**Lab 7: Objects and Classes**

**Question1:**

Following the example of the **Circle** class in Section 7.2, design a class named **Rectangle** to represent a rectangle. The class contains:

* Two data fields named **width** and **height**. s
* A constructor that creates a rectangle with the specified **width** and **height**. The default values are **1** and **2** for the **width** and **height**, respectively.
* A method named **getArea()** that returns the area of this rectangle.
* A method named **getPerimeter()** that returns the perimeter.

Write a test program that creates two **Rectangle** objects—one with width **4** and height **40** and the other with width **3.5** and height **35.7**. Display the width, height, area, and perimeter of each rectangle in this order.

**Question2:**

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 time.
* 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**.

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 display the price-change percentage.

**Question3:**

Design a class named **StopWatch**. The class contains:

* The private data fields **startTime** and **endTime** with get methods.
* A constructor that initializes **startTime** with the current time.
* A method named **start()** that resets the **startTime** to the current time.
* A method named **stop()** that sets the **endTime** to the current time.
* A method named **getElapsedTime()** that returns the elapsed time for the stopwatch in milliseconds.

Write a test program that measures the execution time of adding numbers from 1 to 1,000,000.