Análise de Complexidade

1.(a) Na implementação do meu deque, todos os métodos operam em $O(1)$, com exceção do método libera(), que tem custo $O(n)$;

(b) $O(1)$ para as operações de TAD Pilha e $O(n)$ no método libera().

(c) De forma análoga: $O(1)$ e $O(n)$ apenas para liberar

2. $O(n)$ minha ~~minha~~ solução consiste em usar uma fila para ir "segurando" os elementos enquanto novos não adicionados. Por exemplo:

- quero empilhar a e depois b

Fila1 = a, agora vamos adicionar b

Fila 2 = a e o tiramos da fila 1, em seguida

adicionamos b a Fila 1 e depois a. Gerando a Fila b → a

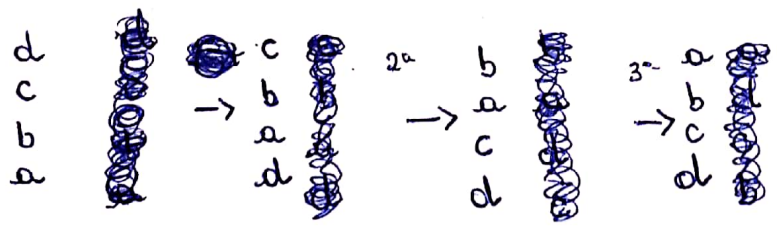
3. A solução é análoga a anterior e também possui complexidade $O(n)$;

4.(a) Essa é simples, basta ir dando pop na pilha e enfileirando, depois é só botar tudo na pilha ~~pilha~~ de novo. $O(n)$.

(b) Nessa, basta empilharmos em p em p1, p1 em p2 e por fim p2 em p: $O(n)$;

*(c) Aqui eu basicamente pensei em remover um elemento da pilha original p e o colocar em uma variável auxiliar, depois transfiro os elementos que sobraram para outra pilha p1, assim eles são empilhados na ordem reversa, por fim, coloco no elemento que removi no começo de volta em p (faço isso a cada iteração).

Ai vou ter uma coisa mais ou menos assim:



↳ Perdão pelo erro =(

No for, a função inverte é chamada 2x e ela vai de i até n - i - 1, como ela é chamada 2 vezes a cada iteração do loop e o loop itera n vezes: $O(n^2)$.

5. (a) Similar 4 (a), complexidade O(n)

(b) Aqui eu pensei em percorrer a fila até o último elemento, pegar ele e enfileirar na outra, depois fazer isso para o antepenúltimo e etc: $O(n^2)$.

6. Como já conversamos sobre as complexidades de pilha anteriormente, vou focar mais no obterMinimo(). Decidi usar uma minHeap para manter os elementos da pilha sempre ordenados. O "problema" é que preciso atualizar o vetor que é minha HeapMin a cada inserção / remoção de elemento na pilha, no entanto, essa operação não é tão custosa, apenas O(lgn). Por outro lado, desempilhar um elemento requer que antes de atualizar a Heap eu a refaça então preciso copiar os elementos da pilha para o vetor da heap, o que custa O(n

7. Essa solução é O(n). Minha ideia foi basicamente percorrer a expressão e usar uma pilha para organizar os operadores de acordo com as regras da Notação Polonesa Reversa.