Data Structures - Session 2 - Assignments

1. Define the stack interface and use it to implement the method for evaluating an infix integer arithmetic expression. The method will take the infix expression as a string and tokenize it to extract the integers and operators. You can assume the expression has only integer constants, and the tokens are separated using spaces. The expression will support variables, arithmetic operators (+, -, \*, /), relational operators (==, !=, &lt;, &gt;, &lt;=, &gt;=), and boolean operators (&amp;&amp;, ||, !), and parentheses. You can assume that arithmetic operators will not be used in unary form. Use the Java rules for precedence and associativity of operators.
2. Define the queue interface and implement it using an array. The queue would support methods to add an item to a queue, delete an item from the queue, check whether the queue is empty, check whether the queue is full. Address the issue related to queue getting full despite unused space in the array by providing circular implementation of the queue.
3. Using the queue interface, implement the counseling process of a College.
   1. The program will receive a list of N programs that the college offers as input. The list will have the name and capacity of each program.
   2. The program will also receive the list students sorted in order of rank. For each student it will have the name of the student, and list of 5 program options defining the preference of the student. The list will be sorted in order of preference.
   3. Add the students to a queue in the received order.
   4. Once the students have been added, process them one by one and allocate programs. If none of the options chosen by a student are available then he will not be allocated any program.
   5. The program should output the list of students along with their allocated programs.