ENGG*1410: "Introductory Programming for Engineers", Assignment #12 "Abstract Data Types"

Prof. Shawki Areibi
School of Engineering, University of Guelph
Fall 2021

Start Date: Week #12, Due Date: Week #12 (Friday, 5:00 PM) in Dropbox

1. Write a program that reads in a sequence of characters and prints them in reverse order. Use a Stack implementation.

2. Write a program that reads in a sequence of characters, and determines whether its parentheses, braces, and curly braces are "balanced.". Hint: for left delimiters, push onto stack; for right delimiters, pop from stack and check whether popped element mathces right delimiter.

3. Write a program that reads in a positive integer and prints the binary representation of that integer. Hine: divide the integer by 2.

4. Write a function called *insertEntry()* to insert a new entry into a linked list. Have the procedure take as arguments a pointer to the list entry to be inserted (of type struct entry as defined earlier in the previous question (2), and a pointer to an element in the list after which the new entry is to be inserted.

5. Write a function called *removeEntry()* to remove an entry from a linked list. The sole argument to the procedure should be a pointer to the list. Have the function remove the entry after the one pointed to by the argument. (why can't you remove the entry pointed to by the argument?) You need to use the special structure you set up in the previous exercise to handle the special case of removing the first element from the list.

6. A *double linked list* is a list in which each entry contains a pointer to the preceding entry in the list as well as a pointer to the next entry in the list. Define the appropriate structure definition for a doubly linked list entry and then write a small program that implements a small doubly linked list and prints out the element of the list.

7. Develop *insertEntry()* and *removeEntry()* functions for a doubly linked list that are similar in function to those developed in previous exercises for a singly linked list. Why can your *removeEntry()* function now take as its argument a direct pointer to the entry to be removed from the list?