PYTHON CODE FOR STAR TYPE MODEL

```
import math


class Bullet:
    def __init__(self, weight, velocity, distance, angle):
        self.weight = weight
        self.velocity = velocity
        self.distance = distance
        self.angle = angle

        # Check if the angle is valid. An angle of 90 degrees is not valid because the bullet would not travel very far.
        if abs(self.angle) > math.pi / 2:
            raise ValueError("Angle must be between -pi/2 and pi/2 radians")

    def calculate_trajectory(self):
        # Calculate the bullet's initial velocity components
        v_x = self.velocity * math.cos(self.angle)
        v_y = self.velocity * math.sin(self.angle)

        # Calculate the bullet's time of flight
        t_flight = self.distance / abs(v_x)

        # Calculate the bullet's trajectory
        x = [v_x * t for t in range(int(t_flight) + 1)]

        # Handle negative angles
```

```
    if self.angle < 0:
        x = [-xi for xi in x]

    y = [v_y * t - 0.5 * 9.8 * t**2 for t in range(int(t_flight) + 1)]

    return x, y

def calculate_impact_point(self):
    # Calculate the bullet's impact time
    t_impact = self.distance / abs(self.velocity)

    # Calculate the bullet's impact point
    x_impact = self.distance * math.cos(self.angle)

    # Handle negative angles
    if self.angle < 0:
        x_impact = -x_impact

    y_impact = self.velocity * t_impact - 0.5 * 9.8 * t_impact**2

    return x_impact, y_impact

def calculate_impact_velocity(self):
    # Check if the angle is close to 90 degrees (pi/2 radians)
    if math.isclose(self.angle, math.pi / 2, rel_tol=1e-9):
        return 0  # Return 0 for angles close to 90 degrees
```

```
        # Calculate the bullet's horizontal and vertical components of velocity at impact
        v_x_impact = self.velocity * math.cos(self.angle)

        # Handle negative angles
        if self.angle < 0:
            v_x_impact = -v_x_impact

        t_impact = self.distance / abs(v_x_impact)

        v_y_impact = self.velocity * math.sin(self.angle) - 9.8 * t_impact

        # Calculate the bullet's impact velocity
        v_impact = math.sqrt(v_x_impact**2 + v_y_impact**2)

        return v_impact

def main():
    # Create a bullet object
    bullet = Bullet(0.009, 380, 100, -math.pi / 2)

    # Calculate the bullet's trajectory
    x, y = bullet.calculate_trajectory()

    # Calculate the bullet's impact point
    x_impact, y_impact = bullet.calculate_impact_point()

    # Calculate the bullet's impact velocity
```

```
    v_impact = bullet.calculate_impact_velocity()

    # Print the results
    print("Bullet trajectory:")
    print(x)
    print(y)
    print("Bullet impact point:")
    print(x_impact)
    print(y_impact)
    print("Bullet impact velocity:")
    print(v_impact)

if __name__ == "__main__":
    main()
```