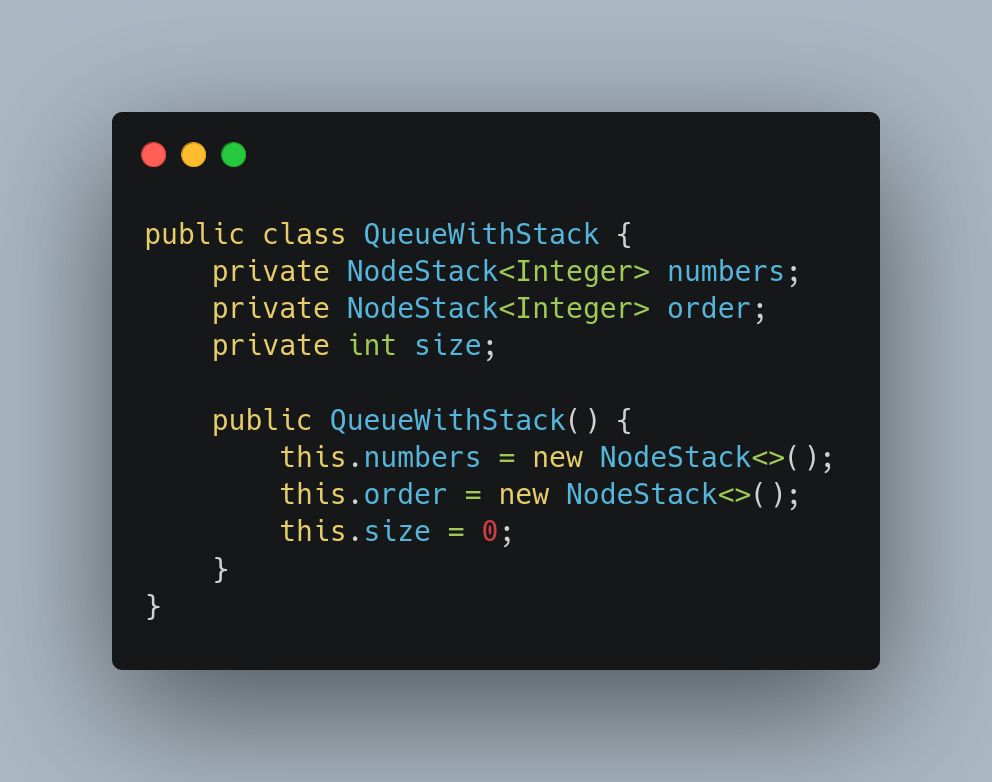
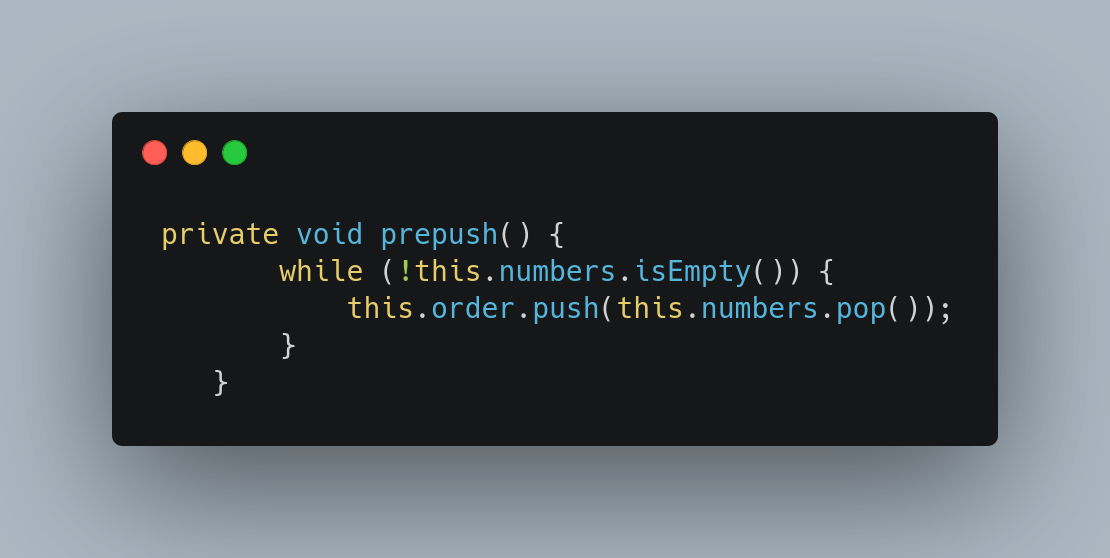
Esta classe contém três atributos:

* Uma pilha para guardar os números
* Uma pilha para ajudar na organização destes números
* Um atributo para guardar o tamanho da lista

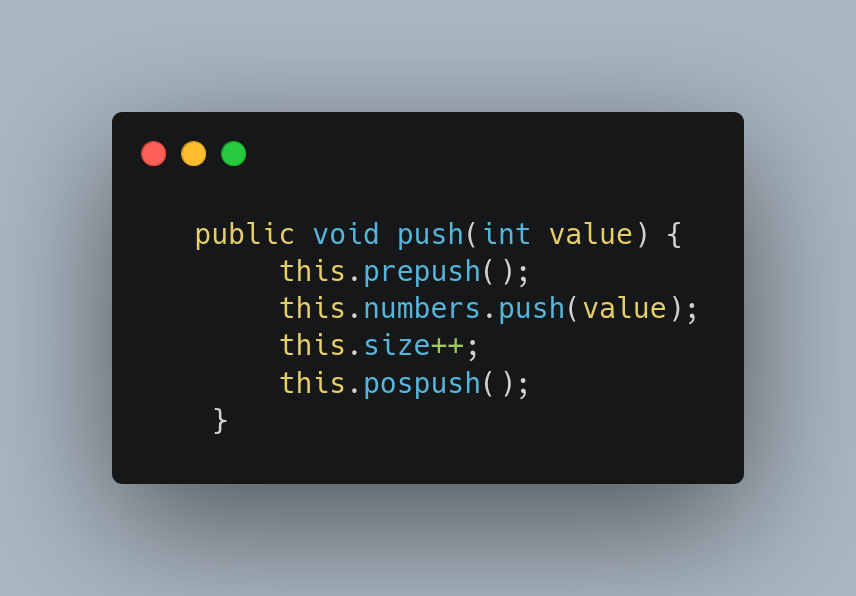
Como pode ser visto aqui:



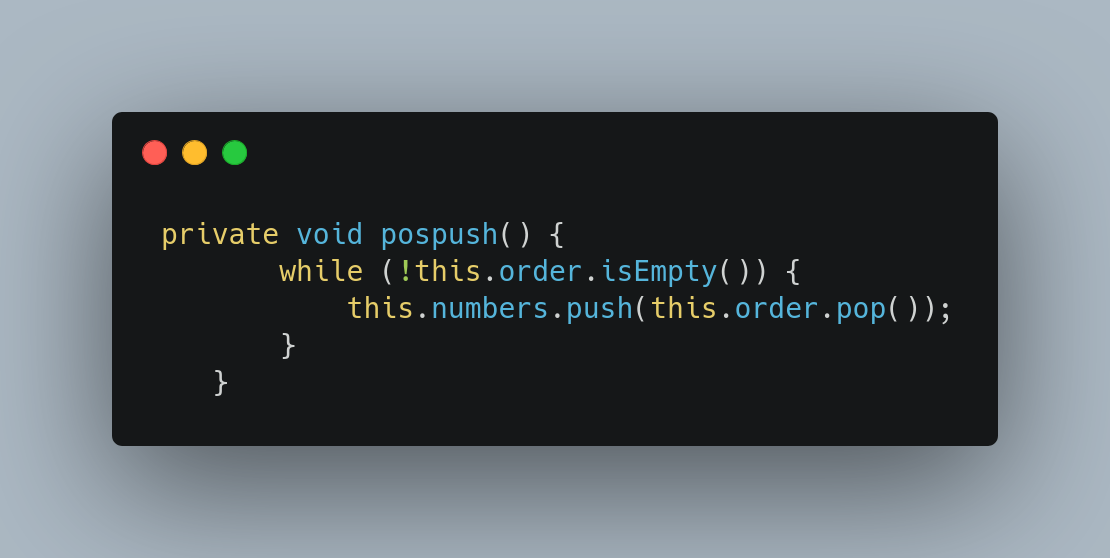
O método inicial a ser implementado nesta classe é o "push". No entanto, antes de realizar essa operação, é crucial considerar que essa classe deve se comportar como uma fila. Portanto, o primeiro número a entrar deve permanecer no topo da pilha, o segundo como o segundo, e assim por diante. Para manter essa ordem, foi desenvolvido o método "prepush", que prepara as duas pilhas para a execução do método "push":



Este método tem como propósito preencher a pilha “order” e esvaziar a pilha “numbers”. Uma vez que esta operação foi concluída é possível passar para o próximo passo:

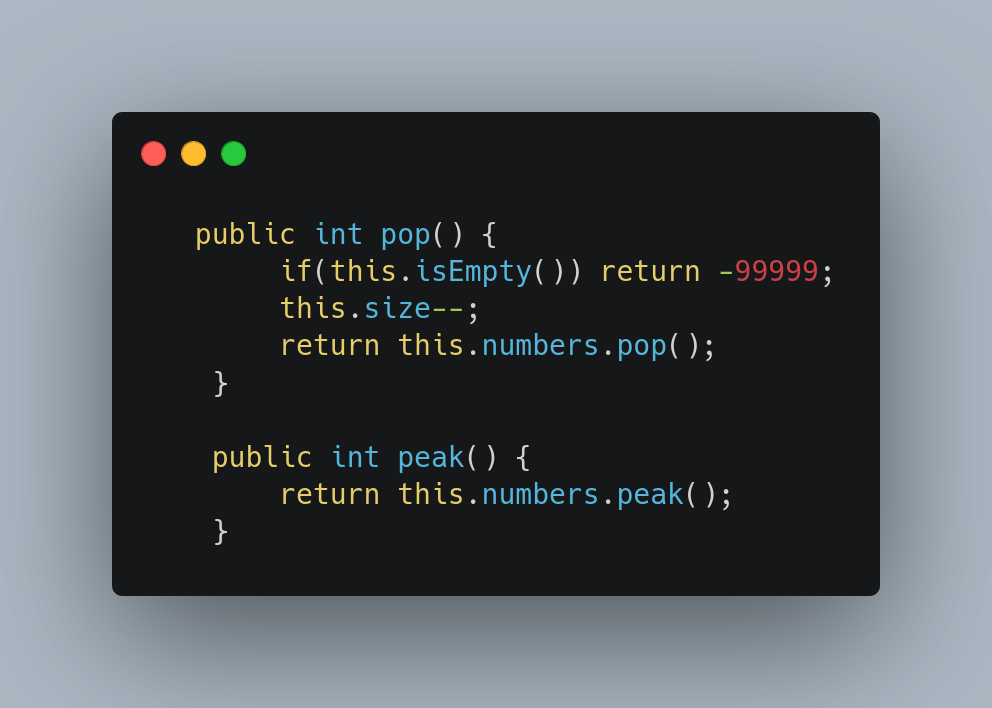


Após a adição do número à pilha e o aumento do tamanho da fila, os outros números que estavam previamente na pilha "numbers" precisam retornar às suas posições originais. Para garantir isso, o método "pospush" é invocado:



Ele devolve os números para seus devidos lugares, garantindo que a ordem dos números permaneça a mesma de antes.

Então, temos o método "pop", que remove o número mais antigo adicionado à pilha, e o método “peak” que devolve este número sem fazer alterações na fila:



Ambos os métodos fazem uso de funções já existentes na classe NodeStack para executar essas operações. Isso funciona porque os números estão todos na ordem correta, graças ao método "push".

Por fim, temos o método "isEmpty", que simplesmente verifica se o atributo "size" é zero, e o método "show", que utiliza outra função pré-existente da classe NodeStack para exibir os elementos dentro da fila. Novamente, a razão pela qual o método "show" funciona corretamente é devido às operações realizadas durante a execução do método "push".

