**Name:** Aditya Nittala **PRN:** 21070521004

**Department:** CSE **Division:** A

**Symbiosis Institute of Technology Department of Applied Science**



Generative AI CA-2

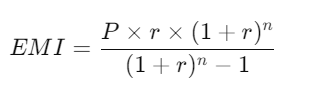
**Computer Science and Engineering Batch 2021-2025**

**Semester - VII**

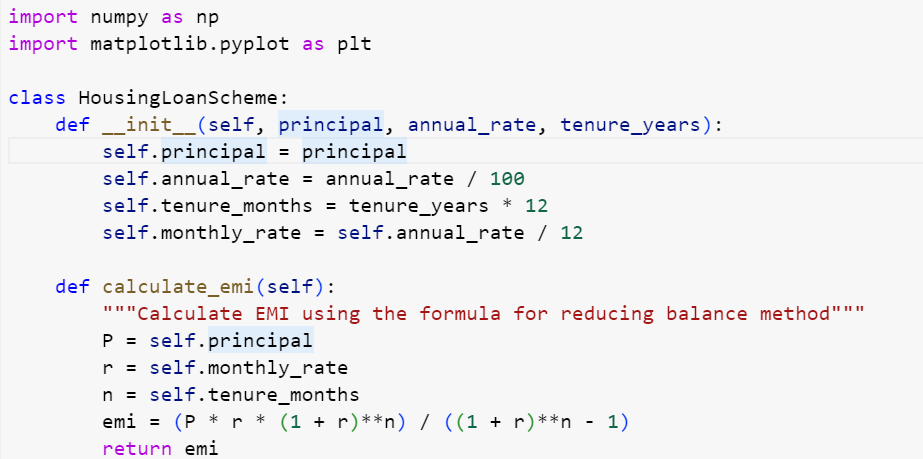
**Q2. Generate a model in Python to represent a Housing loan scheme and create a chart to display the Emi based on rate of interest and reducing balance for a given period. If a customer wishes to close the loan earlier, print the interest lost distributed over the remaining no. of months. Assume suitable data and inputs as necessary.**

The Python model simulates the housing loan process. The model consists of several key components:

1. Principal (Loan Amount): The total amount borrowed by the borrower.
2. Annual Interest Rate: The interest rate at which the loan is offered.
3. Tenure: The period the loan is repaid, typically measured in years.
4. EMI Calculation: The monthly instalment that includes both the repayment of the loan principal and the interest. The EMI is computed using the formula for reducing balance loans:

**` **

# Code:



**\_\_init\_\_(self, principal, annual\_rate, tenure\_years)**

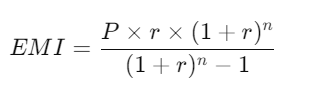
This is the constructor of the HousingLoanScheme class. It initialises the loan parameters:

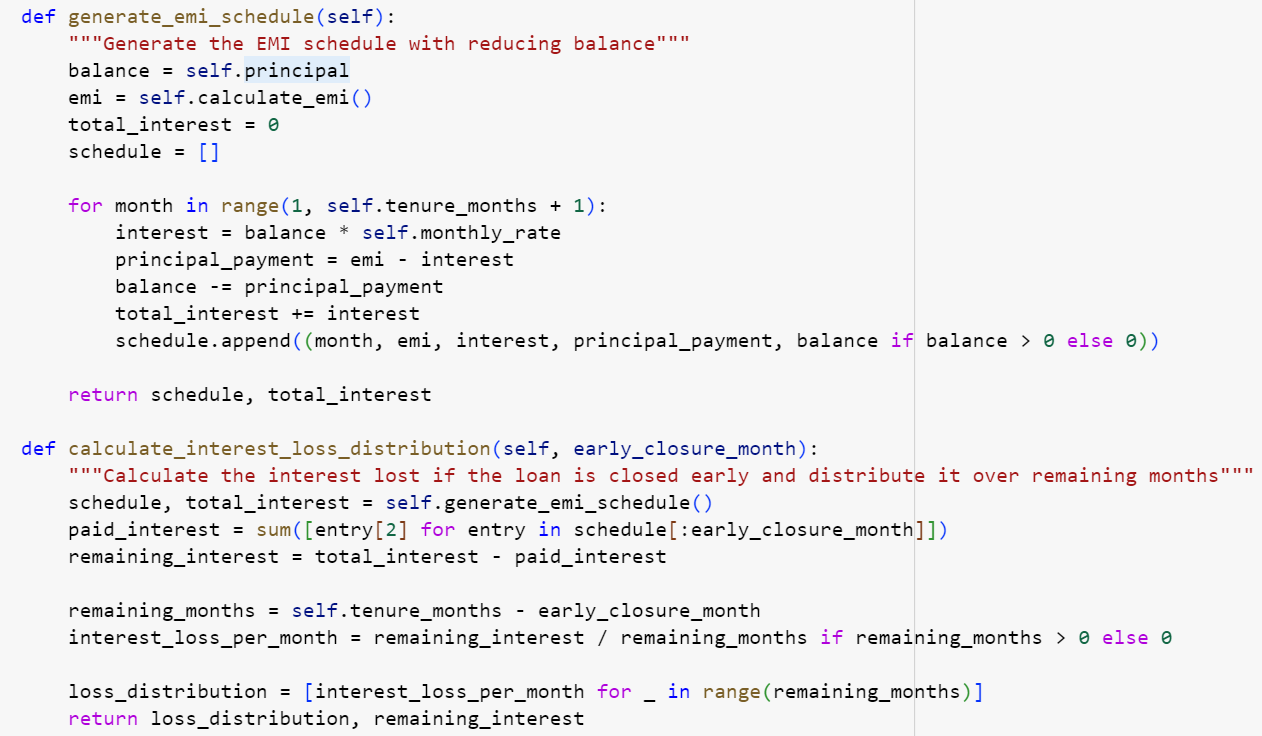
* principal: The loan amount (in currency units).
* annual\_rate: The annual interest rate as a percentage (converted to a decimal value).
* tenure\_months: The total loan tenure in months (years \* 12).
* monthly\_rate: The monthly interest rate is calculated by dividing the annual rate by 12.

**calculate\_emi(self)**

This method calculates the Equated Monthly Installment (EMI) using the reducing balance method.

* P is the loan principal (initial amount).
* r is the monthly interest rate.
* n is the total number of months for repayment.

****The EMI formula is:



**generate\_emi\_schedule(self)**

This method generates a schedule of EMI payments over the loan tenure, splitting each EMI into the portion paid toward **interest** and the portion paid toward the **principal**.

* It uses a monthly loop in the loan tenure (tenure\_months).
* In each iteration, the following is calculated:
  + interest: The interest for that month (balance \* montly\_rate).
  + principal\_payment: The part of the EMI that reduces the principal.
  + balance: The remaining loan balance after the principal is paid.

The function returns:

* A **schedule** list containing tuples for each month (month number, EMI, interest, principal payment, remaining balance).
* The **total interest** paid over the life of the loan.

**calculate\_interest\_loss\_distribution(self, early\_closure\_month)**

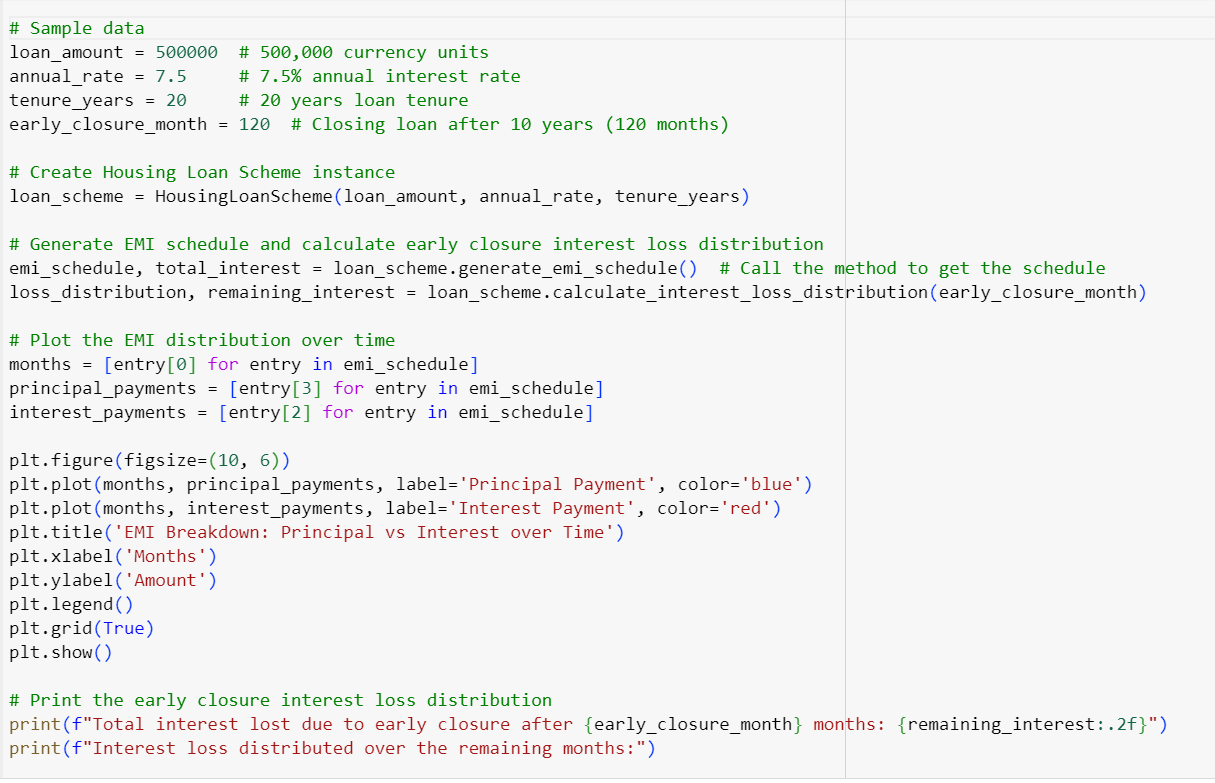
This method calculates the **interest lost** due to early loan closure and distributes it over the remaining loan tenure.

Steps:

* It first generates the EMI schedule using generate\_emi\_schedule().
* Then, it sums the interest paid up until the specified **early closure month**.
* The remaining interest (that would have been paid if the loan were continued) is calculated as: remaining\_interest=total\_interest−paid\_interest\text{remaining\\_interest} = \text{total\\_interest} - \text{paid\\_interest}remaining\_interest=total\_interest−paid\_interest
* The remaining interest is distributed evenly over the months that would have been left if the loan were not closed early.

This method returns:

* The **loss distribution**: a list of how much interest is lost for each month.
* The **remaining interest** that was avoided by closing the loan early.



**Plotting the EMI Breakdown**

The code generates a plot showing how the EMI is divided into **interest** and **principal payments** over time:

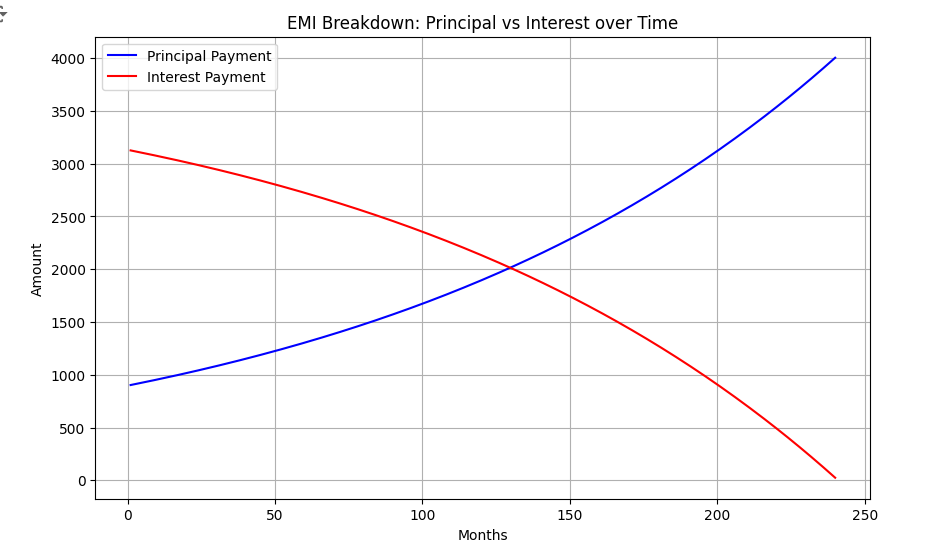
* The x-axis represents the number of months.
* The y-axis represents the payment amount.
* Two lines are plotted:
  + The **blue** line for the principal payments.
  + The **red** line for the interest payments.

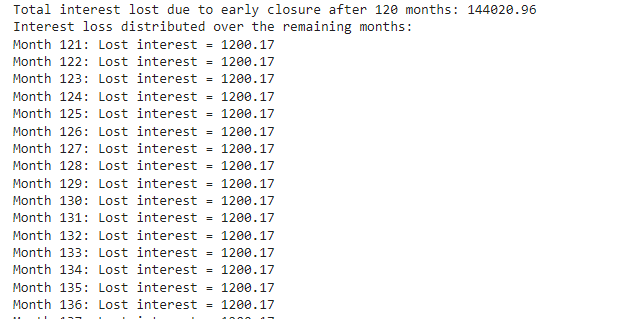
This shows how, over time, the interest portion decreases while the principal payment increases (because of the reducing balance method).

**Print the Early Closure Interest Loss Distribution**

The code prints how much interest is lost each month after the loan is closed early. It shows the loss of interest every month after the early closure point, making it clear how much interest is saved by closing the loan early.

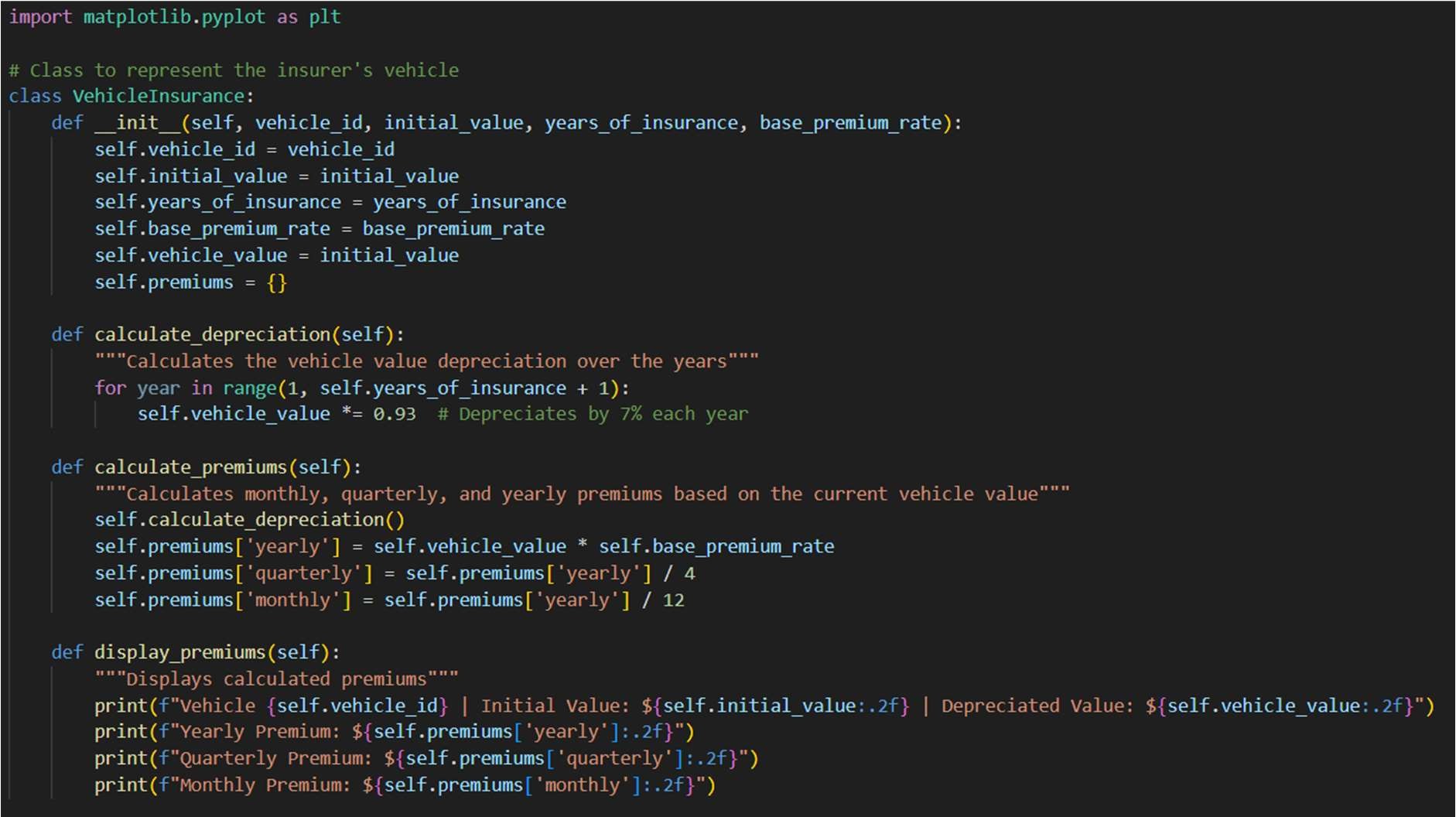
# Output (Generated Data):





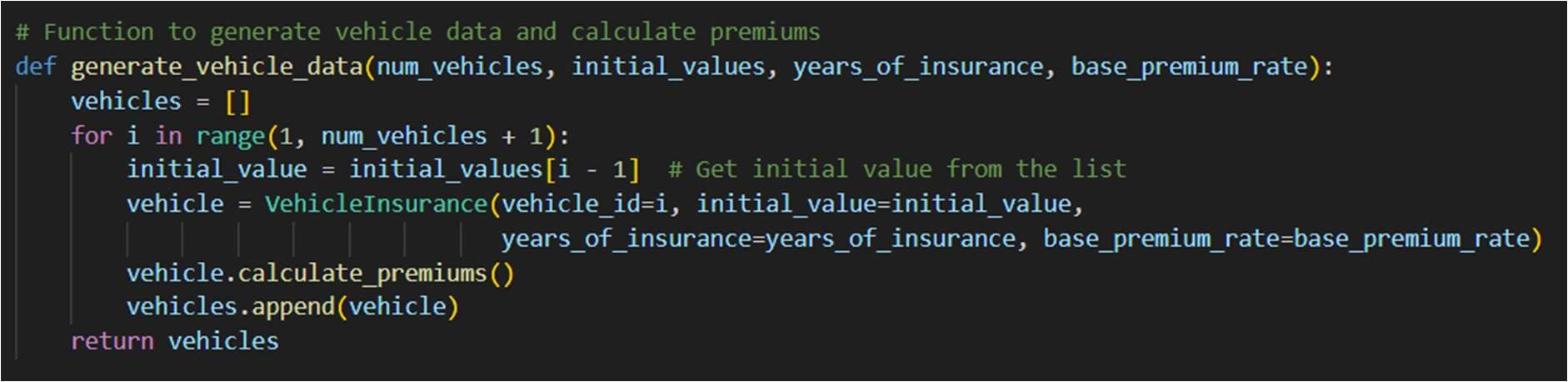
* 1. **Generate a model for an Insurance company to hold information on the insurer's vehicle, and create a chart of monthly, yearly, and quarterly premiums based on no. of years of insurance where in each year, the value of the vehicle depreciates by 7%.**

# Code:



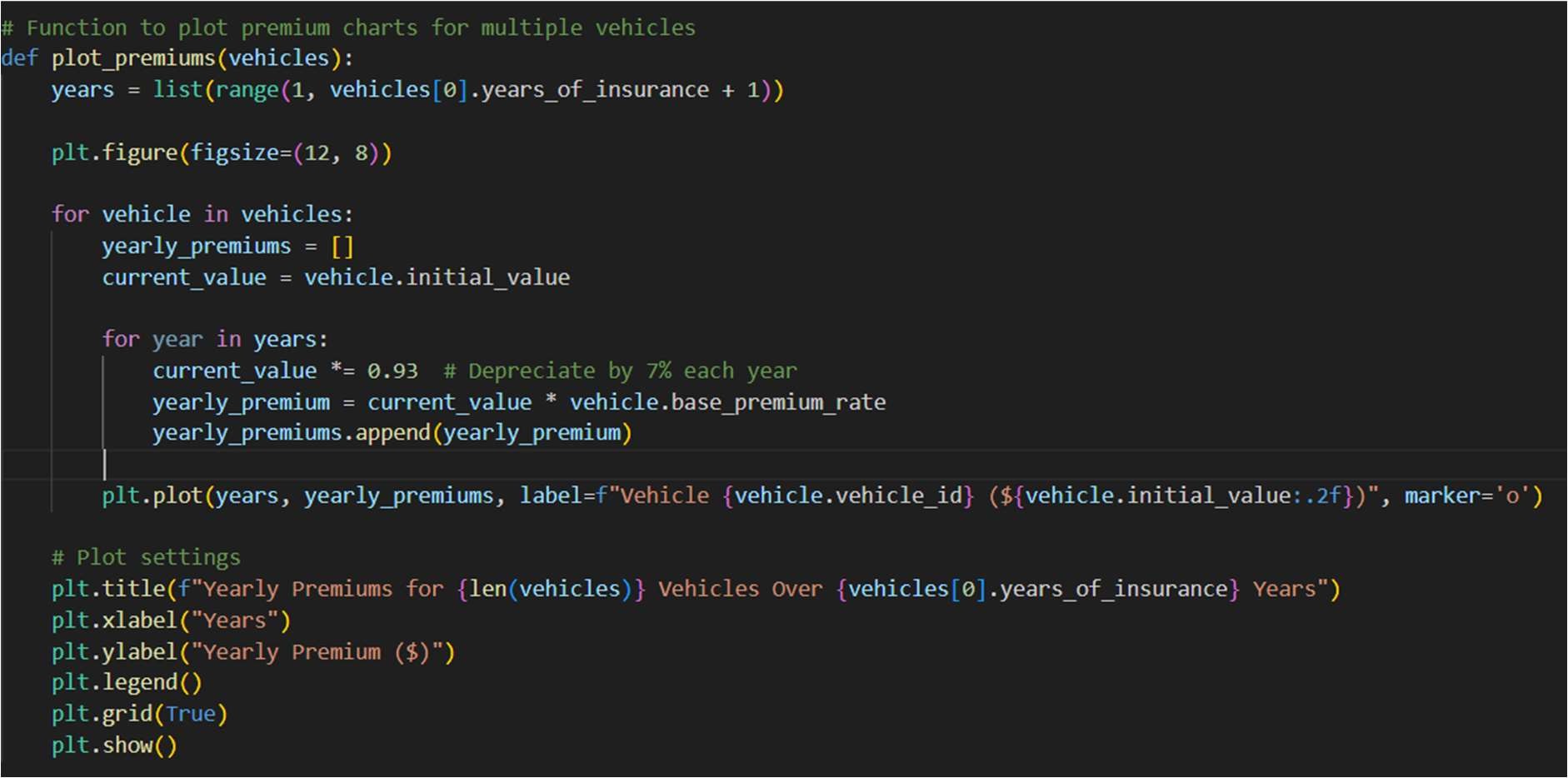
**Class** VehicleInsurance**:**

* + - This class models a vehicle with attributes for the vehicle ID, initial value, number of years of insurance, base premium rate, and a dictionary for storing premium values (monthly, quarterly, yearly).
    - It has methods to calculate the depreciation (7% per year), calculate the premiums based on the depreciated value, display the premiums for the vehicle.



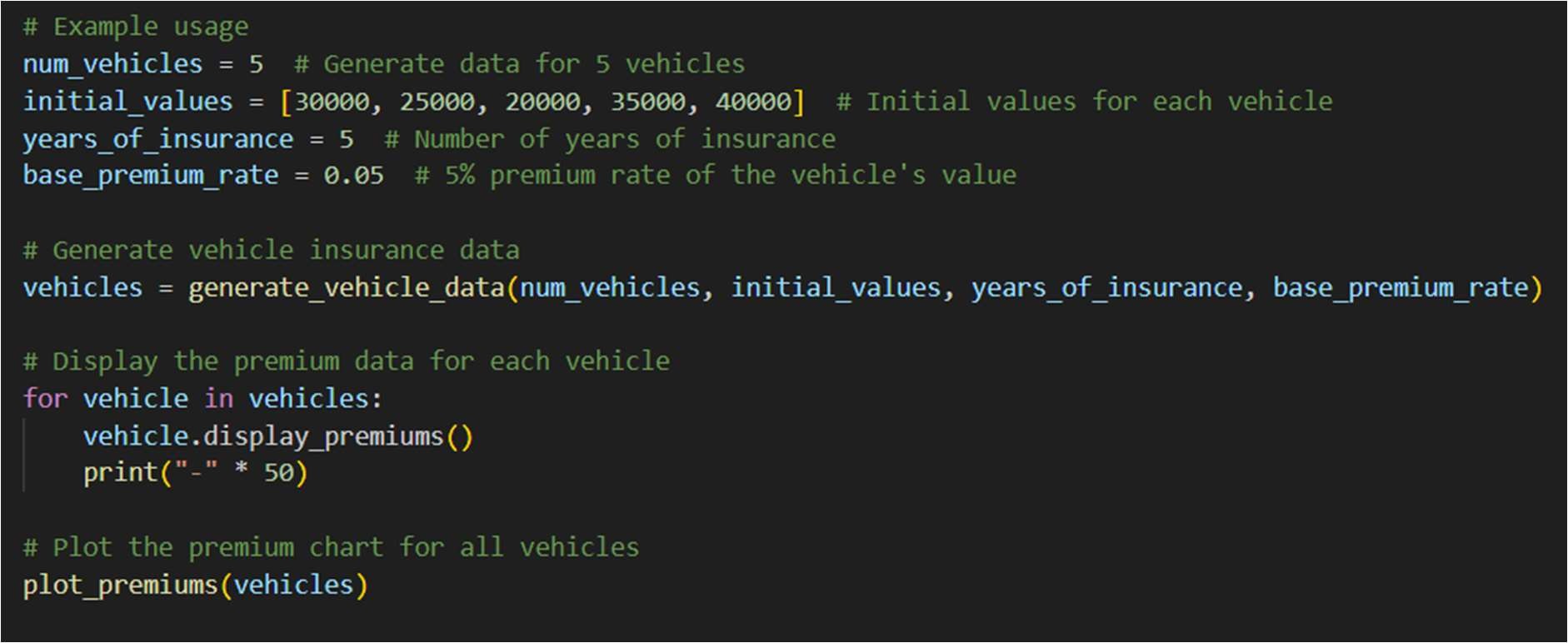
**Function to generate the vehicle data** generate\_vehicle\_data**:**

* + - Generates data for multiple vehicles, but for simplicity, it's generating one vehicle here.



**Function to plot the graphs** plot\_premiums**:**

* + - Plots a line chart showing how the premiums (monthly, quarterly, and yearly) change over the years as the vehicle value depreciates.



**Depreciation and Premium Calculation:**

* + - The vehicle's value depreciates by 7% per year.
    - Premiums are calculated as a percentage of the vehicle's current value after depreciation (5% in this example).
    - Monthly and quarterly premiums are derived from the yearly premium.

# Output (Generated Data):

The console will display the initial vehicle value, the depreciated value after 5 years, and the calculated premiums (monthly, quarterly, yearly).

**Vehicle 1 | Initial Value: $30000.00 | Depreciated Value: $20870.65 Yearly Premium: $1043.53**

**Quarterly Premium: $260.88 Monthly Premium: $86.96**

**--------------------------------------------------**

**Vehicle 2 | Initial Value: $25000.00 | Depreciated Value: $17392.21 Yearly Premium: $869.61**

**Quarterly Premium: $217.40 Monthly Premium: $72.47**

**--------------------------------------------------**

**Vehicle 3 | Initial Value: $20000.00 | Depreciated Value: $13913.77 Yearly Premium: $695.69**

**Quarterly Premium: $173.92 Monthly Premium: $57.97**

**--------------------------------------------------**

**Vehicle 4 | Initial Value: $35000.00 | Depreciated Value: $24349.09 Yearly Premium: $1217.45**

**Quarterly Premium: $304.36 Monthly Premium: $101.45**

**--------------------------------------------------**

**Vehicle 5 | Initial Value: $40000.00 | Depreciated Value: $27827.53**

**Yearly Premium: $1391.35**

**Quarterly Premium: $347.84 Monthly Premium: $115.95**

**Output Graph:**

The program will also display a chart showing how each vehicle's yearly premiums decrease over time as the vehicle value depreciates. Each vehicle is represented by a unique line on the chart, making it easy to compare the premium trends across different vehicles.

