PROJECT REPORT

FOR

PYTHON LAB

Submitted By:

ABHINANDAN DUTTA

(CS23BCAGN018)

BCA 4th SEMESTER

SCHOOL OF COMPUTING SCIENCES

THE ASSAM KAZIRANGA UNIVERSITY, JORHAT, ASSAM

19 MAY, 2025

CONTENT

INTRODUCTION

This Python project covers a range of basic to intermediate programming tasks such as math operations, equations solving, visualization, and game development using Tkinter.

a. WAP using python implementation of any Arithmetic and quadratic operations.

```python
# 1.WAP using python implementation of any arithematic and quadratic operations
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))

print("Addition:", a + b)
print("Subtraction:", a - b)
print("Multiplication:", a * b)
if b != 0:
    print("Division:", a / b)
else:
    print("Division: Cannot divide by zero")

print("\nSolve Quadratic Equation ax^2 + bx + c = 0")
a = float(input("Enter a: "))
b = float(input("Enter b: "))
c = float(input("Enter c: "))

d = b**2 - 4*a*c  # Discriminant

if d > 0:
    root1 = (-b + d**0.5) / (2*a)
    root2 = (-b - d**0.5) / (2*a)
    print("Two Real Roots:", root1, "and", root2)
elif d == 0:
    root = -b / (2*a)
    print("One Real Root:", root)
else:
    print("No Real Roots (Complex roots)")
```

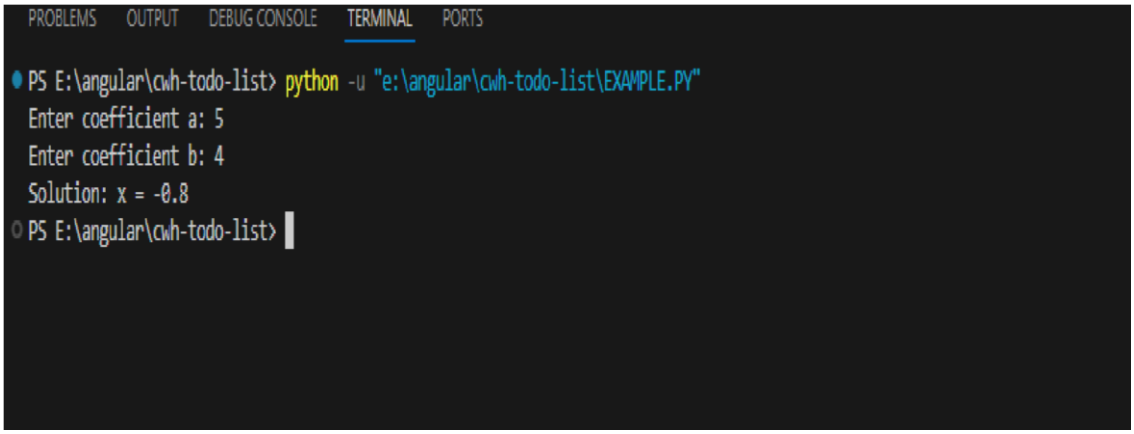
OUTPUT:-

```
PS E:\angular\cwh-todo-list> python -u "e:\angular\cwh-todo-list\EXAMPLE.PY"
Enter first number: 5
Enter second number: 6
Addition: 11
Subtraction: -1
Enter first number: 5
Enter second number: 6
Addition: 11
Subtraction: -1
Enter second number: 6
Addition: 11
Subtraction: -1
Addition: 11
Subtraction: -1
Subtraction: -1
Multiplication: 30
Division: 0.8333333333333334


Solve Quadratic Equation ax^2 + bx + c = 0
Solve Quadratic Equation ax^2 + bx + c = 0
Enter a: 3
Enter b: 4
Enter c: 5
Enter b: 4
Enter c: 5
Enter c: 5
No Real Roots (Complex roots)
PS E:\angular\cwh-todo-list>
```

b. Implementation of linear equation.

```
# 2.Implementation of linear equation.
## Solve linear equation: ax + b = 0

a = float(input("Enter coefficient a: "))
b = float(input("Enter coefficient b: "))

if a != 0:
    x = -b / a
    print("Solution: x =", x)
else:
    if b == 0:
        print("Infinite solutions (Every value of x satisfies the equation)")
    else:
        print("No solution (Equation is inconsistent)")
```

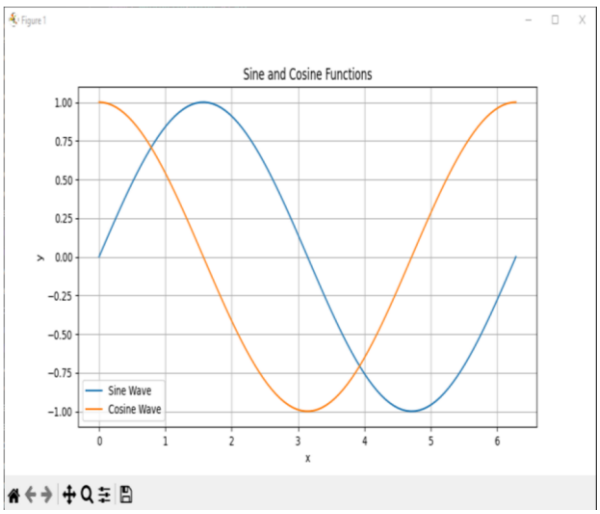
OUTPUT:-

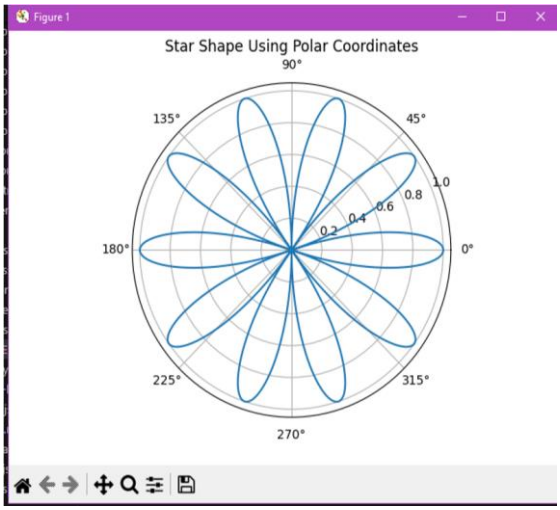


c. Using any mathematical function or equation to geographical representation like star, graph (more complex implementation can be given promisable marks)

```python
import matplotlib.pyplot as plt
import numpy as np

## 1. Sine and Cosine Waves
x = np.linspace(0, 2*np.pi, 100)
y_sin = np.sin(x)
y_cos = np.cos(x)

plt.figure(figsize=(10, 5))
plt.plot(x, y_sin, label='Sine Wave')
plt.plot(x, y_cos, label='Cosine Wave')
plt.title("Sine and Cosine Functions")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.grid(True)
plt.show()


## 2. Star Pattern using polar coordinates
theta = np.linspace(0, 2*np.pi, 1000)
r = np.abs(np.cos(5 * theta))  # Star shape with 5 points

plt.figure()
plt.polar(theta, r)
plt.title("Star Shape Using Polar Coordinates")
plt.show()
```

Output:-





d.WAP to implement function

EXAMPLE.py > ...

```python
# 4.wap to implement function
def add(a, b):
    return a + b

def subtract(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    if b != 0:
        return a / b
    else:
        return "Cannot divide by zero"

x = float(input("Enter first number: "))
y = float(input("Enter second number: "))

print("Addition:", add(x, y))
print("Subtraction:", subtract(x, y))
print("Multiplication:", multiply(x, y))
print("Division:", divide(x, y))
```

Output:-

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Code

```
Enter first number: 4
Enter second number: 5
Addition: 9.0
Subtraction: -1.0
Multiplication: 20.0
Division: 0.8
PS E:\angular\cwh-todo-list>
```

e. Using tinker make any formatted application according to your ideas(Tretis, snake, cardblock)

EXAMPLE.py > ...

```python
import tkinter as tk
import random

# Constants
WIDTH = 400
HEIGHT = 400
SPEED = 100  # Snake speed (milliseconds)
SIZE = 20    # Snake block size

# Snake game class
class SnakeGame:
    def __init__(self, master):
        self.master = master
        self.canvas = tk.Canvas(master, width=WIDTH, height=HEIGHT, bg="black")
        self.canvas.pack()

        self.snake = [(100, 100), (80, 100), (60, 100)]  # Starting snake
        self.food = self.place_food()

        self.direction = "Right"
        self.running = True

        self.draw_snake()
        self.draw_food()

        self.master.bind("<Key>", self.change_direction)
        self.move_snake()

    def draw_snake(self):
        self.canvas.delete("snake")
        for x, y in self.snake:
            self.canvas.create_rectangle(x, y, x + SIZE, y + SIZE, fill="green", tag="snake")

    def draw_food(self):
```

```python
class SnakeGame:
    def draw_food(self):
        x, y = self.food
        self.canvas.create_oval(x, y, x + SIZE, y + SIZE, fill="red", tag="food")

    def place_food(self):
        x = random.randint(0, (WIDTH - SIZE) // SIZE) * SIZE
        y = random.randint(0, (HEIGHT - SIZE) // SIZE) * SIZE
        return (x, y)

    def move_snake(self):
        if not self.running:
            return

        head_x, head_y = self.snake[0]
        if self.direction == "Right":
            head_x += SIZE
        elif self.direction == "Left":
            head_x -= SIZE
        elif self.direction == "Up":
            head_y -= SIZE
        elif self.direction == "Down":
            head_y += SIZE

        new_head = (head_x, head_y)

        # Check for collisions
        if (
            head_x < 0 or head_x >= WIDTH or
            head_y < 0 or head_y >= HEIGHT or
            new_head in self.snake
        ):
            self.canvas.create_text(WIDTH/2, HEIGHT/2, text="Game Over", fill="white", font=("Arial", 24))
            self.running = False
```

```
43      def move_snake(self):
59          # Check for collisions
60          if (
61              head_x < 0 or head_x >= WIDTH or
62              head_y < 0 or head_y >= HEIGHT or
63              new_head in self.snake
64          ):
65              self.canvas.create_text(WIDTH/2, HEIGHT/2, text="Game Over", fill="white", font=("Arial", 24))
66              self.running = False
67              return
68
69          self.snake.insert(0, new_head)
70
71          if new_head == self.food:
72              self.food = self.place_food()
73              self.canvas.delete("food")
74              self.draw_food()
75          else:
76              self.snake.pop()
77
78          self.draw_snake()
79          self.master.after(SPEED, self.move_snake)
80
81      def change_direction(self, event):
82          key = event.keysym
83          opposites = {"Up": "Down", "Down": "Up", "Left": "Right", "Right": "Left"}
84          if key in opposites and opposites[key] != self.direction:
85              self.direction = key
86
87  root = tk.Tk()
88  root.title("Snake Game - Tkinter")
89  game = SnakeGame(root)
90  root.mainloop()
```

Output:-