Assignment 3 – Part 2 & 3

Part 2 and 3 due March 19 at midnight

We use the same instructions for grading as for previous assignments.

Problem.

- Write a C++ program for a simple calculator based on expression postfix form.
  1. Part 2: implement templated stack and queue data structures using the templated linked list class.
  2. Part 3:
     (a) implement the class `parser` with a function `to_postfix()` to translate input infix form of an algebraic expression into its postfix form using stack and queue as the auxiliary data structures, see lecture notes for the algorithm.
     (b) implement the class `evaluator` with a function `evaluate()` to get the value of an algebraic expression based on its postfix form. Notice that during the evaluation process of an expression the stack data structure should be used.
- Useful hints.
  - Use the templated linked list class from Part 1 of the assignment to implement the templated versions of the queue and stack data structures. Test the correctness of the queue using string type and floating point type (type `double`).
  - The infix expression will consist of a valid combination of the following operators: $(,), +, -, *, /, \hat{}$. The operator $\hat{}$ stands for the exponential operator (raising to the power).
  - Operands include 26 variables, namely, a through z. Values for the operands can be entered from the keyboard (interactively).
  - Throw the exception "`Invalid input`" in the case when an invalid name of a variable or incorrect operator is used.
  - Read the expression from the input token by token until the termination character # of the expression is found. Then your program should be able to display the infix queue. Write a string tokenizer function to extract tokens from the input string.
  - Print out the postfix queue.
  - Evaluate the obtained postfix expression using the algorithm described in class. Replace variables $a, .., z$ by associated values.
- The basic user menu for this program
  1. Read an infix expression from the keyboard
  2. Check whether or not parenthesis are balanced correctly
  3. Display a correct infix expression on the screen or a message that the expression is invalid
  4. Convert infix form to its postfix form and display a postfix queue on the screen
  5. Evaluate postfix form of the expression for floating point values entered from the keyboard
  6. Display the value of an algebraic expression on the screen
- What to include in the assignment report?
  1. Provide the design of your program for the parts 2 and 3 of the assignment. Write about the relation between classes and justification why you chose them.
  2. Describe each class private and public members, your algorithms and their implementations.

3. Write the names of the templated classes of your assignment. Which types have you used to test them for correctness?
4. Describe all tests done to verify correctness of your program for the parts 2 and 3 of the assignment. Give an explanation why you chose such tests. Include your tests in your assignment report.

- Please submit to CSNet a tar file that includes:
  - The templated version of the linked list, part 1 of the assignment.
  - The templated version of the queue and stack classes based on doubly linked list (part 2)
  - The parser class (part 3)
  - The evaluator class (part 3)
  - The testing class with the main function and the menu of this program.