Lab 05 Reverse Polish Calculator

Bradley Grose

# Problem

In this lab, we must make a reverse polish calculator. This must be able to take in a string and calculate the value using the reverse polish notation, which means the operator is at the end of the expression of numbers. We must use a stack and push the operations and numbers to create this calculator to solve the reverse polish notation problems

# Solution

Using the given driver and stack linked list interface, I first created the linked list stack class. First I created list nodes which has the data and a link for the data, as well as an establishing the data and link creation. Then I have an Iterator for it, which sets the curr to the head, then checks if it has a next function, as well as an advancement function through the list. Then I set up the stack, starting with a null stack, then a push function which creates a list node and adds it to the top of the stack. I also made a pop function which gets ride of the node on the top of the function. Then there is a peek function which gives the top function. Finally, there is a print function to print the nodes using a while loop. Then in the reversePolishCalculator function, I first create the object of a stack, using type integer. Then I have the calculate function, which creates a parse scanner that looks at the input function. If the value is a number, it will push it onto the stack. If it is a sign, such as + - / \*, it will first check to make sure it is a proper operation with more than 2 values. Then it takes the two numbers and does the operation, checking to make sure its not divided by 0 for the division problem. Finally, there is a clear stack function that removed everything from the stack.

# Implementation Problems Encountered

No problems were encountered in this lab.

# Lab Report Questions

1)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | Head |
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | Head |  |  |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  | Head |
| 15 | 14 | 13 | 12 | 11 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | Head |  |  |
| 12 | 11 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |  |  |

2)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Head |  |  |  |  |  |  |  |  |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Head |  |  |  |  |  |  |  |  |  |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Head |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Head |  |  |  |  |  |  |  |  |  |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |