Homework 5
Advanced Python Programming
Due Date: 10/31

1. (*The **Stock** class*) Design a class named **Stock** to represent a company's stock that contains:

   - A private string data field named **symbol** for the stock's symbol.
   - A private string data field named **name** for the stock's name.
   - A private float data field named **previousClosingPrice** that stores the stock price for the previous day.
   - A private float data field named **currentPrice** that stores the stock price for the current price.
   - A constructor that creates a stock with the specified symbol, name, previous price, and current price.
   - A get method for returning the stock name.
   - A get method for returning the stock symbol.
   - Get and set methods for getting/setting the stock's previous price.
   - Get and set methods for getting/setting the stock's current price.
   - A method named **getChangePercent()** that returns the percentage changed from **previousClosingPrice** to **currentPrice**.

   Draw the UML diagram for the class, and then implement the class. Write a test program that creates a **Stock** object with the stock symbol INTC, the name Intel Corporation, the previous closing price of **20.5**, and the new current price of **20.35**, and displays the price change percentage.

2. (*The **Account** class*) Design a class named **Account** that contains:

   - A private **int** data field named **id** for the account.
   - A private float data field named **balance** for the account.
   - A private float data field named **annualInterestRate** that stores the current interest rate.
   - A constructor that creates an account with the specified id (default 0).
   - The accessor and mutator method for **id**, **balance**, and **annualInterestRate**.
   - A method named **getMonthlyInterestRate()** that returns the monthly interest rate.
   - A method named **getMonthlyInterestRate()** that returns the monthly interest.
   - A method named **withdraw** that withdraws a specified amount from the account.
   - A method named **deposit** that deposits a specified amount to the account.

   Draw the UML diagram for the class, and then implement the class. (Hint: the method **getMonthlyInterestRate()** is to return the monthly interest amount, not the interest rate. Use this formula to calculate the monthly interest: **balance * monthlyInterestRate**. **monthlyInterestRate** is **annualInterestRate / 12**. Note that **annualInterestRate** is a percentage (like 4.5%). You need to divide it by **100**.)

   Write a test program that creates an **Account** object with an account id of 1122. a balance of $20,000, and an annual interest rate of 4.5%. Use the XXX method to withdraw $2,500, use the deposit method to deposit $3,000, and print the id, balance, monthly interest rate, and the monthly interest.

3. (*Geometry: n-sided regular polygon*) An *n*-sided regular polygon's sides all have the same length and all of its angles have the same degree (i.e., the polygon is both equilateral and equiangular). Design a class named **RegularPolygon** that contains:

   - A private **int** data field named **n** that defines the number of sides in the polygon.
   - A private float data field named **side** that stores the length of the side.

- A private float data field named **x** that defines the *x*-coordinate of the center of the polygon with a default value of **0**.
- A private float data field named **y** that defines the *y*-coordinate of the center of the polygon with a default value of **0**.
- A constructor that creates a regular polygon with the specified *n* (default 3), side (default 1), *x* (default **0**), and *y* (default **0**).
- The accessor and mutator methods for all data fields.
- The method **getPerimeter()** that returns the perimeter of the polygon.
- The method **getArea()** that returns the area of the polygon. The formula for computing the area of a regular polygon is $Area = \frac{n \times s^2}{4 \times \tan\frac{\pi}{n}}$.

Draw the UML diagram for the class, and then implement the class. Write a test program that creates three **RegularPolygon** objects, created using **RegularPolygon()**, using **RegularPolygon(4, 6)**, and **RegularPolygon(10, 4, 5.6, 7.8)**. For each object display its perimeter and area.