CSCI 241 Assignment 3 Stock Price Breakdown

Due: Thursday, February 25, 2021 at 11:59 p.m.

➔ **NOTE: You earn 2 points for specified email 2 days BEFORE the above deadline. See details on page 3!**

Description

To begin:

- Create a new BlueJ project named **`Assign3Stock`**.
- Create a new class in your BlueJ project named **`StockReport`**.
- Add a `main()` method to **`StockReport`**.

Note: capitalize and spell the project and class names correctly! This is part of your grade.

`StockReport` Description

The United States has had a stock market since the New York Stock Exchange began in 1792. The early stockbrokers modeled their system on that used in Spain. This was largely due to the fact that the U.S. dollar's value had been based on the value of the Spanish *real* (meaning "royal", and pronounced rā-älˈ).

The *real* was the Spanish silver dollar and was divided into eight parts. (This is where the phrase "pieces of eight" comes from.) The *real* could be broken into two, four or eight parts. When the U.S. stock market began, they based the stock values on one-eighth fractions. Stocks traded at not only eighths, but also sixteenths and thirty-seconds, and sometimes even sixty-fourths.

For example, you might have heard something like:
`IBM is currently trading at 57 5/8`

Eighths, of course, do not match neatly with our decimal number system. In 1997, the Common Cents Stock Pricing Act was signed to simplify the stock market by converting the fractions to decimal. This decimalization of the stock market began in August 2000 and was completed within the following year.

For the purposes of this project, we will pretend that we are working with data in the format of the old system, using eighths. So, given a decimal value, your program will calculate and report the equivalent value broken down into eighths, sixteenths, thirty-seconds and sixty-fourths. For example, a stock price of `32.234375` would be equivalent to
`32 + 1/8 + 1/16 + 1/32 + 1/64`.

Here is the algorithm for your program:

1. Print an introductory paragraph explaining to the user what the program will accomplish (see text in examples on next page – the first 6 lines).

2. Use a `Scanner` to ask the user for the company name and save it in a `String`. Since a company name may contain spaces, you must use the `nextLine()` method to read in the name. For example, assume you have already declared a `Scanner` named `input`

that is attached to the keyboard. To read a phrase from the keyboard, you write:
`String phrase = input.nextLine();`

3. Ask the user for a stock price (per share); this will contain a decimal point.

4. Ask the user for a number of shares; you may assume this is an integer.

5. Calculate the stock price (per share) in eighths, sixteenths, thirty-seconds, and sixty-fourths.

6. Print a report (sample shown below) containing the following information:
   - company name, as read in
   - the stock price per share, as read in (in decimal format)
   - the stock price per share using eighths, sixteenths, thirty-seconds and sixty-fourths
   - the total value of the stock (price multiplied by number of shares) – this number should be rounded to 2 decimal places (sound familiar?)

Here are 2 examples of how your output should look (input data is **bold and underlined** to distinguish it from what the program prints): \

```
Welcome to Ye Olde Stock Report!
--------------------------------
This program will report a stock price
in historic format using eighths, as well
as the current value of the holding.

Enter a company name:  Microsoft Corporation
Enter the price of the stock:  243.79
Enter the number of shares held:  50

=======================================
Report for Microsoft Corporation
---------------------------------------
Current stock quote value:  $243.79
'Old' stock quote value:    $243 6/8 0/16 1/32 0/64
Total value of stock shares:  $12189.5
```

```
Welcome to Ye Olde Stock Report!
--------------------------------
This program will report a stock price
in historic format using eighths, as well
as the current value of the holding.

Enter a company name:  IBM
Enter the price of the stock:  120.73
Enter the number of shares held:  25

=======================================
Report for IBM
---------------------------------------
Current stock quote value:  $120.73
'Old' stock quote value:    $120 5/8 1/16 1/32 0/64
Total value of stock shares:  $3018.25
```

`StockReport` Hints

Review how the change was calculated in `ComputeChange.java` on pages 63-64 of your textbook. The calculations for the fractional part of the stock price will be very similar. In this program, the original number is multiplied by 100, because a penny (0.01 cent) is the smallest coin. For this program, your "coins" are 1/8th of a dollar, 1/16th of a dollar, etc.
Here are the equivalencies:

1/8 = 12.5 cents	1/32 = 3.125 cents
1/16 = 6.25 cents	1/64 = 1.5625 cents

By what factor should you multiply your original number to be able to work with an integer to find your values? In the ComputeChange program, you see it starts by multiplying the original decimal number by 100 because .01 is the worth of a penny, the smallest coin. So, if 1.5625 cents is the smallest value you will use, what would you use for a multiplication factor to make sure all of the digits you need are to the left of the decimal point?

2 points of your score can be earned by emailing your instructor:

- The number you will multiply the stock price by to get all digits to the left of the decimal so you can work with integer-only math
- The calculation for the number of eighths and how you find the remainder

Send an email with this enclosed code to your instructor by 11:59 p.m. on Tuesday, February 23. The rest of the assignment has the deadline of Thursday, February 25 at 11:59 p.m.

Submission Requirements

1. **Computer Science Lab Submission**: Submit your complete **`Assign3Stock`** project through BlueJ's submission process *from your Computer Science lab account*. This will send a copy of your completed implementation over to be graded. Remember, you can send it more than once if you need to make changes.

2. **Canvas Submission:** upload a copy of your `StockReport.java` file to Canvas.

Grading Criteria

Your programs *must compile without errors* to be accepted. All of the following criteria will be used to evaluate your programs:

Correct submission of project	2 pts
Appropriate and complete comments	6 pts
Adherence to style conventions (identifier names, indentation, etc.)	4 pts
Correct program behavior (including clear output with correct spelling)	13 pts
Total points:	**25 pts**