CIS 106 – Loops Part 2

For each problem prepare an IPO chart. Then write the code for each. Save the IPO within this document and upload to your repository. After code is complete upload the files (.py) to your repository. Paste the link to your repository into the assignment completion link in Blackboard.

1. Allow the user to enter a principle amount and interest rate repeatedly (need a loop to control the program execution). Compute the annual interest (principle x rate). Compute ending balance to be principle (beginning balance + interest). Display year, beginning balance and ending balance for each of the 5 years. Display the accumulated interest for the 5 years. Note: the new balance by year (this will be the principle for the following year. Format the output.

Example:

Enter principle amount: 10000.00

Enter interest rate: 0.10

Year	Beginning Balance	Ending Balance
1	$10,000.00	$11,000.00
2	$11,000.00	$12,100.00
3	$12,100.00	$13,310.00
4	$13,310.00	$14,641.00
5	$14,641.00	$16,105.00

Total interest earned: $6,156.00

Input:

- Principle amount (float)
- Interest rate (float)

Process:

- Initialize the beginning balance with the principle amount.
- For each of 5 years:
    - Calculate the interest for the year (principle * rate).
    - Calculate the ending balance (beginning balance + interest).
    - Set the beginning balance for the next year to the ending balance of the current year.
    - Accumulate the total interest.
- Format the output values.

Output:

- Display year, beginning balance, and ending balance for each of the 5 years.
- Display the total interest earned over the 5 years.

2. Fibonacci sequence is a sequence of natural order. The sequence is:
   1, 1, 2, 3, 5, 8 etc
   Use of for loop compute and display first 20 numbers in the sequence. Hint: start with 1 , 1.

    - Input
- There is no external input for this problem. The computation will start with the initial values of the Fibonacci sequence, which are 1 and 1.
    - Process
- Initialize the first two numbers in the sequence: fib1 = 1 and fib2 = 1.
- Create a list to store the Fibonacci numbers and initialize it with the first two numbers.
- Use a for loop to compute the next 18 numbers in the sequence:
    - In each iteration, calculate the next number in the sequence by adding the last two numbers.
    - Update the variables to hold the last two numbers for the next iteration.
    - Append the newly calculated number to the list.
- Continue the loop until the list contains 20 numbers.
    - Output
- Display the first 20 numbers in the Fibonacci sequence.

3. Create a text file that contains employee last name and salary. Read in this data. Determine the bonus rate based on the chart below. Use that rate to compute bonus. For each line display the employee last name, salary and bonus. After the loop display the sum of all bonuses paid out.

Salary	Bonus Rate
100,000.00 and up	20%
50,000.00	15%
All other salaries	10%

Example file (create your own data with at least 5 lines:

Adams

50000.00

Baker

75000.00

Smith

45000.00

Etc

Input

- A text file containing employee last names and their respective salaries. Each employee's data is on two lines: the first line is the last name, and the second line is the salary.

Process

- Open and read the text file containing employee data.
- Initialize variables to store the total sum of bonuses.
- Read the employee data line by line.
- For each employee:
    - Read the last name.
    - Read the salary.
    - Determine the bonus rate based on the salary:
        - 20% for salaries $100,000.00 and up
        - 15% for salaries $50,000.00 to $99,999.99
        - 10% for salaries below $50,000.00
    - Compute the bonus by applying the bonus rate to the salary.
    - Display the employee's last name, salary, and computed bonus.
    - Add the computed bonus to the total sum of bonuses.
- After processing all employees, display the total sum of all bonuses paid out.

Output

- For each employee, display the last name, salary, and bonus.
- Display the total sum of all bonuses paid out.

4. Create a text file with item, quantity and price. Read through the file one line at a time. Compute the extended price (quantity x price). For each line display the item, quantity, price and extended price. After the loop display the sum of all the extended prices, the count of the number of orders and the average order.

Example Data File

Widget

10

50

Hammer

2

10

Saw

4

8

Etc

Input

- A text file containing item names, quantities, and prices. Each item's data is on three lines: the first line is the item name, the second line is the quantity, and the third line is the price.

Process

- Open and read the text file containing item data.
- Initialize variables to store the total sum of extended prices, count of orders, and the total quantity.
- Read the item data line by line.
- For each item:
    - Read the item name.
    - Read the quantity.
    - Read the price.
    - Compute the extended price by multiplying the quantity by the price.
    - Display the item name, quantity, price, and extended price.
    - Add the extended price to the total sum of extended prices.
    - Increment the count of orders.
- After processing all items, compute the average order by dividing the total sum of extended prices by the count of orders.
- Display the total sum of all extended prices, the count of orders, and the average order.

Output

- For each item, display the item name, quantity, price, and extended price.
- Display the total sum of all extended prices.
- Display the count of the number of orders.
- Display the average order.

5. Create a text file with student last name, district code (I or O) and number of credits taken. Compute tuition owed (credits taken x cost per credit). Cost per credit for in district students (district code I) is 250.00. Out of district students pay 500.00 per credit. For each line display student last name, credits taken and tuition owed. After the loop display sum of all tuition owed and the number of students.

Example file

Jones

I

12

Adams

I

10

Baker

O

12

Smith

O

16

Input

- A text file containing student last names, district codes, and number of credits taken. Each student's data is on three lines: the first line is the last name, the second line is the district code (I or O), and the third line is the number of credits taken.

Process

- Open and read the text file containing student data.
- Initialize variables to store the total sum of tuition owed and the count of students.
- Define the cost per credit for in-district and out-of-district students:
    - In-district (I): $250.00 per credit
    - Out-of-district (O): $500.00 per credit
- Read the student data line by line.
- For each student:
    - Read the last name.
    - Read the district code.
    - Read the number of credits taken.
    - Determine the cost per credit based on the district code.
    - Compute the tuition owed by multiplying the credits taken by the cost per credit.
    - Display the student's last name, credits taken, and tuition owed.
    - Add the tuition owed to the total sum of tuition owed.

    - Increment the count of students.
- After processing all students, display the total sum of all tuition owed and the count of students.

Output

- For each student, display the last name, credits taken, and tuition owed.
- Display the total sum of all tuition owed.
- Display the count of the number of students.