**Chapter\_2\_Exercises\_and\_Project\_Notes**

2.12 Suggested Exercises Try the following exercises to reinforce your understanding of the concepts covered in this chapter. Each exercise is designed to help you practice different aspects of Python programming. You can find the solutions to each of the exercises in Appendix A at the back of the book.

**Exercise 1: Basic Variable Assignment**

**Description:**

In this exercise, you will practice assigning values to variables of different data types and printing them. This will help you understand how to work with integers, floats, strings, and booleans in Python.

**Task:**

Assign values to variables of different data types (integer, float, string, boolean) and print them.

**Exercise 2: Type Conversion**

**Description:**

This exercise focuses on type conversion, which is converting one data type to another. You will write a script that takes a user input (string), converts it to an integer, performs a calculation, and then prints the result.

**Task:**

Write a script that takes a user input (string), converts it to an integer, performs a calculation, and then prints the result.

**Exercise 3: String Formatting**

**Description:**

Learn how to format strings using f-strings and the format() method. This exercise will help you create strings that include variable values and format them to display properly.

**Task:**

Create a string using both f-strings and the format() method that includes a variable integer and float, displaying them to two decimal places.

**Exercise 4: Control Structures**

**Description:**

Practice using control structures to make decisions in your code. In this exercise, you will write a script that asks the user for a number and checks if the number is positive, negative, or zero, printing an appropriate message for each case.

**Task:**

Write a script that asks the user for a number and checks if the number is positive, negative, or zero, printing an appropriate message for each case.

**Exercise 5: Arithmetic Operations**

**Description:**

Get hands-on with Python’s arithmetic operators by performing various calculations. This exercise will help you understand how to use operators like addition, subtraction, multiplication, division, and exponentiation.

**Task:**

Write a script that performs and prints the results of addition, subtraction, multiplication, division, and exponentiation of two numbers provided by the user.

**Exercise 6: Comparison and Logical Operators**

**Description:**

Explore Python’s comparison and logical operators by creating a script that categorizes a user’s age. This exercise will enhance your understanding of how to compare values and combine conditions.

**Task:**

Create a script that asks the user for their age and determines if they are a child, teenager, adult, or senior, using both comparison and logical operators.

**Exercise 7: Ternary Conditional Expression**

**Description:**

Learn to use ternary conditional expressions for concise decision-making in your code. This exercise involves writing a script that uses a ternary expression to determine whether a user’s input number is even or odd.

**Task:**

Write a script that determines whether a user’s input number is even or odd using a ternary conditional expression.

**2.13 Project: Loan Payment Calculator**

Are you ready to tackle your first project? I think so. Like all projects in this book, you can download the final solution from GitHub. For more details, refer to the Appendix.

**Objective:**

Enhance your understanding of Python by developing a Loan Payment Calculator that accounts for optional down payments and handles a 0% interest rate. This project will deepen your knowledge of variables, data types, and arithmetic operations and introduce you to making decisions in your code using conditional logic.

**Step-by-Step Guide:**

**Prompt for Basic Loan Information:** Ask the user to input the total loan amount for the car. Request the annual interest rate as a percentage. The user should be able to input a number like 7.5 for a 7.5% rate. Inquire about the loan duration in years from the user.

**Inquire About Down Payment:** Ask the user if they want to include a down payment. If the user answers yes, prompt for the down payment amount and adjust the loan amount accordingly by subtracting the down payment from the initial loan amount.

**Calculate Monthly Payment:** First, check if the annual interest rate is greater than 0. If yes, convert the annual interest rate to a monthly rate and proceed with the regular loan payment calculation formula. If the interest rate is 0%, calculate the monthly payment by simply dividing the loan amount (after adjusting for any down payment) by the total number of payments (loan duration in years multiplied by 12). Use the formula to calculate the monthly payment of a loan.

**The formula is:** where: M is the monthly payment. P is the loan amount (principal). r is the monthly interest rate (annual interest rate divided by 12). n is the total number of payments (loan duration in years multiplied by 12). The point of this project is not to know how to read math formulas, so here is the code to calculate the monthly payment: # Car loan monthly payment calculation

numerator = loan\_amount \* monthly\_interest\_rate

denominator = 1 - (1 + monthly\_interest\_rate) \*\* -total\_payments

monthly\_payment = numerator / denominator

**Output Detailed Loan Information:** Display the loan amount (after any down payment adjustment). Show the total number of payments and the duration of the loan in years. Present the interest rate to the user. Finally, display the calculated monthly payment, formatted to two decimal places.

**Tips for Success:** Clarity and Precision: Ensure your prompts are clear so the user knows exactly what information to enter. Precision in instructions leads to accurate inputs.

**Variable Naming:** Use descriptive names for your variables to make your code self-explanatory and easier to follow. Implement

**Conditional Logic:** Use if statements effectively to guide the program's flow based on the user's inputs about the down payment and interest rate.

**Test Thoroughly:** Run your program with different scenarios (including with and without a down payment, with a 0% and a positive interest rate) to ensure it behaves as expected under various conditions. By completing this project, you'll not only practice basic Python programming concepts but also learn how to incorporate conditional logic to make your programs more dynamic and adaptable to user input.

This practical application reinforces the fundamentals and Example of Possible Output

* Enter purchase amount: 15000
* Annual interest rate: 10
* Loan duration (years): 5
* Include down payment? (y/n): y
* Down payment amount: 2000

Loan Details:

* Purchase Amount: $15000.00
* Down Payment: $2000.00
* Loan Amount: $13000.00
* Number of Payments: 60 (5 years)
* Interest Rate: 10.000
* Monthly Payment: $276.21

**Steve’s Working Draft Workflow:**

* Document the Problem
* Develop or Understand a Logic Narrative
* Develop Psudo Code
* Code
* Test and Debug