JAVA WEEKEND ASSIGNMENT

31, May – 01, June

-Jahnavi Lakshmi Muttireddy

Car Customization and Option Display

Create a program that allows users to customize a car build and prints all the selected options. The user will provide input via standard input, and the output will be displayed on standard output.

Car Manufacturer:

- Mahindra
- Tata
- Maruti

Model (for Mahindra):

- Scorpio
- Thar
- Scorpio N
- XUV 700

Transmission Variant:

- Manual
- Automatic

Fuel Type:

- Diesel
- Petrol
- CNG

Accessories:

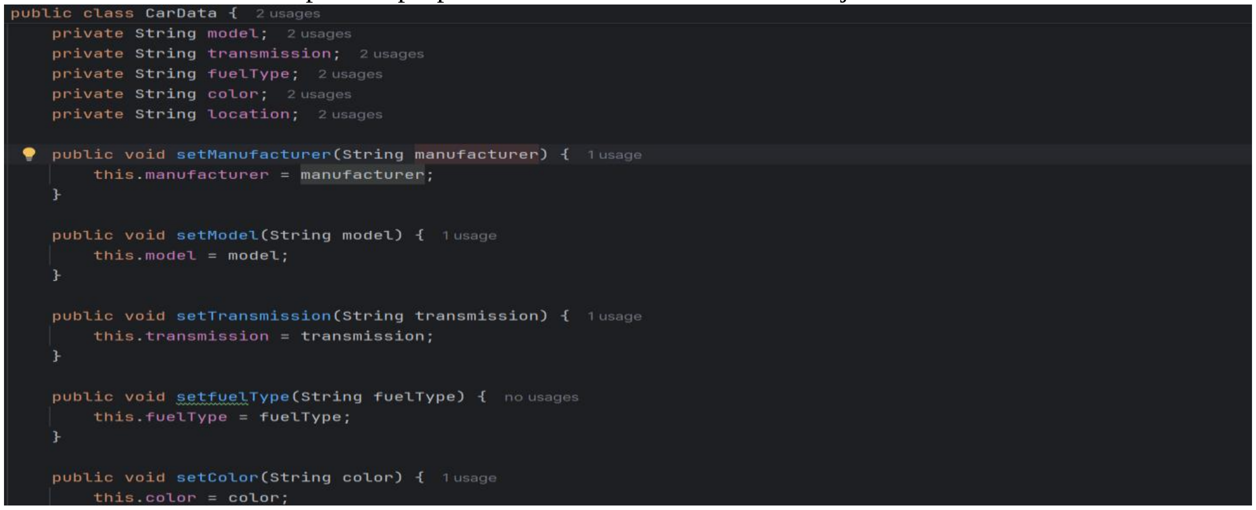
Color

- Silver
- Blue
- Yellow

Location:

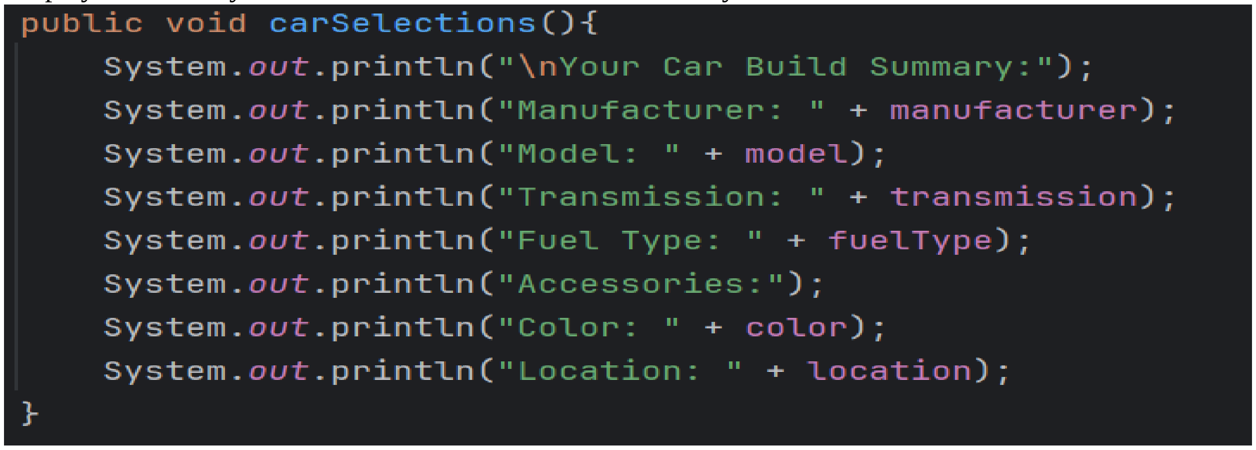
- Delhi
- Bangalore
- Hyderabad
- Chennai

Solution:
CarData Class:
These methods set the respective properties of the car customization object.



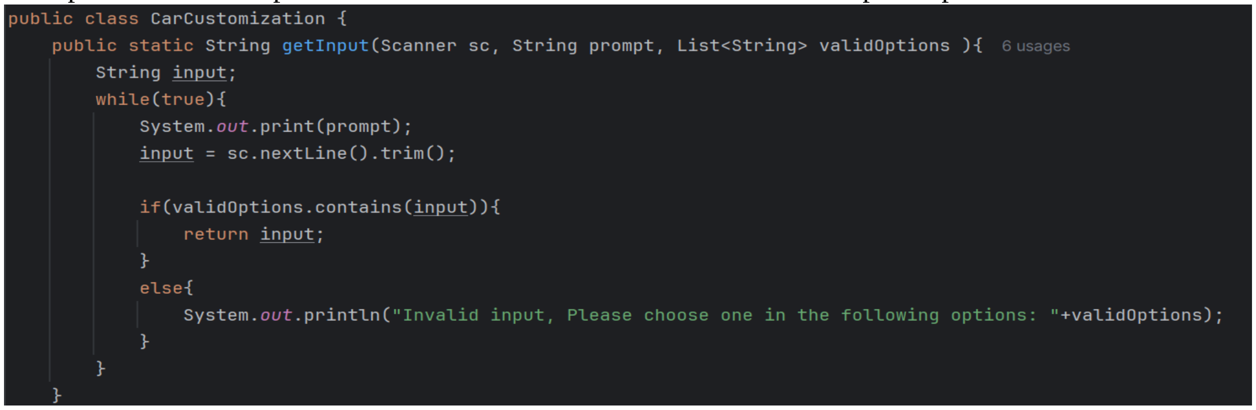
```
public class CarData {  2 usages
    private String model;  2 usages
    private String transmission;  2 usages
    private String fuelType;  2 usages
    private String color;  2 usages
    private String location;  2 usages

    public void setManufacturer(String manufacturer) {  1 usage
        this.manufacturer = manufacturer;
    }

    public void setModel(String model) {  1 usage
        this.model = model;
    }

    public void setTransmission(String transmission) {  1 usage
        this.transmission = transmission;
    }

    public void setfuelType(String fuelType) {  no usages
        this.fuelType = fuelType;
    }

    public void setColor(String color) {  1 usage
        this.color = color;
```

carSelections() Method:
Displays a summary of the car customization choices made by the user.



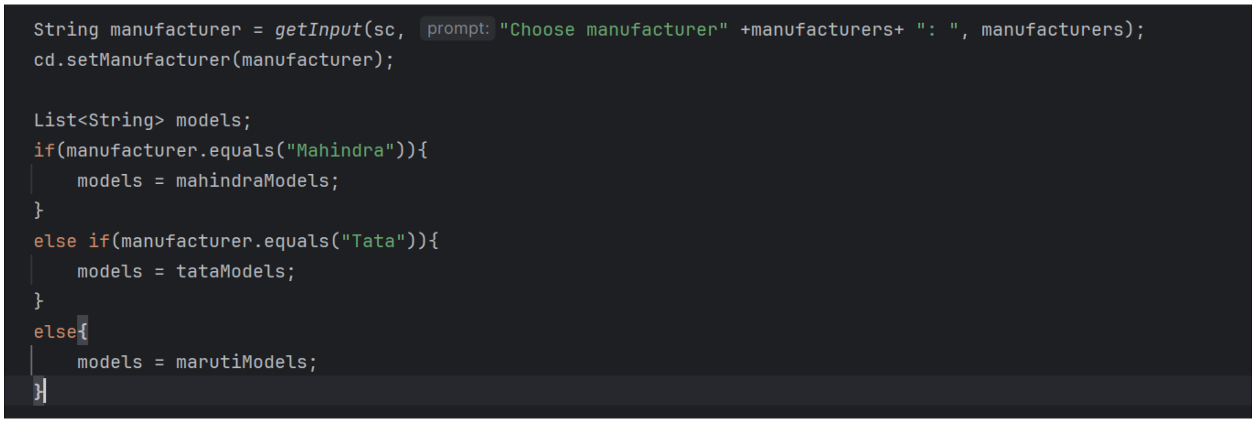
```
public void carSelections(){
    System.out.println("\nYour Car Build Summary:");
    System.out.println("Manufacturer: " + manufacturer);
    System.out.println("Model: " + model);
    System.out.println("Transmission: " + transmission);
    System.out.println("Fuel Type: " + fuelType);
    System.out.println("Accessories:");
    System.out.println("Color: " + color);
    System.out.println("Location: " + location);
}
```

CarCustomization Class:
getInput():
Prompts the user for input and validates that it matches one of the valid options provided.

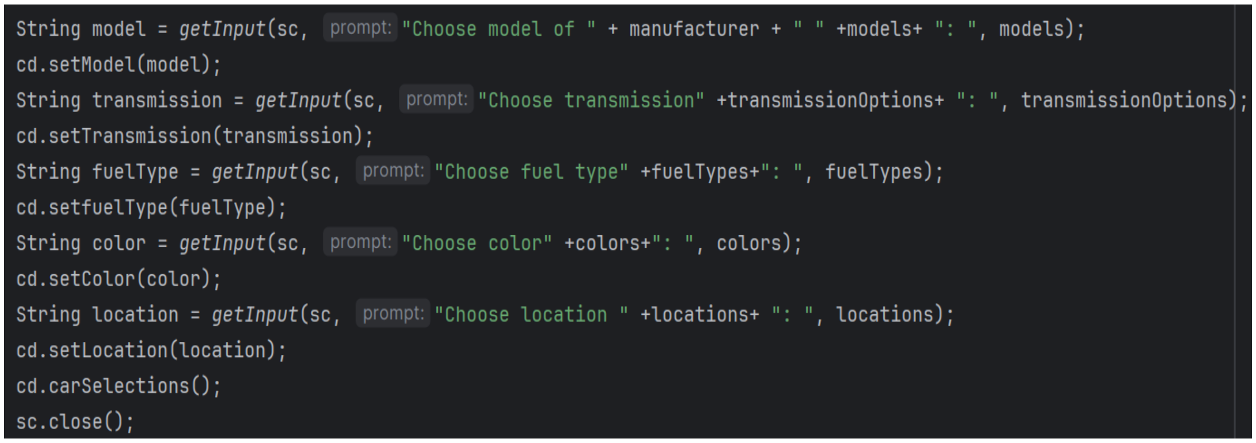


```
public class CarCustomization {
    public static String getInput(Scanner sc, String prompt, List<String> validOptions ){  6 usages
        String input;
        while(true){
            System.out.print(prompt);
            input = sc.nextLine().trim();

            if(validOptions.contains(input)){
                return input;
            }
            else{
                System.out.println("Invalid input, Please choose one in the following options: "+validOptions);
            }
        }
    }
```

Input Collection and Validation:
Prompts the user to select a manufacturer, model, transmission type, fuel type, color, and location, ensuring the inputs are valid and displays a summary of the car

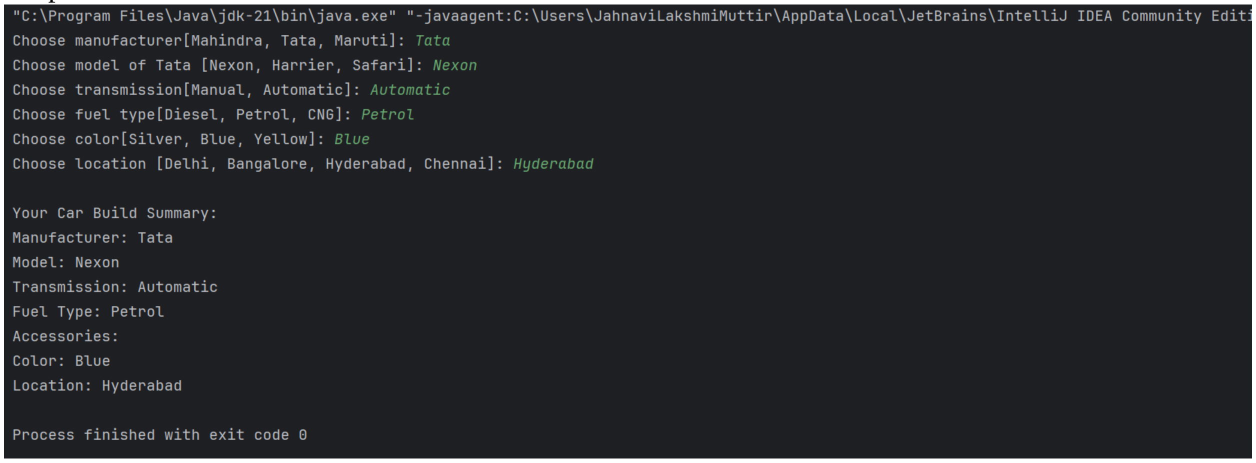


```
String manufacturer = getInput(sc, prompt: "Choose manufacturer" +manufacturers+ ": ", manufacturers);
cd.setManufacturer(manufacturer);

List<String> models;
if(manufacturer.equals("Mahindra")){
    models = mahindraModels;
}
else if(manufacturer.equals("Tata")){
    models = tataModels;
}
else{
    models = marutiModels;
}
```



```
String model = getInput(sc, prompt: "Choose model of " + manufacturer + " " +models+ ": ", models);
cd.setModel(model);
String transmission = getInput(sc, prompt: "Choose transmission" +transmissionOptions+ ": ", transmissionOptions);
cd.setTransmission(transmission);
String fuelType = getInput(sc, prompt: "Choose fuel type" +fuelTypes+": ", fuelTypes);
cd.setfuelType(fuelType);
String color = getInput(sc, prompt: "Choose color" +colors+": ", colors);
cd.setColor(color);
String location = getInput(sc, prompt: "Choose location " +locations+ ": ", locations);
cd.setLocation(location);
cd.carSelections();
sc.close();
```

Output:



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Users\JahnaviLakshmiMuttir\AppData\Local\JetBrains\IntelliJ IDEA Community Editi
Choose manufacturer[Mahindra, Tata, Maruti]: Tata
Choose model of Tata [Nexon, Harrier, Safari]: Nexon
Choose transmission[Manual, Automatic]: Automatic
Choose fuel type[Diesel, Petrol, CNG]: Petrol
Choose color[Silver, Blue, Yellow]: Blue
Choose location [Delhi, Bangalore, Hyderabad, Chennai]: Hyderabad

Your Car Build Summary:
Manufacturer: Tata
Model: Nexon
Transmission: Automatic
Fuel Type: Petrol
Accessories:
Color: Blue
Location: Hyderabad

Process finished with exit code 0
```

Problem Statement: Tax Calculation

Write a program to calculate the annual tax owed by an individual based on their salary, age, and other parameters. The user will input their details, and the program will output the total tax amount.

Parameters:

1. **Salary (in INR):**
    i. Annual salary of the individual.
2. **Age (in years):**
    i. Age of the individual.
3. **Investment in Tax-saving Instruments (in INR):**
    i. Amount invested in tax-saving instruments like PPF, ELSS, etc.
4. **Health Insurance Premium (in INR):**
    i. Annual health insurance premium paid by the individual.
5. **Home Loan Interest (in INR):**
    i. Annual interest paid on a home loan.

Tax Slabs:

1. **For individuals below 60 years:**

a. Up to ₹2,50,000: No tax
b. ₹2,50,001 to ₹5,00,000: 5%
c. ₹5,00,001 to ₹10,00,000: 20%
d. Above ₹10,00,000: 30%

2. **For individuals between 60 and 80 years:**

a. Up to ₹3,00,000: No tax
b. ₹3,00,001 to ₹5,00,000: 5%
c. ₹5,00,001 to ₹10,00,000: 20%
d. Above ₹10,00,000: 30%

3. **For individuals above 80 years:**

a. Up to ₹5,00,000: No tax
b. ₹5,00,001 to ₹10,00,000: 20%
c. Above ₹10,00,000: 30%

Deductions:

1. **Section 80C:**
    i. Maximum deduction of ₹1,50,000 for investments in tax-saving instruments.
2. **Section 80D:**
    i. Maximum deduction of ₹25,000 for health insurance premium (₹50,000 for senior citizens).
3. **Section 24:**
    i. Maximum deduction of ₹2,00,000 for home loan interest.

Output:

The program should output the total tax amount owed by the individual after considering the applicable deductions.

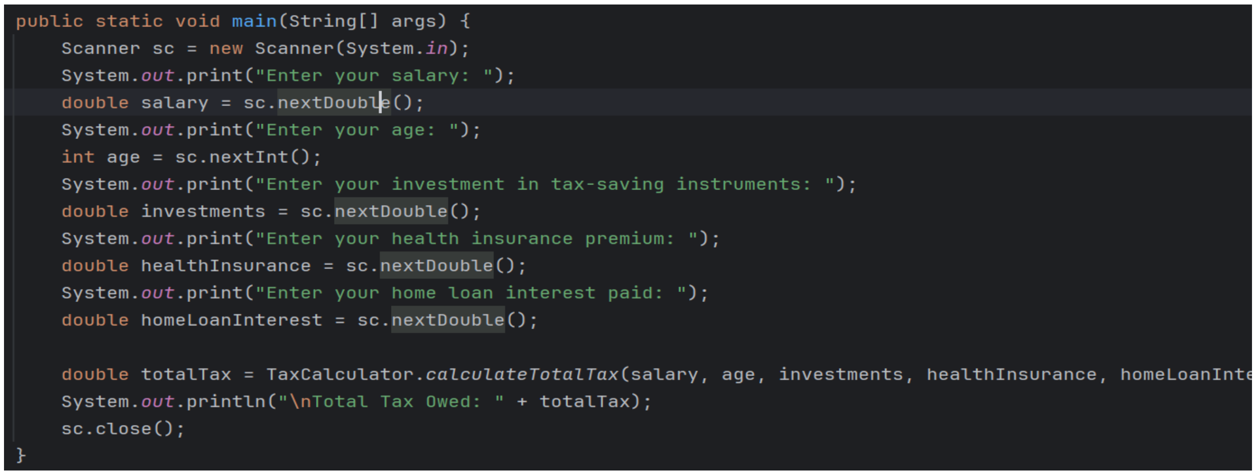
Solution:

program is divided into two classes:

- **TaxCalculator** handles the tax calculation based on income, age, and deductions,.

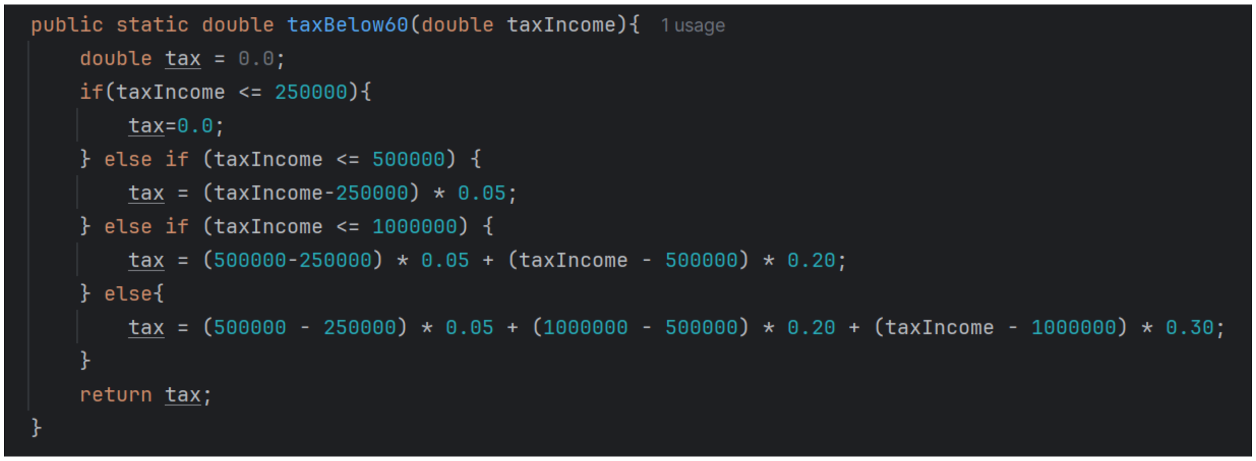
- **TaxMain** manages user input and displays the final tax owed

Main Method for User Interaction:



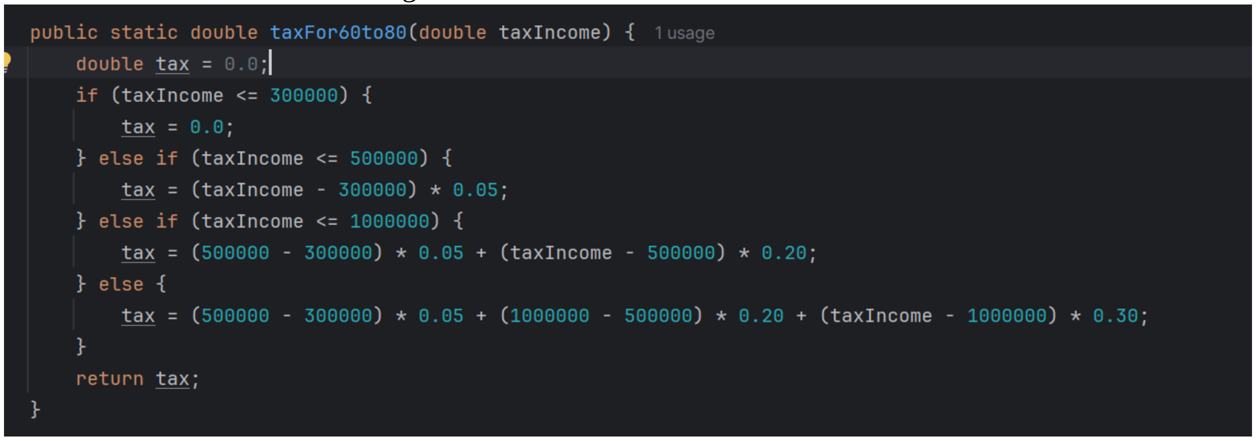
```java
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter your salary: ");
    double salary = sc.nextDouble();
    System.out.print("Enter your age: ");
    int age = sc.nextInt();
    System.out.print("Enter your investment in tax-saving instruments: ");
    double investments = sc.nextDouble();
    System.out.print("Enter your health insurance premium: ");
    double healthInsurance = sc.nextDouble();
    System.out.print("Enter your home loan interest paid: ");
    double homeLoanInterest = sc.nextDouble();

    double totalTax = TaxCalculator.calculateTotalTax(salary, age, investments, healthInsurance, homeLoanInte
    System.out.println("\nTotal Tax Owed: " + totalTax);
    sc.close();
}
```

Tax Calculation for Individuals Below 60 Years:



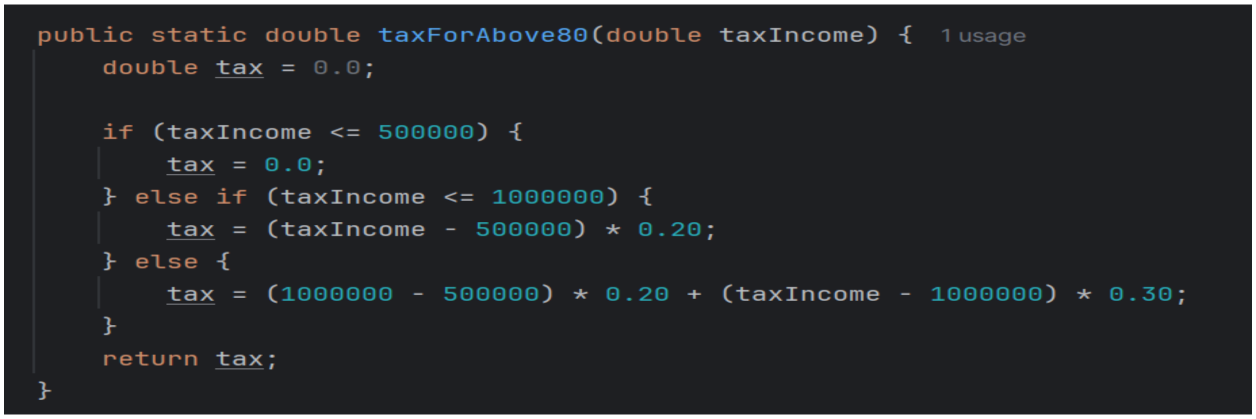
```java
public static double taxBelow60(double taxIncome){
    double tax = 0.0;
    if(taxIncome <= 250000){
        tax=0.0;
    } else if (taxIncome <= 500000) {
        tax = (taxIncome-250000) * 0.05;
    } else if (taxIncome <= 1000000) {
        tax = (500000-250000) * 0.05 + (taxIncome - 500000) * 0.20;
    } else{
        tax = (500000 - 250000) * 0.05 + (1000000 - 500000) * 0.20 + (taxIncome - 1000000) * 0.30;
    }
    return tax;
}
```

Tax Calculation for Individuals Aged 60-80 Years:



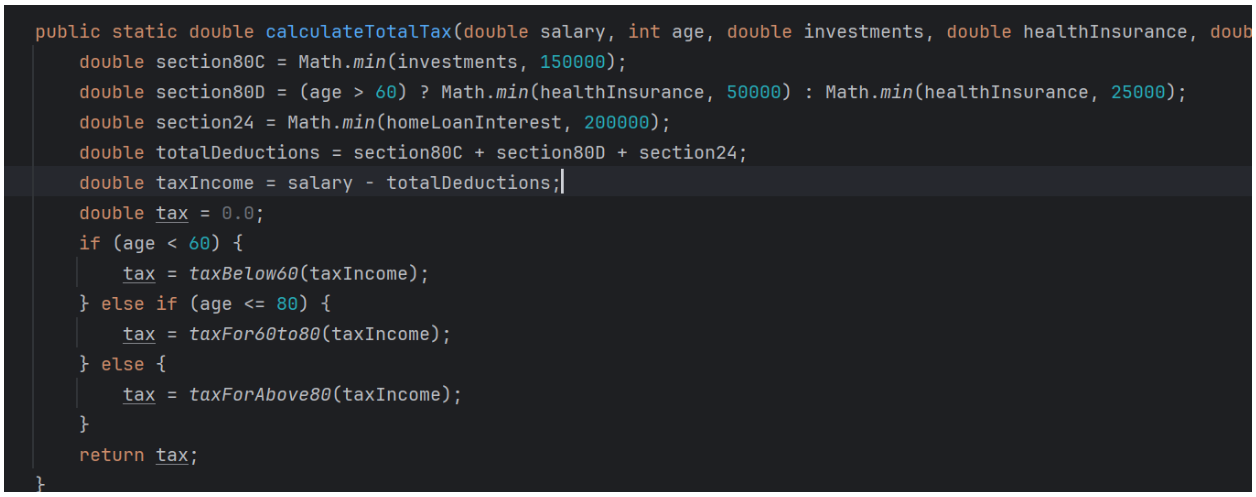
```java
public static double taxFor60to80(double taxIncome) {
    double tax = 0.0;
    if (taxIncome <= 300000) {
        tax = 0.0;
    } else if (taxIncome <= 500000) {
        tax = (taxIncome - 300000) * 0.05;
    } else if (taxIncome <= 1000000) {
        tax = (500000 - 300000) * 0.05 + (taxIncome - 500000) * 0.20;
    } else {
        tax = (500000 - 300000) * 0.05 + (1000000 - 500000) * 0.20 + (taxIncome - 1000000) * 0.30;
    }
    return tax;
}
```

Tax Calculation for Individuals Above 80 Years:



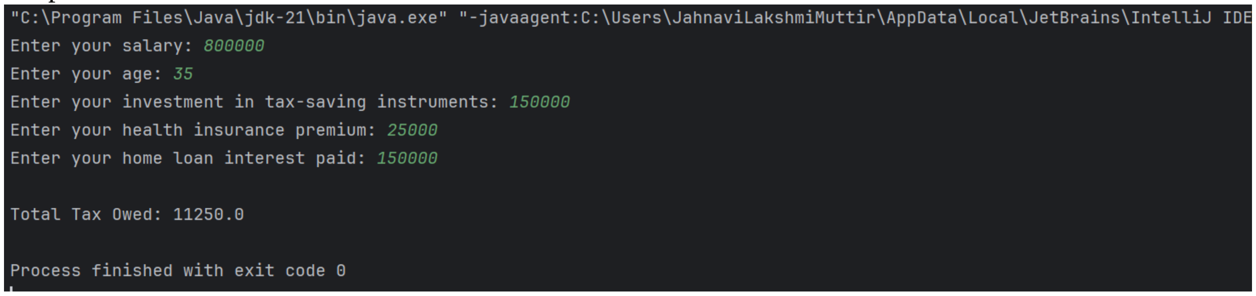
```java
public static double taxForAbove80(double taxIncome) {  1 usage
    double tax = 0.0;

    if (taxIncome <= 500000) {
        tax = 0.0;
    } else if (taxIncome <= 1000000) {
        tax = (taxIncome - 500000) * 0.20;
    } else {
        tax = (1000000 - 500000) * 0.20 + (taxIncome - 1000000) * 0.30;
    }
    return tax;
}
```

Main Tax Calculation Method applies deductions under Section 80C, Section 80D, and Section 24:



```java
public static double calculateTotalTax(double salary, int age, double investments, double healthInsurance, doub
    double section80C = Math.min(investments, 150000);
    double section80D = (age > 60) ? Math.min(healthInsurance, 50000) : Math.min(healthInsurance, 25000);
    double section24 = Math.min(homeLoanInterest, 200000);
    double totalDeductions = section80C + section80D + section24;
    double taxIncome = salary - totalDeductions;
    double tax = 0.0;
    if (age < 60) {
        tax = taxBelow60(taxIncome);
    } else if (age <= 80) {
        tax = taxFor60to80(taxIncome);
    } else {
        tax = taxForAbove80(taxIncome);
    }
    return tax;
}
```

Output:



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Users\JahnaviLakshmiMuttir\AppData\Local\JetBrains\IntelliJ IDE
Enter your salary: 800000
Enter your age: 35
Enter your investment in tax-saving instruments: 150000
Enter your health insurance premium: 25000
Enter your home loan interest paid: 150000

Total Tax Owed: 11250.0

Process finished with exit code 0
```