Data Structures – Infix to Postfix Grade Sheet

Name: **Buchanan, Brandon William** in section: **201**

Item	Points		Comments
	Earned	Possible	
Project arrived successfully - Contains all files it should (.sln, .csproj, .cs,.ico, etc.) - The zipped file is named according to conventions described in the Course Facts - Sent to both ***professor*** and ***Teaching Assistant***	10	10	
Program compiles and builds without errors - Compiles without errors - Warnings are minimal/may be safely ignored - Build was successful - Includes all assigned features	5	5	
Documentation - Each file has appropriate ***header comments*** - Methods have ***XML (///) block comments*** - Individual ***comments*** used appropriately	6	10	Not all classes are commented.
Main Program and User Interaction - Menu driven or GUI - Allows for input of infix expressions from a text file that the user can select using a standard ***OpenFile*** dialog; accepts grader's input file	15	15	
Utility Class - Used where appropriate rather than re-writing the parsing, Welcome message, and other previously written code	15	15	
Operator Class - Represents ***operator symbol*** and its ***precedence*** - ***Precedence values*** are ***reasonable*** for the conversion algorithm described in the assignment - Supports ***comparisons*** of ***two operators*** based on ***precedence***	15	15	

Postfix Class
- Maintains properties for the ***infix*** and ***postfix*** expressions
- Contains a ***Convert*** method
- ***Convert*** method makes appropriate use of the ***Utility*** and ***Operator*** classes to avoid
  o Duplication of effort
  o Excessive brute force
- Utilizes a ***Stack<Operator>*** as described in the assignment
- Correctly identifies and reports ***unpaired parentheses***
- ***Convert*** produces ***correct results***
	12	15	Doesn't identify unpaired parenthesis.

Results are correct - Program runs without crashes - Output is appropriate and reasonably professional looking - Results are correct	12	15	Invalid parenthesis is not considered.
Total	90	100	

Brandon Buchanan

- File not named according to course fact sheet.
- Environment in which data collected needs to be specified (Visual Studio Ultimate 2010 Profiler).
- Should describe the process how data is collected.
- Contains grammatical errors.
- References (may be in the form of footnotes) need to be made to data in supporting the observations.
- The statements are too generic. They may change for different sizes of datasets.
- For large numbers of data members, the quicksort is the most efficient sorting algorithm when about 90% of the integers in the list are already sorted. → Doesn't hold for smaller sets.
- Doesn't relate the findings to the theoretical performance of the sorts.
- More comparisons need to be made on the data collected using combinations of size, data, and algorithm.

Grade: 85