**LAB 03**

**Stack and Queues**

Nguyễn Minh Đạt – ITDSIU22166

Problem 1: Simple Stack Application

1. Convert a decimal number and convert it to octal form

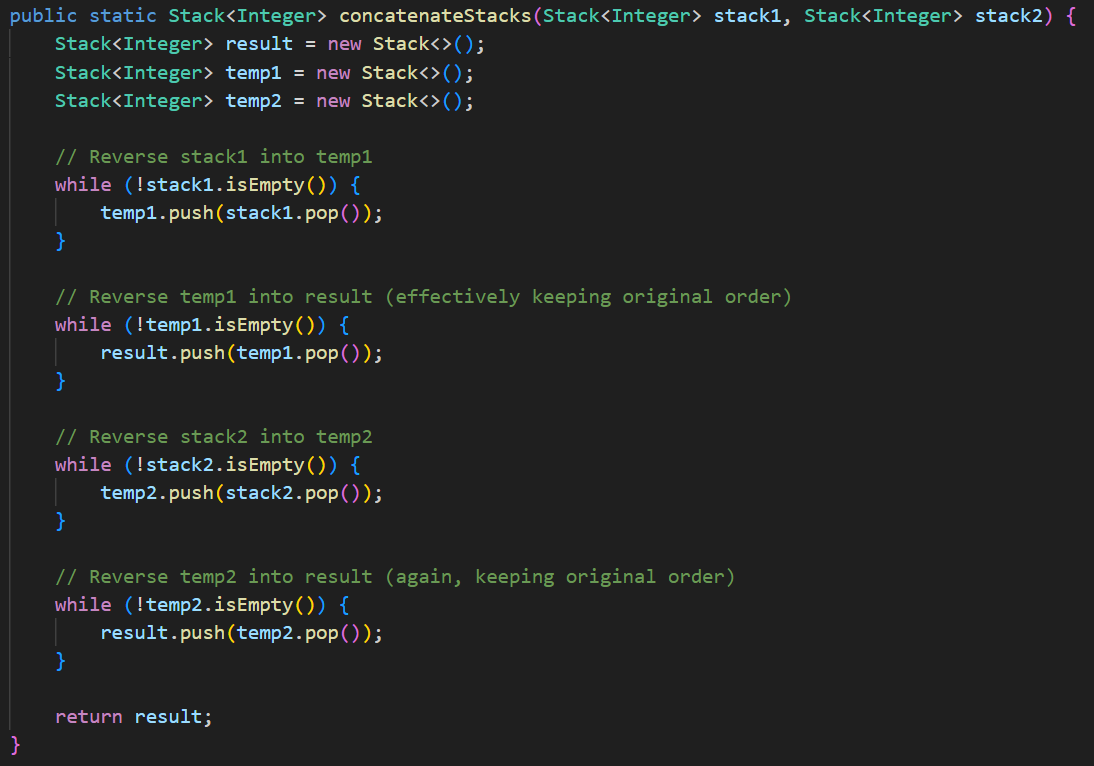
A computer screen with colorful text

AI-generated content may be incorrect.

The algorithm works because:

* Octal is base-8, so we repeatedly divide by 8 using a while loop.
* The remainders give us the octal digits from right to left.
* Reversing at the end gives us the correct order. The result will be the octal form of the decimal number.
* If the input is 0, it immediately returns "0" since 0 in decimal is 0 in octal. This prevents unnecessary processing for this edge case

1. Concatenate two stacks.

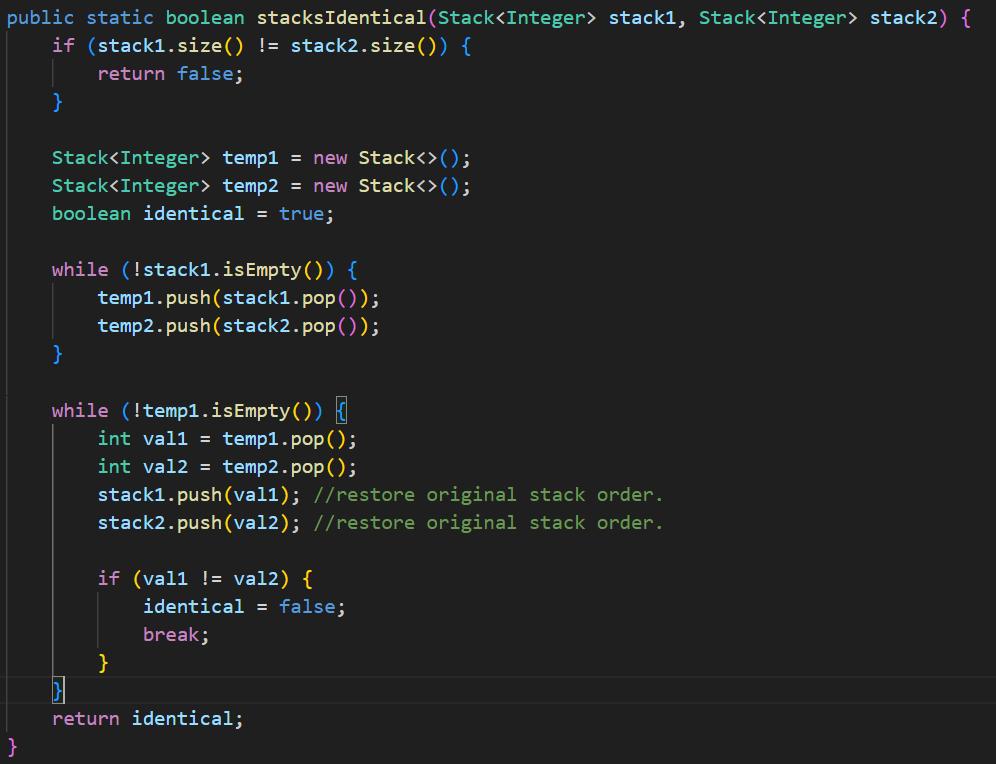


The algorithm works because:

* I use the LIFO structures for Stacks, so direct transfer would reverse order of the array.
* Using two reversals (via temporary stacks) preserves the original order of the array.
* First stack's elements end up at the bottom, second stack’s elements end up at the top.
* The double-reversal process is needed because we can't access stack elements in their original order directly

Example with stack1 = [1,2,3] (top is 3) and stack2 = [4,5] (top is 5):

1. stack1 → temp1: [3,2,1] (top is 1)
2. temp1 → result: [1,2,3] (top is 3)
3. stack2 → temp2: [5,4] (top is 4)
4. temp2 → result: [1,2,3,4,5] (top is 5)
5. Determine if the contents of one stack are identical to that of another.



The algorithms works because:

* First the method will check the Size of two stacks. If they are not the same, they can't be identical, so they return false immediately.
* Creates two temporary stacks to store elements during comparison, Initializes identical flag as true
* Use the while loop to push them to respective temporary stacks and reverse the order of both stacks.
* Use another while loop to compare each pair of elements. If any pair doesn't match, sets identical to false and breaks

Output:

A screenshot of a computer program

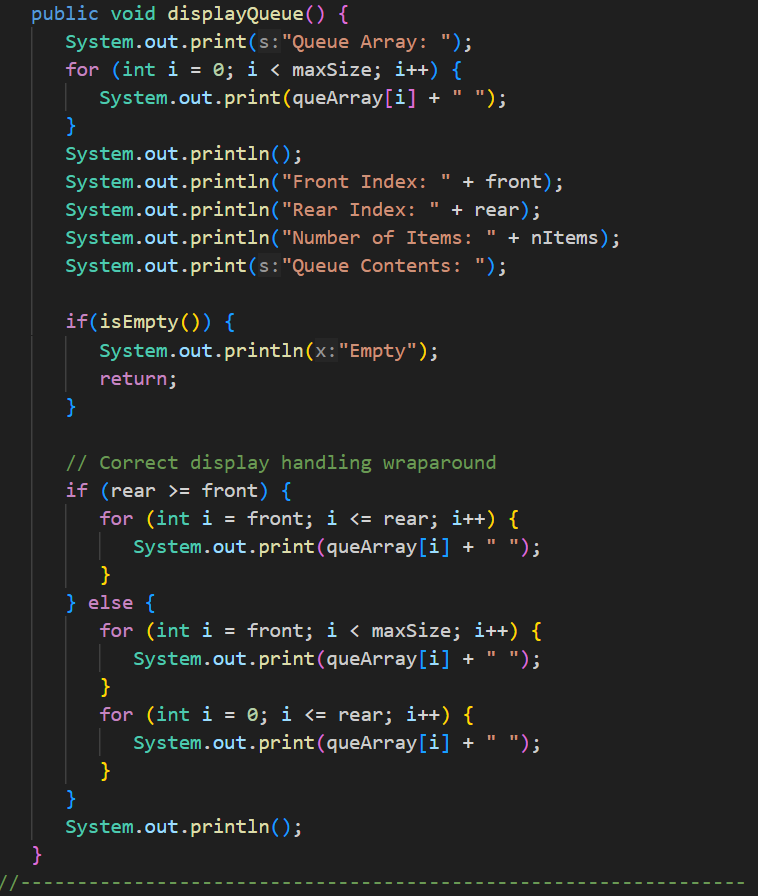
AI-generated content may be incorrect.

Problem 2: Arithmetic Expression Evaluation

1. Can calculate postfix and infix
2. Multiple digit numbers in expressions
3. Allow variables in evaluation:

Problem 3: QueueApp.java

1. Display the array, the queue, and the front and rear indices.



The algorithm works because:

* It uses a for loop to print out the array with condition is maxSize. So, if the array is empty, the result will be [0 0 0 0]
* Then it will display the front, rear index, number of items and the content of the queue
* The if-else condition will make sure it will display by the last item.

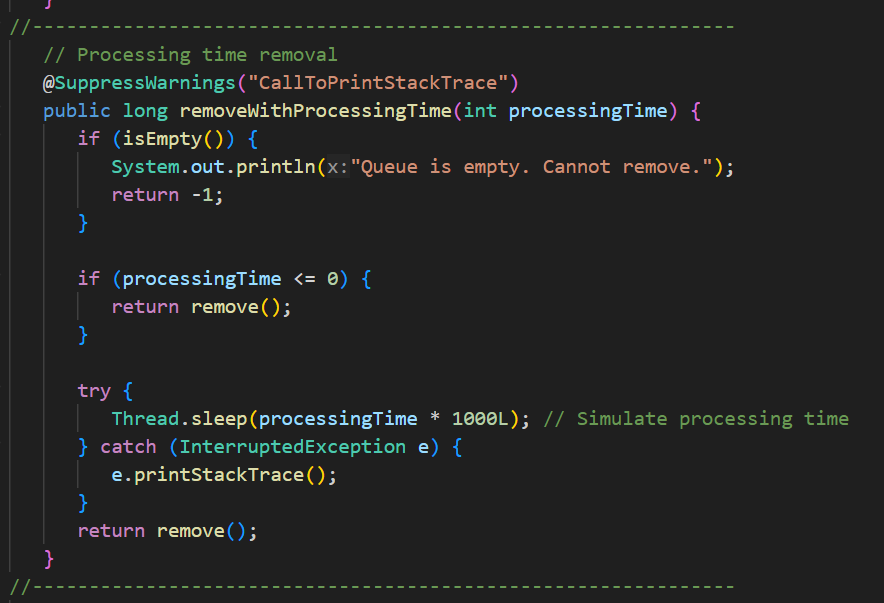
1. Extend the insert and remove methods to deal with a full and empty queue.

A computer screen shot of code

AI-generated content may be incorrect.

The method now has a if condition to check if the queue array is full or empty. If the array is full, the insert method will throw an exception and break, as same as the remove method.

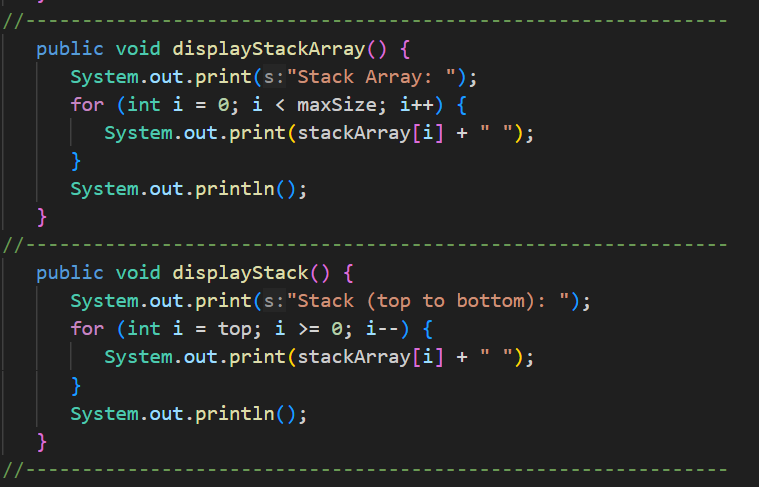
1. Add processing time to the queue. Create a new remove method that removes item N after N calls the method.



The method now have a processingTime variable (seconds) for calling the remove() method. If the array is empty, it will throw an exception and return -1. If the processingTime <= 0, it will call the remove() method immediately. In other case, the program will run for X seconds with X is processingTime, after the time, the method will call the remove().

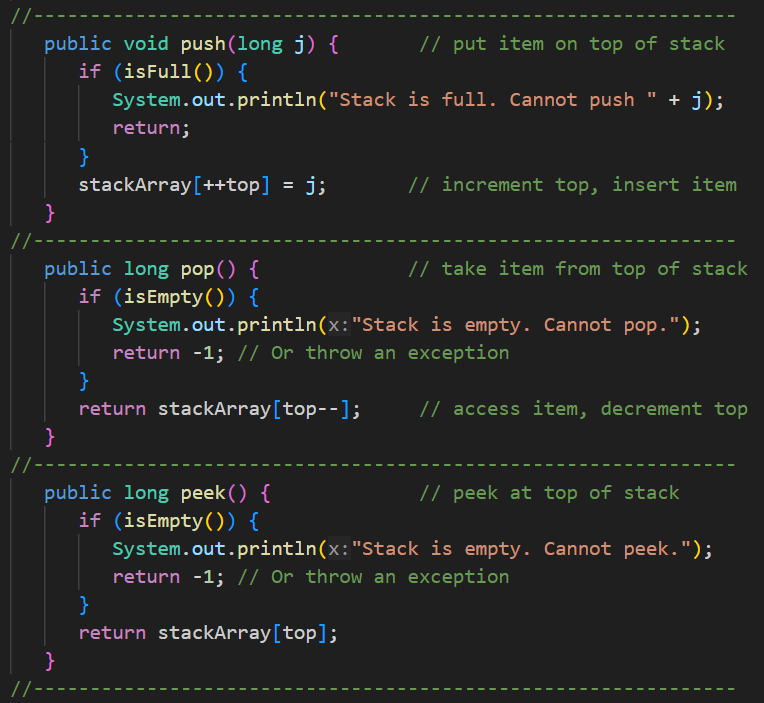
Problem 4: StackApp.java

1. Display the stack array and the stack itself.

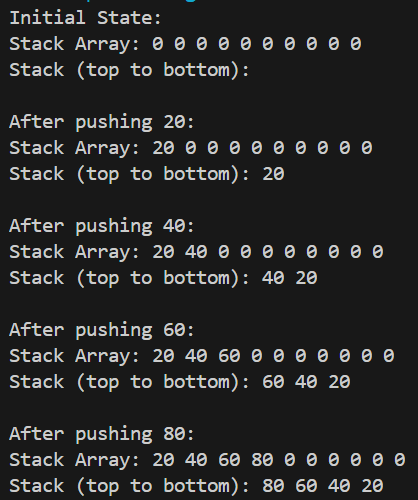
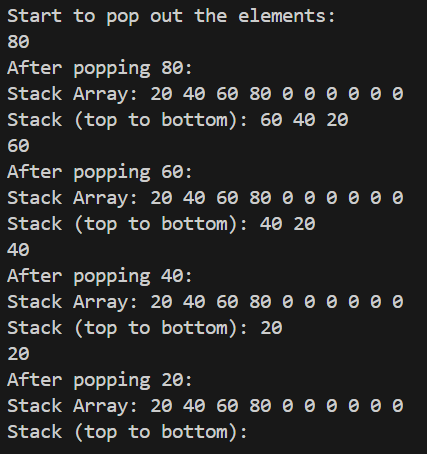


These two method will display the Stack array with [a b c] while the displayStack() will display a b c. Both method use a for loop for display the content.

1. Extend the push and pop methods to deal with a full and empty stack.

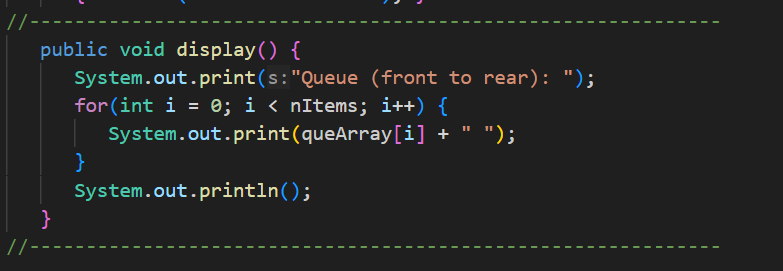


These methods have been added the if conditions for checking if the stack array is full or empty. If the condition is true, they will throw an exception.

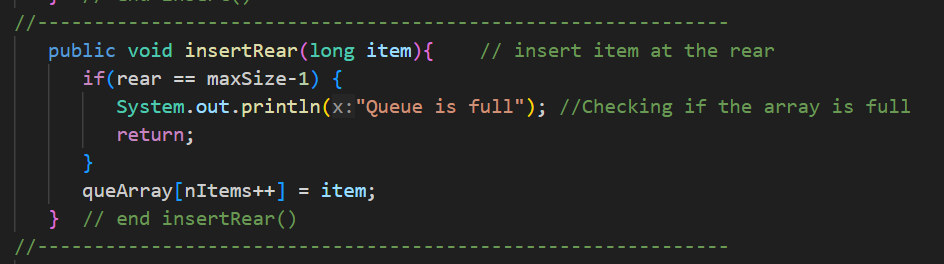
Problem 5: PriorityQApp.java

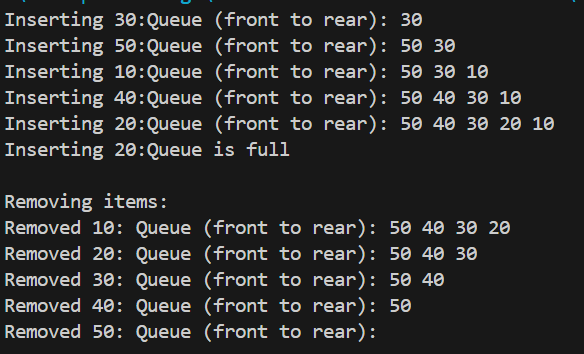
1. Write a method to display the queue and use to trace the queue operation.



The method use a for loop for displaying the Queue by checking the nItems.

1. Modify the insert method to insert the new item at the rear. Compare this queue with QueueApp.java. Which one is more efficient?

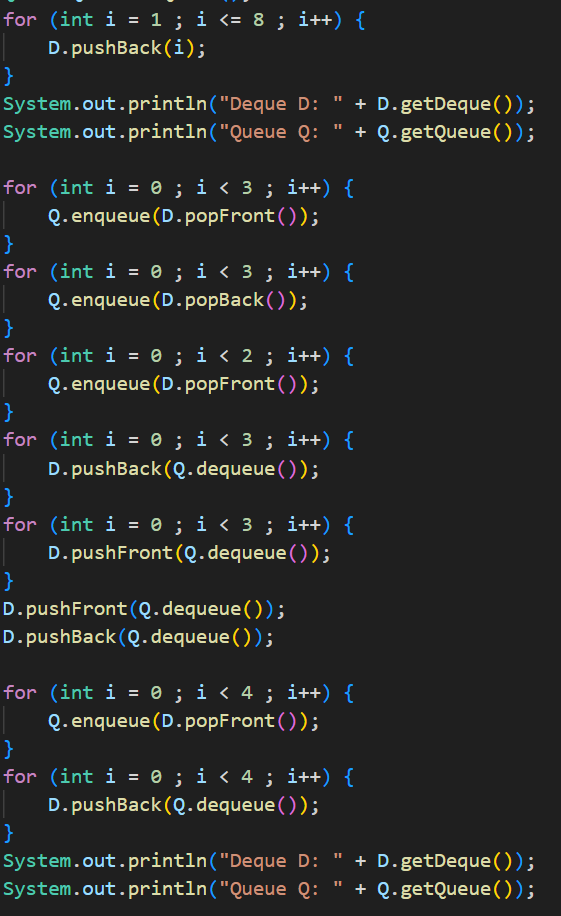


A screenshot of a computer program

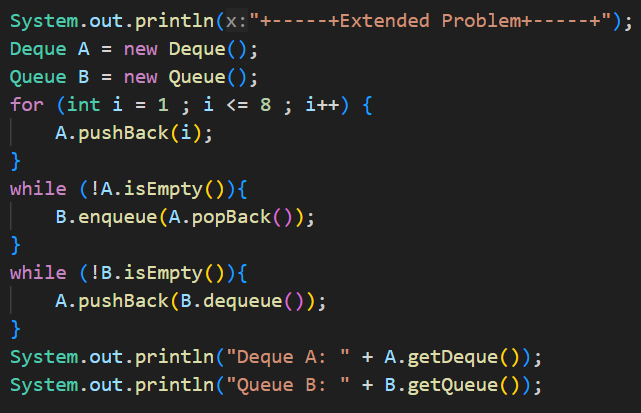
AI-generated content may be incorrect.

Problem 6: Challenge.java

1. Basis Operation:

+

1. Extended Problem:



A screen shot of a computer program

AI-generated content may be incorrect.