HW2

Textbook Questions: 3.8, 3.22, and 3.23a (3.23b and c are not required)
70 points
Due Thursday, September 27th, 11:59pm, submitted on Canvas

Written Question
3.8 (10 points)

3.8 The following routine removes the first half of the list passed as a parameter:

```
public static void removeFirstHalf( List<?> lst )
{
    int theSize = lst.size( ) / 2;

    for( int i = 0; i < theSize; i++ )
        lst.remove( 0 );
}
```

a. Why is `theSize` saved prior to entering the `for` loop?
b. What is the running time of `removeFirstHalf` if `lst` is an `ArrayList`?
c. What is the running time of `removeFirstHalf` if `lst` is a `LinkedList`?
d. Does using an iterator make `removeHalf` faster for either type of `List`?

Java Program
3.22 and 3.23a (60 Points)

3.22 Write a program to evaluate a postfix expression.

3.23 a. Write a program to convert an infix expression that includes (,), +, -, *, and / to postfix.

Write a Java class called InfixToPostfix that accepts a String in the constructor called *expression*. This string will contain a infix mathematical expression. Include a method called *convert*() which returns the postfix version of the mathematical expression. *Convert*() must use a stack, to convert the expression, as shown in class.

Write a Java class called PostfixEvaluate that accepts a String in the constructor called *expression*. This string will contain a postfix mathematical expression. Include a method called *evaluate*() which calculates the postfix mathematical expression, and returns the result. *Evaluate*() must use a stack, to evaluate the postfix expression, as shown in class.

Write a program, TestInfixToPostfix.java, which reads a file called input.txt. Input.txt will contain an infix mathematical expression that includes: (,), +, -, *, and /. You do not need to include exponents in your expression.

-In your test program, create a InfixToPostfix object with the expression from input.txt. Call the *convert*() method to obtain the postfix expression. Print the postfix expression to the console.
-Create a PostfixEvaluate object with the postfix expression returned from *convert*() method above. Call the *evaluate*() method to calculate the postfix expression, and print the results to the screen.

Note: the 2 classes and test program, will satisfy all the requirements for 3.22 and 3.23a.

Style Guidelines: Use good comments, variables, and coding style to make your code as easy to read and understand as possible.

Example:

Input.txt
(1+2*4)/3-1+6/3

Output to console
Postfix expression: 124*+3/1-63/+
Result: 4

The grader will use a different input.txt for verification.