number} is even.")
else:
    print(f"{number} is odd.")
```

Output2(a):

```
Enter a number: 34
34 is even.
```

Program 2(b): Write a Python Program to Check if a Number is Positive, Negative or 0.

Solution:

```
number = float(input("Enter a number: "))
if number > 0:
    print(f"{number} is positive.")
elif number < 0:
    print(f"{number} is negative.")
else:
    print("The number is zero.")
```

Output2(b):

```
Enter a number: -29
-29.0 is negative.
```

Program 2(c): Write a Python Program to Check Armstrong Number

Solution:

```
number = int(input("Enter a number: "))
temp = number
sum_of_powers = 0
num_digits = len(str(number))

while temp > 0:
    digit = temp % 10
    sum_of_powers += digit ** num_digits
    temp //= 10
if sum_of_powers == number:
    print(f"{number} is an Armstrong number.")
else:
    print(f"{number} is not an Armstrong number.")
```

Output2(c):

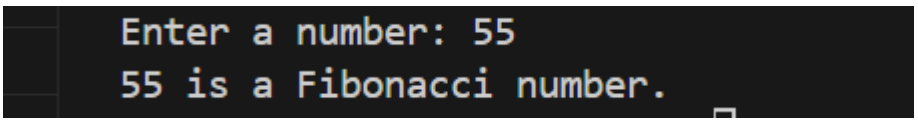
```
Enter a number: 371
371 is an Armstrong number.
```

Program 3(a): Write a Python program to check if a given number is Fibonacci number?

Solution:

```
def is_fibonacci(n):
     if n < 0:
          return false
    a,b  =  0,1
    while b < n:
          a,b=b,a+b
  return b ==n
num=int(input("enter a number: ")
if is_fibonacci(num):
        print(f"{num} is a Fibonacci number.")
else:
        print(f"{num} is not a Fibonacci number.")
```

Output3(a):

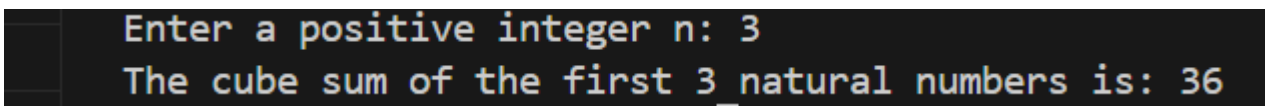


Program 3(b): Write a Python program to print cube sum of first n natural numbers.

Solution:

```
n = int(input("Enter a positive integer n: "))
cube_sum = 0
for i in range(1, n + 1):
cube_sum += i ** 3
print(f"The cube sum of the first {n} natural numbers is: {cube_sum}")
```

Output 3(b):



Program 3(c): Write a Python program to print all odd numbers in a range.

Solution:

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

print("Odd numbers in the range:")
for number in range(start, end + 1):
    if number % 2 != 0:
        print(number)
```

Output 3(c):

```
Enter the start of the range: 7
Enter the end of the range: 21
Odd numbers in the range:
7
9
11
15
17
19
21
```

Program 4(a):Write a Python Program to Print Pascal Triangle
Hint: Enter number of rows: 4

```
      1
    1   1
  1   2   1
1   3   3   1
```

Solution4(a):

```
def pascal_triangle(rows):
    for i in range(rows):
        print(" "*(rows-i-1),end="")
        value=1
        for j in range(i+1):
            print(value,end="")
            value=value*(i-j)//(j+1)
        print()
num_rows=int(input("enter a  number of rows:")
pascal_triangle(num_rows)
```

Output4(a):

```
Enter number of rows: 7
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
1 6 15 20 15 6 1
```

Program 4(b): WAP to draw the following Pattern for n number:

1 1 1 1 1
2 2 2 2
3 3 3
4 4
5

Solution:

```
n = int(input("Enter a number: "))
for i in range(1, n + 1):
    for j in range(n - i + 1):  columns
        print(i, end="")
print()
```

Output4(b):

```
Enter a number: 6
1 1 1 1 1 1
2 2 2 2 2
3 3 3 3
4 4 4
5 5
6
```

Program 5: Write a program with a function that accepts a string from keyboard and create a new string after converting character of each word capitalized. For instance, if the sentence is "stop and smell the roses" the output should be "Stop And Smell The Roses"

Solution:

```
def capitalize_words(sentence):
    return' '.join(word.capitalize()for word in sentence.split())

input_string = input("Enter a sentence: ")
output_string = capitalize_words(input_string)
print("Capitalized Sentence:",output_string)
```

Output5:

```
Enter a sentence: stop and smell the roses
Capitalized Sentence: Stop And Smell The Roses
```

Program 6(a):Write a program that accepts a list from user. Your program should reverse the content of list and display it. Do not use reverse () method.

Solution:

```
user_input = input("Enter a list of elements separated by spaces: ")
elements = user_input.split()  # Split the input string into a list

# Initialize an empty list to hold the reversed elements
reversed_list = []

# Loop through the original list in reverse order
for i in range(len(elements) - 1, -1, -1):
reversed_list.append(elements[i])  # Append each element to the new list

# Display the reversed list
print("Reversed list:", reversed_list)
```

Output6(a):

```
Enter a list of elements separated by spaces: 1 2 3 4
Reversed list: ['4', '3', '2', '1']
```

Program 6(b): Find and display the largest number of a list without using built-in function max (). Your program should ask the user to input values in list from keyboard.

Solution:

```
user_input = input("Enter a list of numbers separated by spaces: ")
numbers = user_input.split()  # Split the input string into a list of strings

# Convert strings to integers
numbers = [int(num) for num in numbers]

# Initialize largest number to the first element
largest_number = numbers[0]

# Loop through the list to find the largest number
for num in numbers:
    if num >largest_number:
largest_number = num  # Update largest number

# Display the largest number
print("The largest number in the list is:", largest_number)
```

Output6(b):

```
Enter a list of numbers separated by spaces: 67 67 89 65
The largest number in the list is: 89
```

Program 7: Find the sum of each row of matrix of size m x n. For example, for the following matrix output will be like this:

```
2  11  7  12
5  2   9  15
8  3   10 42
```

Sum of row 1 = 32
Sum of row 2 = 31
Sum of row 3 = 63

Solution:

```
def row_sums(matrix):
    for i, row in enumerate(matrix, start=1):
        print(f"Sum of row {i} = {sum(row)}")
m = int(input("Enter the number of rows (m): "))
n = int(input("Enter the number of columns (n): "))
matrix = []
print("Enter the elements of the matrix:")
for i in range(m):
    row = list(map(int, input(f"Enter elements of row {i+1}, separated by spaces: ").split()))
    matrix.append(row)
row_sums(matrix)
```

Output7:

```
Enter the number of rows (m): 4
Enter the number of columns (n): 3
Enter the elements of the matrix:
Enter elements of row 1, separated by spaces: 4 5 6
Enter elements of row 2, separated by spaces: 8 7 4
Enter elements of row 3, separated by spaces: 1 4 2
Enter elements of row 4, separated by spaces: 9 6 4
Sum of row 1 = 15
Sum of row 2 = 19
Sum of row 3 = 7
Sum of row 4 = 19
```

Program 8(a): Write a program that reads a string from keyboard and display:

*** The number of uppercase letters in the string.**

*** The number of lowercase letters in the string.**

*** The number of digits in the string.**

*** The number of whitespace characters in the string.**

Solution:

```
def Count(str):
 upper, lower, space = 0, 0, 0
 for i in range(len(str)):
    if str[i].isupper():
        upper += 1
    elif str[i].islower():
         lower += 1
    elif str[i] == " ":
        space += 1
 print('Upper case letters:', upper)
 print('Lower case letters:', lower)
 print('Spaces:', space)
str = input()
print("Length of string:" , len(str))
Count(str)
```

Output 8(a):

```
Length of string: 130
Upper case letters: 34
Lower case letters: 67
Spaces: 4
```

Program 8(b): Python Program to Find Common Characters in Two Strings

Solution:

```
string1 = input("Enter the first string: ")
string2 = input("Enter the second string: ")
common_characters = set(string1) & set(string2)
if common_characters:
print("Common characters:", ' '.join(common_characters))
else:
print("No common characters found.")
```

Output8(b):

```
Enter the first string: python coding
Enter the second string: on anaconda navigator
Common characters: t g n o i d c
```

Program 8(c): Python Program to Count the Number of Vowels in a String.

Solution:

```
input_string = input("Enter a string: ")
vowels = 'aeiouAEIOU'
vowel_count = 0
for char in input_string:
    if char in vowels:
vowel_count += 1
print(f"Number of vowels in the string: {vowel_count}")
```

Output8(c):

```
Enter a string: My name is khushpreet kaur
Number of vowels in the string: 8
```

Program 9(a): Write a Python program to check if a specified element presents in a tuple of tuples.
Original list:
((‘Red’ ,’White’ , ‘Blue’),(‘Green’, ’Pink’ , ‘Purple’), (‘Orange’, ‘Yellow’, ‘Lime’))
Check if White present in said tuple of tuples!
True
Check if Olive present in said tuple of tuples!
False

Solution:

```
tuple_of_tuples = (('Red', 'White', 'Blue'), ('Green', 'Pink', 'Purple'), ('Orange', 'Yellow', 'Lime'))
element_to_check_1 = 'White'
element_to_check_2 = 'Olive'
is_present_1 = any(element_to_check_1 in tup for tup in tuple_of_tuples)
is_present_2 = any(element_to_check_2 in tup for tup in tuple_of_tuples)
print(f"Check if '{element_to_check_1}' present in said tuple of tuples: {is_present_1}")
print(f"Check if '{element_to_check_2}' present in said tuple of tuples: {is_present_2}")
```

Output: 9(a):

```
Check if 'White' present in said tuple of tuples: True
Check if 'Olive' present in said tuple of tuples: False
```

Program 9(b): Write a Python program to remove an empty tuple(s) from a list of tuples.
Sample data: [(), (), (''), ('a', 'b'), ('a', 'b', 'c'), ('d')]
Expected output: [(''), ('a', 'b'), ('a', 'b', 'c'), 'd']

Solution:

```
sample_data = [(), (), (''), ('a', 'b'), ('a', 'b', 'c'), ('d',)]
cleaned_data = [tup for tup in sample_data if tup]  # List comprehension to filter out empty tuples
print("Expected output:", cleaned_data)
```

Output9(b):

```
rs/STAR COMPUTER/Desktop/python1stprogram.py"
Expected output: [('',), ('a', 'b'), ('a', 'b', 'c'), ('d',)]
```

Program 10: Write a Program in Python to Find the Differences Between Two Lists Using Sets.

Solution:

```
list1_input = input("Enter the first list of elements separated by spaces: ")
list1 = list1_input.split()  # Convert input string to a list
list2_input = input("Enter the second list of elements separated by spaces: ")
list2 = list2_input.split()  # Convert input string to a list
set1 = set(list1)
set2 = set(list2)
difference_list1 = set1 - set2  # Elements in list1 but not in list2
difference_list2 = set2 - set1  # Elements in list2 but not in list1
print("Elements in the first list but not in the second:", difference_list1)
print("Elements in the second list but not in the first:", difference_list2)
```

Output 10:

```
Enter the first list of elements separated by spaces: 1 2 3 4
Enter the second list of elements separated by spaces: 5 6 3 2
Elements in the first list but not in the second: {'4', '1'}
Elements in the second list but not in the first: {'5', '6'}
```

Program 11(a):Write a Python program Remove duplicate values across Dictionary Values.

Input :test_dict = {'Manjeet': [1], 'Akash': [1, 8, 9]}
Output : {'Manjeet': [], 'Akash': [8, 9]}
Input :test_dict = {'Manjeet': [1, 1, 1], 'Akash': [1, 1, 1]}
Output : {'Manjeet': [], 'Akash': []}

Solution:

```
test_dict1 = {'Manjeet': [1], 'Akash': [1, 8, 9]}
test_dict2 = {'Manjeet': [1, 1, 1], 'Akash': [1, 1, 1]}
unique_values = set()
for values in test_dict1.values():
    unique_values.update(values)
for key in test_dict1.keys():
   test_dict1[key] = [value for value in test_dict1[key] if value not in unique_values or
unique_values.remove(value)]
print("Output for test_dict1:", test_dict1)
   unique_values = set()
for values in test_dict2.values():
unique_values.update(values)
for key in test_dict2.keys():
   test_dict2[key] = [value for value in test_dict2[key] if value not in unique_values or
unique_values.remove(value)]
print("Output for test_dict2:", test_dict2)
```

Output11(a):

```
Output for test_dict1: {'Manjeet': [], 'Akash': [1]}
Output for test_dict2: {'Manjeet': [1, 1], 'Akash': [1, 1, 1]}
```

Program 11(b): Write a Python program to Count the frequencies in a list using dictionary in Python.

Input : [1, 1, 1, 5, 5, 3, 1, 3, 3, 1,4, 4, 4, 2, 2, 2, 2]
Output :
1 : 5
2 : 4
3 : 3
4 : 3
5 : 2

Explanation : Here 1 occurs 5 times, 2 occurs 4 times and so on...

Solution:

```
input_list = [1, 1, 1, 5, 5, 3, 1, 3, 3, 1, 4, 4, 4, 2, 2, 2, 2]
frequency_dict = {}
for number in input_list:
    if number in frequency_dict:
        frequency_dict[number] += 1  # Increment count
    else:
        frequency_dict[number] = 1     # Initialize count
for number, count in frequency_dict.items():
    print(f"{number} : {count}")
```

Output11(b):

```
1 : 5
5 : 2
3 : 3
4 : 3
2 : 4
```

Program 12(a): Write a Python Program to Capitalize First Letter of Each Word in a File.

Solution:

```
filename = 'input.txt'
with open(filename, 'r') as file:
    content = file.readlines()  # Read all lines
capitalized_content = []
for line in content:
capitalized_line = ' '.join(word.capitalize() for word in line.split())
capitalized_content.append(capitalized_line)
with open('output.txt', 'w') as file:
file.write('\n'.join(capitalized_content))
print("Capitalized content has been written to 'output.txt'.")
```

Output12(a):

```
Capitalized content: This Is First Line. This Is Second Line. This
Is Third Line.
```

Program 12(b): Write a Python Program to Print the Contents of File in Reverse Order.

Solution:

```
filename = 'input.txt'
with open(filename, 'r') as file:
    content = file.readlines()  # Read all lines
for line in reversed(content):
    print(line.strip())
```

Output12(b):

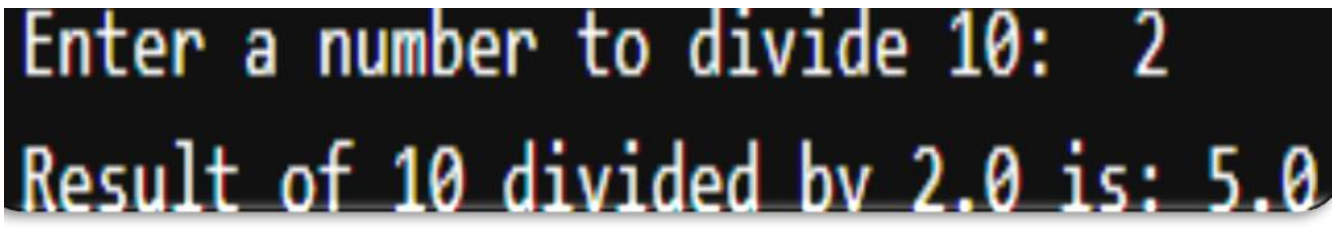
```
This is third line.
This is second line.
This is first line.
```

Program 13: WAP
to catch an exception and handle it using try and except code blocks._

Solution:

```
try:
    # Prompt the user to enter a number
user_input = input("Enter a number to divide 10: ")
    number = float(user_input)  # Convert the input to a float

    # Attempt to perform the division
    result = 10 / number
print(f"Result of 10 divided by {number} is: {result}")

except ValueError:
    # Handle the case where conversion to float fails
print("Error: Please enter a valid number.")

except ZeroDivisionError:
    # Handle the case where the user tries to divide by zero
print("Error: Division by zero is not allowed.")

except Exception as e:
    # Catch any other exceptions that may occur
print(f"An unexpected error occurred: {e}")
```

Output 13:



Program 14: Write a Python Program to Append, Delete and Display Elements of a List using Classes.

Solution:

```
class ListManager:
    def __init__(self):
self.elements = []  # Initialize an empty list
manager = ListManager()
manager.elements.append(10)
print(f"Element '10' appended.")
manager.elements.append(20)
print(f"Element '20' appended.")
manager.elements.append(30)
print(f"Element '30' appended.")
if manager.elements:
print("Current elements in the list:")
    for elem in manager.elements:
        print(elem)
else:
print("The list is empty.")
element_to_delete = 20
if element_to_delete in manager.elements:
manager.elements.remove(element_to_delete)
print(f"Element '{element_to_delete}' deleted.")
else:
print(f"Error: Element '{element_to_delete}' not found in the list.")
if manager.elements:
print("Current elements in the list:")
    for elem in manager.elements:
        print(elem)
else:
print("The list is empty.")
element_to_delete = 40
if element_to_delete in manager.elements:
manager.elements.remove(element_to_delete)
print(f"Element '{element_to_delete}' deleted.")
else:
print(f"Error: Element '{element_to_delete}' not found in the list.")
if manager.elements:
print("Final elements in the list:")
    for elem in manager.elements:
        print(elem)
else:
print("The list is empty.")
```

```
1. Append Element
2. Delete Element
3. Display Elements
4. Quit
Enter your choice (1/2/3/4): 1
Enter the element to append: 6
Element 6 appended successfully.

1. Append Element
2. Delete Element
3. Display Elements
4. Quit
Enter your choice (1/2/3/4): 2
Enter the element to delete: 2
Element 2 not found in the list.

1. Append Element
2. Delete Element
3. Display Elements
4. Quit
Enter your choice (1/2/3/4): 3
Elements in the list:
6

1. Append Element
2. Delete Element
3. Display Elements
4. Quit
Enter your choice (1/2/3/4): 4
Exiting the program.
```

Program 15: Write a Python Program to Find the Area and Perimeter of the Circle using Class.

Solution:

```
import math

class Circle:
    def __init__(self, radius):
self.radius = radius  # Initialize the radius

    def area(self):
        return math.pi * (self.radius ** 2)  # Calculate area

    def perimeter(self):
        return 2 * math.pi * self.radius  # Calculate perimeter (circumference)
radius = float(input("Enter the radius of the circle: "))
circle = Circle(radius)
circle_area = circle.area()
circle_perimeter = circle.perimeter()
print(f"Area of the circle: {circle_area:.2f}")
print(f"Perimeter (Circumference) of the circle: {circle_perimeter:.2f}")
```

Output 15:

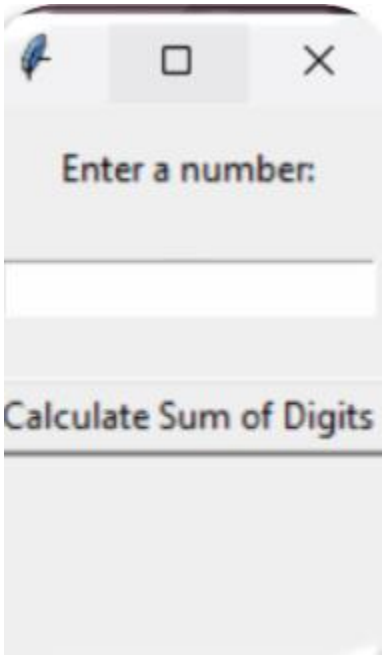
```
Enter the radius of the circle:  4
Area of the circle: 50.27
Perimeter (Circumference) of the circle: 25.13
```

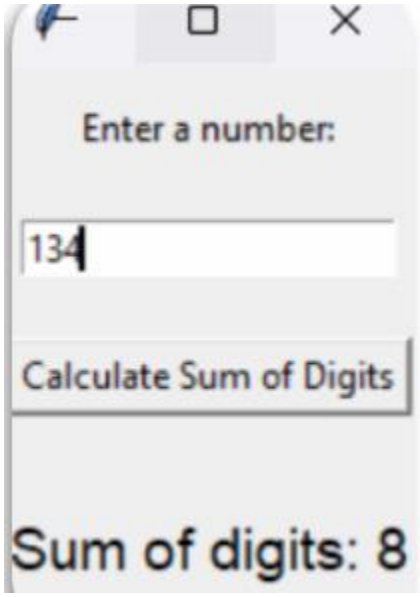
Program 16: Create an interactive application using Python's Tkinter library for graphics programming.

Solution:

```
import tkinter as tk
class SumOfDigitsApp:
    def __init__(self, root):
self.root = root
self.root.title("Sum of Digits Calculator")
self.label = tk.Label(root, text="Enter a number:")
self.label.pack(pady=10)
self.entry = tk.Entry(root)
self.entry.pack(pady=10)
self.calculate_button = tk.Button(root, text="Calculate Sum of Digits",
command=self.calculate_sum)
self.calculate_button.pack(pady=10)
self.result_label = tk.Label(root, text="", font=('Helvetica', 14))
self.result_label.pack(pady=20)
  def calculate_sum(self):
        # Get the input number
number_str = self.entry.get()

        try:
digit_sum = sum(int(digit) for digit in number_str if digit.isdigit())
self.result_label.config(text=f"Sum of digits: {digit_sum}")
        except ValueError:
self.result_label.config(text="Please enter a valid number.")

if __name__ == "__main__":
    root = tk.Tk()
    app = SumOfDigitsApp(root)
root.mainloop()
```

Output 16:


Enter a number:
Calculate Sum of Digits


Enter a number:
134
Calculate Sum of Digits
Sum of digits: 8