CET1012 - Programming Methodologies: Java - Practicum 02

Topics Covered: Variables, Datatypes, Literals, Operators, Advanced Datatypes, Statements, Expressions, Blocks, and Flow Control

Learning Objectives:

- Familiarize with basic Java programming language. This includes
  - basic Java program structure
  - relationships between variables, datatypes, literals, and operators
  - advanced datatypes such as `Strings`
  - using statements, expressions, and blocks
  - applying flow control techniques with `if`, `switch`, and loop statements
- Apply coding best practices such as
  - proper and consistent naming conventions
  - proper and sufficient comments

Deliverables:

- Submit a single Java file called `CET1012_P02_<Your_Name>.java` (e.g. `CET1012_P02_John_Doe.java`).
- Note that a non-working submission will result in a zero

Background

A financial calculator is a specialized calculator that performs specific financial calculations that are usually not available in the typical calculator. Some of the functions include calculating cash flow amounts, solving interest rates, present and future values, depreciation, and tax.

Task

Your task is to implement a simple financial calculator program using Java.

Your calculator program has only 2 financial functions:

1. Calculating the final amount based on simple interest formula $A = P(1 + rt)$

   A = final amount

   P = initial principal balance

   r = annual interest rate

   t = time period (in years)

2. Calculating the final value based on compound interest formula: $A = P(1 + \frac{r}{n})^{nt}$

   A = final amount

   P = initial principal balance

   r = annual interest rate

   t = time period (in years)

   n = number of times interest applied per year

Program Requirements

- Your program will take in the following as input arguments
  - type of calculation (`simple` or `compound`)
  - formula parameters depending of type of calculation chosen. Note that the order of the arguments entered should follow according to the order shown in the formula in the 'Task' section.
- You may assume that the inputs entered will be of the correct type. This means that if the input is a numeral, the numeral can be of any classification of numerals.
- While the inputs can be assumed to be of the correct type, your program must still check for the correct number of input arguments (refer to example 3 below)
- However, if the number of arguments are correct, you may also assume the input arguments are of the correct type.
- Your program will display the final amount up to 2 decimal places as output

- Examples of running the program using the command-line are shown below.
  - Example 1 - Simple Interest

    To calculate the final amount based on simple interest formula given an initial principal balance of $18000, an annual interest rate of 6%, and a time period of 3 years, the user will enter the following:

    ```
    java CET1012_P02_John_Doe simple 18000 6 3
    ```

    The resulting output will be:

    ```
    Final amount is $21240.00
    ```

  - Example 2 - Compound Interest

    To calculate the final amount based on compound interest formula given an initial principal balance of $1500, an annual interest rate of 4.3% compounded quarterly for a time period of 6 years, the user will enter the following:

    ```
    java CET1012_P02_John_Doe compound 1500 4.3 6 4
    ```

    The resulting output will be:

    ```
    Final amount is $1938.84
    ```

  - Example 3 - Compound Interest with incorrect number of arguments

    An example is if the user enters only four input arguments instead of the five as required to compute the final value based on compound interest such as

    ```
    java CET1012_P02_John_Doe compound 1500 4.3 6
    ```

    then, an error message will be shown as output instead.

    ```
    You did not enter the correct number of inputs
    ```

- Use the `switch` statement to control the program flow depending on interest type.

  If there is no case match, it will display the following error message

  ```
  Command not found
  ```

- Use loops to compute exponentiation operations (Note for example, 2^3 can be rewritten as $2 \times 2 \times 2$)
- All code to be written within the `main` function
- As this is your first Java practicum, you are to also required to fulfil the following requirements
  - proper and consistent naming conventions
  - proper and sufficient comments to explain your code

For the maximum allocation of marks, refer to the table below.

Description	**Marks (%)**
Successful implementation of simple interest function.	30
Successful implementation of compound interest function including usage of loops for exponentiation operations.	30
Proper error handling for erroneous scenarios	20
Implement the `switch` statement to control the program flow depending on interest type.	10
Proper and consistent naming conventions.	5
Proper and sufficient comments to explain code.	5

Once you have completed, save your file in the following format `CET1012_P02_<Your_Name>.java` e.g. `CET1012_P02_John_Doe.java`.