

Name:- Gayatri Gorakh Jadhav

Roll No:- CS024

Date:- _____

Practical No:1

AIM:- Write a python program to display all types of pyramids of stars

CODE:-

```
def print_right_angle_triangle(n):
    print("Right-Angle Triangle:")
    for i in range(1, n + 1):
        print('*' * i)
    print()

def print_isosceles_triangle(n):
    print("Isosceles Triangle:")
    for i in range(1, n + 1):
        print(' ' * (n - i) + '*' * (2 * i - 1))
    print()

def print_inverted_triangle(n):
    print("Inverted Triangle:")
    for i in range(n, 0, -1):
        print('*' * i)
    print()

def print_full_pyramid(n):
    print("Full Pyramid:")
    for i in range(1, n + 1):
        print(' ' * (n - i) + '*' * (2 * i - 1))
    print()

def print_diamond(n):
    print("Diamond Shape:")
    # Upper part
    for i in range(1, n + 1):
        print(' ' * (n - i) + '*' * (2 * i - 1))
    # Lower part
    for i in range(n - 1, 0, -1):
        print(' ' * (n - i) + '*' * (2 * i - 1))
    print()

# Set the height of the pyramids
n = 5
```

```
print_right_angle_triangle(n)
print_isosceles_triangle(n)
print_inverted_triangle(n)
print_full_pyramid(n)
print_diamond(n)
```

Output:-

Right-Angle Triangle:

```
*
**
***
****
*****
```

Isosceles Triangle:

```
  *
 ***
*****
*****
*****
```

Inverted Triangle:

```
*****
****
***
**
*
```

Full Pyramid:

```
  *
 ***
*****
*****
*****
```

Diamond Shape:

```
  *
 ***
*****
*****
*****
*****
***
  *
```

Name:- Gayatri Gorakh Jadhav
Roll No:- CS024

Date:-_____

Practical No:2

AIM:- Write a program to display multiplication table of all numbers from 1 to 10.

CODE:-

```
# Loop through numbers 1 to 10 for the multiplication table
for i in range(1, 11):
    # Print a header for the current multiplication table
    print("\n\nMULTIPLICATION TABLE FOR %d\n" % (i))
    # Loop through numbers 1 to 10 for the multipliers
    for j in range(1, 11):
        # Print the multiplication expression and result with formatted output
        print("%-5d X %-5d = %-5d" % (i, j, i * j))
```

Output:-

MULTIPLICATION TABLE FOR 1 MULTIPLICATION TABLE FOR 4 MULTIPLICATION TABLE FOR 7

1	X	1 =	1	4	X	1 =	4	7	X	1 =	7
1	X	2 =	2	4	X	2 =	8	7	X	2 =	14
1	X	3 =	3	4	X	3 =	12	7	X	3 =	21
1	X	4 =	4	4	X	4 =	16	7	X	4 =	28
1	X	5 =	5	4	X	5 =	20	7	X	5 =	35
1	X	6 =	6	4	X	6 =	24	7	X	6 =	42
1	X	7 =	7	4	X	7 =	28	7	X	7 =	49
1	X	8 =	8	4	X	8 =	32	7	X	8 =	56
1	X	9 =	9	4	X	9 =	36	7	X	9 =	63
1	X	10 =	10	4	X	10 =	40	7	X	10 =	70

MULTIPLICATION TABLE FOR 2 MULTIPLICATION TABLE FOR 5 MULTIPLICATION TABLE FOR 8

2	X	1 =	2	5	X	1 =	5	8	X	1 =	8
2	X	2 =	4	5	X	2 =	10	8	X	2 =	16
2	X	3 =	6	5	X	3 =	15	8	X	3 =	24
2	X	4 =	8	5	X	4 =	20	8	X	4 =	32
2	X	5 =	10	5	X	5 =	25	8	X	5 =	40
2	X	6 =	12	5	X	6 =	30	8	X	6 =	48
2	X	7 =	14	5	X	7 =	35	8	X	7 =	56
2	X	8 =	16	5	X	8 =	40	8	X	8 =	64
2	X	9 =	18	5	X	9 =	45	8	X	9 =	72
2	X	10 =	20	5	X	10 =	50	8	X	10 =	80

MULTIPLICATION TABLE FOR 3 MULTIPLICATION TABLE FOR 6 MULTIPLICATION TABLE FOR 9 MULTIPLICATION TABLE FOR 10

3	X	1 =	3	6	X	1 =	6	9	X	1 =	9	10	X	1 =	10
3	X	2 =	6	6	X	2 =	12	9	X	2 =	18	10	X	2 =	20
3	X	3 =	9	6	X	3 =	18	9	X	3 =	27	10	X	3 =	30
3	X	4 =	12	6	X	4 =	24	9	X	4 =	36	10	X	4 =	40
3	X	5 =	15	6	X	5 =	30	9	X	5 =	45	10	X	5 =	50
3	X	6 =	18	6	X	6 =	36	9	X	6 =	54	10	X	6 =	60
3	X	7 =	21	6	X	7 =	42	9	X	7 =	63	10	X	7 =	70
3	X	8 =	24	6	X	8 =	48	9	X	8 =	72	10	X	8 =	80
3	X	9 =	27	6	X	9 =	54	9	X	9 =	81	10	X	9 =	90
3	X	10 =	30	6	X	10 =	60	9	X	10 =	90	10	X	10 =	100

Name:- Gayatri Gorakh Jadhav
Roll No:- CS024

Date:-_____

Practical No: 3

AIM:- Write a program to calculate simple interest except amount, duration and rate of interest from user.

CODE:-

```
# Function to calculate simple interest
def calculate_simple_interest(principal, rate, duration):
    # Calculate simple interest using the formula: (P * R * T) / 100
    return (principal * rate * duration) / 100

# Prompt the user to enter the principal amount and convert it to a float
principal = float(input("Enter the principal amount: "))

# Prompt the user to enter the duration in years and convert it to a float
duration = float(input("Enter the duration (in years): "))

# Prompt the user to enter the rate of interest and convert it to a float
rate = float(input("Enter the rate of interest (in %): "))

# Calculate simple interest using the user inputs
simple_interest = calculate_simple_interest(principal, rate, duration)

# Print the calculated simple interest, formatted to two decimal places
print(f"The simple interest is: {simple_interest:.2f}")
```

Output:-

```
Enter the principal amount: 200000
Enter the duration (in years): 2
Enter the rate of interest (in %): 7
The simple interest is: 28000.00
```

Name:- Gayatri Gorakh Jadhav
Roll No:- CS024

Date:-_____

Practical No: 4

AIM:- Write a program to count even and odd number in the list.

CODE:-

```
# Function to count even and odd numbers in a list
def count_even_odd(numbers):
    # Initialize counters for even and odd numbers
    even_count = 0
    odd_count = 0

    # Iterate through each number in the provided list
    for number in numbers:
        # Check if the number is even
        if number % 2 == 0:
            even_count += 1 # Increment even count
        else:
            odd_count += 1 # Increment odd count

    # Return the counts of even and odd numbers
    return even_count, odd_count

# List of numbers from 1 to 20
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

# Call the function and store the results in even_count and odd_count
even_count, odd_count = count_even_odd(numbers)

# Print the count of even numbers
print(f"Even numbers: {even_count}")

# Print the count of odd numbers
print(f"Odd numbers: {odd_count}")
```

Output:-

Even numbers: 10

Odd numbers: 10

Name:- Gayatri Gorakh Jadhav
Roll No:- CS024

Date:-_____

Practical No: 5

AIM:- Write a program to find sum of all numbers, mean, max, average of numbers in a list.

CODE:-

```
from collections import Counter
def calculate_statistics(numbers):
    if not numbers:
        return None, None, None, None, None
    # Calculate sum
    total_sum = sum(numbers)
    # Calculate minimum
    minimum = min(numbers)

    # Calculate maximum
    maximum = max(numbers)

    # Calculate mean
    mean = total_sum / len(numbers)
    # Calculate mode
    frequency = Counter(numbers)
    mode_data = frequency.most_common()
    mode = [num for num, freq in mode_data if freq == mode_data[0][1]]
    return total_sum, minimum, maximum, mean, mode
# Example usage
numbers = [1, 2, 2, 3, 4, 4, 4, 5]
total_sum, minimum, maximum, mean, mode = calculate_statistics(numbers)
print(f"Sum: {total_sum}")
print(f"Min: {minimum}")
print(f"Max: {maximum}")
print(f"Mean: {mean}")
print(f"Mode: {mode}")
```

Output:-

```
Sum: 25
Min: 1
Max: 5
Mean: 3.125
Mode: [4]
|
```

