COMSC-200
Lab 10

Ryan Jacoby

8 November 2020

1 R14.27

In our circular array implementation of a queue, can you compute the value of len from the values of the *head* and *tail* data members? Why or why not?

No, the len must be kept track of or can be calculated if the head and tail are kept track of. There is no way to calculate the length from the values stored at head and tail, but it is easy to calculate len if the locations of head and tail are known along with the capacity of the circular array.

2 E14.21

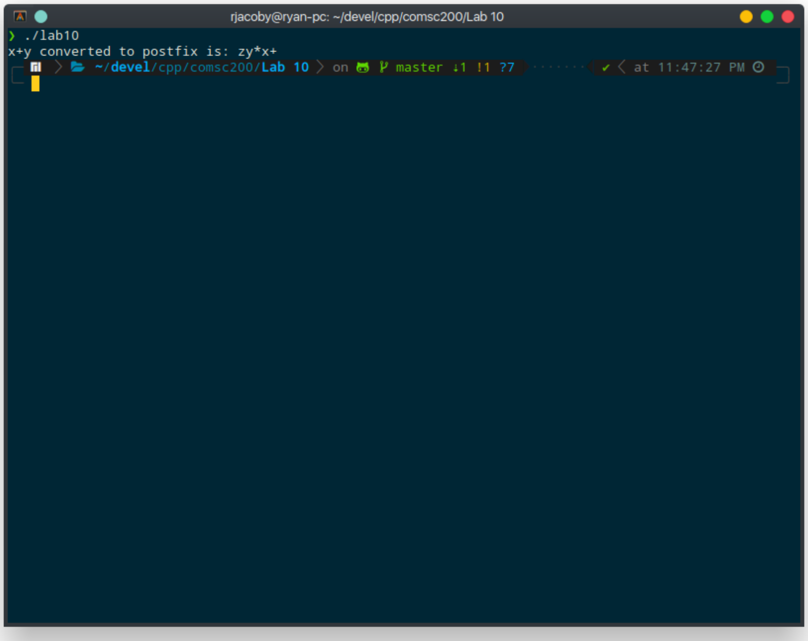
I currently do not have access to the textbook; I will e-mail you my solution when I do.

3 Infix to Postfix Conversion

```
// Ryan Jacoby

#include<iostream>
#include<stack>
#include<string>

using namespace std;

string convertToPostfix(stack<char>);

int main() {
    stack<char> equation;

    equation.push(’x’);
    equation.push(’+’);
    equation.push(’y’);
    equation.push(’*’);
    equation.push(’z’);

    cout << "x+y*z converted to postfix is: ";
    cout << convertToPostfix(equation) << ’\n’;

    return 0;
```

```cpp
}

string convertToPostfix(stack<char> infix) {
    stack<char> operatorStack = stack<char>();
    string postfix = "";

    while(!infix.empty()) {
        char nextCharacter = infix.top();
        infix.pop();
        switch(nextCharacter) {
            case 'a' : case 'b' : case 'x' : case 'y' : case 'z':
                postfix += nextCharacter;
                break;
            case '^':
                operatorStack.push(nextCharacter);
                break;
            case '+' : case '-' : case '*' : case '/':
                while(!operatorStack.empty()) { // Implement elegant way of anding with operator precedence; nextChar <= operatorStack.top();
                    postfix += operatorStack.top();
                    operatorStack.pop();
                }
                operatorStack.push(nextCharacter);
                break;
            case '(':
                operatorStack.push(nextCharacter);
                break;
            case ')':
                {
                    char topOperator = operatorStack.top();
                    operatorStack.pop();
                    while(topOperator != '(') {
                        postfix += topOperator;
                        topOperator = operatorStack.top();
                        operatorStack.pop();
                    }
                    break;
                }
            default: break;
        }
    }

    char topOperator;
    while(!operatorStack.empty()) {
        topOperator = operatorStack.top();
        operatorStack.pop();
        postfix += topOperator;
    }

    return postfix;
}
```

Listing 1: main.cpp



```
❯ ./lab10
x+y converted to postfix is: zy*x+
  > ~/devel/cpp/comsc200/Lab 10 > on  master ↓1 !1 ?7 ········ ✔ < at 11:47:27 PM
```