

Lab Assignment #4: Bank Loan Management System

Objective:

Create a Python program that simulates a bank loan system offering three types of loans:

- **Housing Loan:** 5.2% interest rate, term up to 25 years.
- **Auto Loan:** 7.5% interest rate, term up to 6 years.
- **Personal Loan:** 9.6% interest rate, term up to 10 years.

The system should calculate the monthly payment, total interest paid by the end of the term, and ensure that the total monthly installment does not exceed **50% of the user's monthly income** (debt ratio). The system should allow users to adjust the loan term to lower the monthly payment if needed to meet this constraint.

Assignment Requirements:

User Input:

- Allow the user to:
 - Select the type of loan.
 - Enter the loan amount.
 - Specify the term (within the allowable limits for the selected loan).
 - Enter their monthly income to check debt ratio compliance.
 - **Input Validation:**
 - Ensure loan term limits are enforced.
 - Prevent negative or unrealistic entries.
 - Guide the user through correcting invalid inputs.
-

Loan and Payment Calculations:

- Use a function to calculate the **monthly payment** using the formula:

$$M = \frac{P \times r \times (1 + r)^n}{(1 + r)^n - 1}$$

Where:

- M = Monthly payment
- P = Loan amount
- r = Monthly interest rate (annual_rate/12)
- n = Total number of payments (term×12)

- Check that the monthly payment does not exceed **50% of the user's monthly income**. If it does, allow the user to:
 - Increase the loan term (if possible) to reduce the monthly payment.
 - Select a different loan amount.

Formula explanation:

Formula:

$$M = \frac{P \times r \times (1 + r)^n}{(1 + r)^n - 1}$$

Explanation of Variables:

- M : Monthly payment amount that the borrower needs to pay.
- P : Loan amount or principal — the amount borrowed by the customer.
- r : Monthly interest rate, which is derived from the annual interest rate.

$$r = \frac{\text{annual interest rate}}{100 \times 12}$$

For example:

- Annual interest rate of 5.2% $\rightarrow r = \frac{5.2}{100 \times 12} = 0.00433$
- n : Total number of payments, calculated by:

$$n = \text{loan term in years} \times 12$$

For example:

- 25-year loan $\rightarrow n = 25 \times 12 = 300$ payments.

How It Works:

1. Convert Annual Rate to Monthly Rate:

Since interest is typically quoted annually, divide the rate by 12 to get the monthly rate.

2. Compound Growth Factor:

$(1 + r)^n$ accounts for the compounding of the loan over the period.

3. Amortization Factor:

The denominator $(1 + r)^n - 1$ adjusts for the fact that the loan amount is being repaid over time with interest.

Example Calculation:

Suppose a customer takes a \$100,000 housing loan at 5.2% interest for 20 years:

- $P = 100,000$
- Annual interest rate = 5.2%, so:

$$r = \frac{5.2}{100 \times 12} = 0.00433$$


- Term = 20 years $\rightarrow n = 20 \times 12 = 240$

Substitute the values:

$$M = \frac{100,000 \times 0.00433 \times (1 + 0.00433)^{240}}{(1 + 0.00433)^{240} - 1}$$

After calculation:

$$M \approx 672.89$$

-  The monthly payment would be approximately **\$672.89**.
-

Key Points:

- The formula ensures that both **principal** and **interest** are included in every payment.
 - In the beginning, a larger portion of the payment goes toward interest, and over time, more goes toward reducing the principal.
-

Loan Options and Summary:

- Display a summary of options showing:
 - Monthly payment amount.
 - Total interest paid by the end of the loan period.
 - Option to proceed or adjust inputs.
- Store the finalized loan record in a **CSV file** named `loan_records.csv` with the following format:

Loan Type, Loan Amount, Interest Rate, Term, Monthly Payment, Total Interest, Status

Error Handling:

- Prevent invalid input formats.
- Warn the user if they exceed the allowed debt ratio.
- Allow re-entry of valid values where necessary.

Program Features:

Loan Calculation Function:

```
def calculate_monthly_payment(principal, annual_rate, term_years):  
    r = annual_rate / 100 / 12  
    n = term_years * 12  
    if r == 0:  
        return principal / n  
    return principal * r * (1 + r) ** n / ((1 + r) ** n - 1)
```

User Interaction:

- Guide the user through selecting the loan type and entering loan details.
- Warn users if their monthly payment exceeds the 50% debt ratio.
- Suggest extending the term if applicable.

Group Tasks:

To ensure active participation:

- **Group Member 1:** Handle user input validation and error handling.
- **Group Member 2:** Implement loan calculation logic and enforce term/debt ratio constraints.
- **Group Member 3:** Develop the summary display and finalize the loan process.
- **Group Member 4 (if applicable):** Test for edge cases and document the program logic.

Deliverables:

1. **Submit the Python Code (.py file).**
2. **Include a short document explaining:**
 - The role of each group member.
 - Challenges faced and how they were resolved.

- Sample test cases and outcomes.

Grading Criteria:

- Correct implementation of loan logic (30%)
- Proper input validation and error handling (20%)
- Accurate debt ratio check and user guidance (20%)
- Proper file handling and record storage (20%)
- Documentation and group collaboration report (10%)