Exercícios Estruturas Simplesmente encadeadas

1- Crie um programa que gerencie uma PILHA de TAREFAS a serem cumpridas. As tarefas são Strings que descrevem uma ação a ser executada. Crie um menu no qual o usuário tem duas opções, inserir uma nova tarefa, recuperar uma tarefa na pilha:

- Inserir tarefa na pilha: escreve a descrição da tarefa, e inseri na pilha
- recuperar tarefa: a tarefa é apresentada no terminal para o cliente.

2- Escreva uma função que receba como parâmetro uma lista encadeada contendo apenas valores 0 e 1 e que retorne as posições inicial e final da maior sequência de elementos 0 dentro da lista. Ex: Lista={0,1,1,0,0,0,1,0} posicaoInicial = 4 e posicaoFim = 6 sequencia={0,0,0} Obs: No caso de empate em tamanho, a primeira ocorrência de sequencia é a que deve ser retornada.

3- Desenvolva uma função para testar se duas pilhas sem tamanho definidos P1 e P2 são iguais. Duas pilhas são ditas iguais, se possui todos os elementos na mesma posição.

4- Crie uma função na pilha simplesmente encadeada que remove um item com chave **key** fornecida pelo usuário da pilha. Esta função deve fazer parte da classe StackDinamic, e sua função deve manter as propriedades da pilha, ou seja, a manipulação deve-se dar a partir do Top. Obs.: Após o ítem de chave key removido, a pilha deve permanecer em sua configuração anterior, apenas sem o o ítem de chave key.

5- Criar um código que teste se uma fórmula é bem formada usando estrutura de pilha sem tamanho definido (dinâmica). Uma fórmula bem formada testa se para cada parêntese aberto, existe um parêntese fechado.

- Entrada: vai ser uma string inserida pelo usuário (uma fórmula)
- Saída: veradeiro ou falso
- Exemplos:
    - String entrada: "((9+5) + 8)" Saída: True
    - String entrada: "(10*4)+((9+5)" Saída: false

6- Escreva uma função void MoveMenor que, dada uma fila simplesmente encadeada com um número qualquer de elementos, acha o menor elemento da fila e o mova-o para o início.

7- Altere sua estrutura de fila simplesmente encadeada, para que a estrutura tenha capacidade de lidar com prioridades. Existem três tipos de prioridade para um nó: alta, baixa, média (configure como achar melhor, número, enum). Esta nova fila dinâmica com prioridades, segue as seguintes regras:

- Só existe uma única fila, organizada por prioridade.
- Um nó de prioriodadade maior não pode estar atrás de um nó de prioride menor.

- Ao inserir na fila deve-se organizar a fila a fim de obedecer as regras anteriores.

8- Uma central de atendimento a clientes tem vários atendentes, mas um número muito maior de linhas telefônicas recebendo chamadas. As chamadas são colocadas em uma fila de espera segundo a ordem de chegada (e atendidas quando possível). Ocorre que algumas destas chamadas vêm de longe, e neste caso, se elas ficam esperando na linha, elas ficam causando uma despesa muito maior do que as chamadas que vem de perto. Uma solução alternativa seria colocar as chamadas em fila segundo a prioridade definida primeiramente pelo custo (as mais caras devem esperar menos) e secundariamente por ordem de chamada. Projete uma estrutura para modelar essa situação alternativa.

9- Dadas duas filas encadeadas e dinâmicas L1 e L2, implemente a operação UNION, que cria uma terceira fila L3 com a união entre as duas listas.

10- Dada uma lista encadeada de caracteres formada por uma sequência alternada de letras e dígitos, construa um método que retorne uma lista na qual as letras são mantidas na sequência original e os dígitos são colocados na ordem inversa.

Exemplos:

- A 1 E 5 T 7 W 8 G → A E T W G 8 7 5 1
- 3 C 9 H 4 Q 6 → C H Q 6 4 9 3

11- Construa um método que recebe uma lista encadeada de números inteiros e retorna uma lista sem repetições, ou seja, uma lista onde cada número apareça apenas uma vez.

12- Crie uma função search na lista simplesmente encadeada que busca e remove o elemento da lista. Se o elemento não existir, soltar uma exceção específica criada para elemento buscado e não encontrado. Ao final do exercício, a lista deve manter sua configuração original, sem o elemento buscado.

13- Dadas duas listas encadeadas e dinâmicas L1 e L2, sem elementos repetidos, implemente a operação INTER, que cria uma terceira lista L3 com a intersecção entre as duas listas, também sem elementos repetidos.

14- Crie uma função na estrutura de Lista dinâmica que insere elementos inteiros de forma crescente. Sempre que um novo número é inserido, este deve ser inserido de forma que toda lista esteja ordenada crescente.

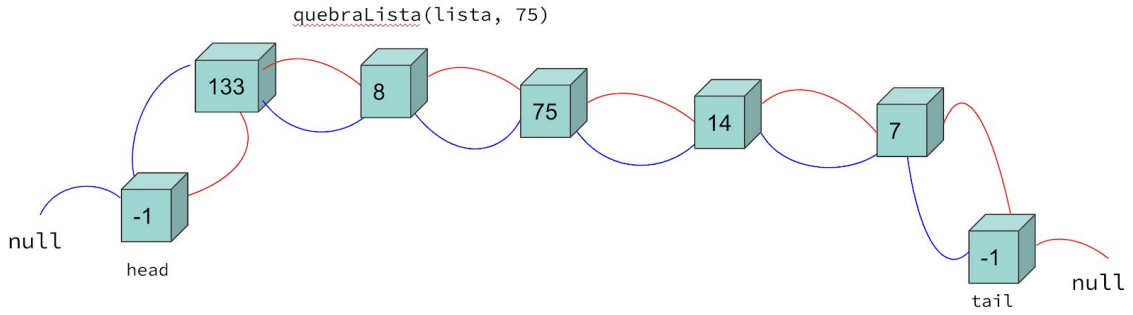
15- Dadas duas listas encadeadas e dinâmicas L1 e L2, implemente a operação UNION, que cria uma terceira lista L3 com a união entre as duas listas.

16- Uma lista duplamente encadeada possui registros que tem ligações com o sucessor e o predecessor de cada nó. Tendo como base a lista apresentada em sala de aula, crie uma nova lista, que seja circular, ou seja, o próximo nó do último elemento aponta para o primeiro elemento da lista. Desta forma, exclua o tail da da sua lista circular (visto que como é uma

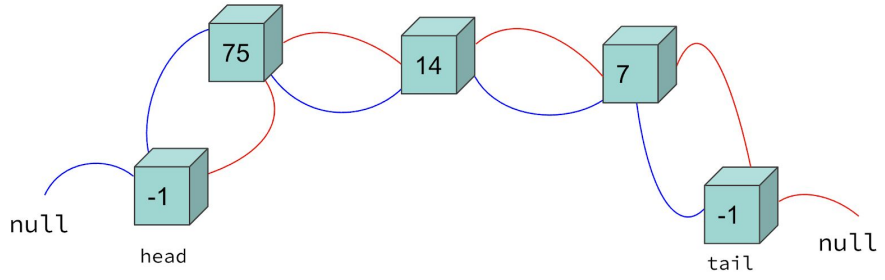
lista circular, sua nova lista não deverá contar com um final), o Head continua existindo, o próximo do head aponta para o primeiro elemento, e o anterior do head aponta para null (comportamento semelhante na lista duplamente encadeada tradicional). Adapte todos os métodos para que o comportamento da lista duplamente encadeada circular funciona adequadamente.

17- Escreva um método para lista duplamente encadeada tradicional (apresentada em sala), que quebre a lista a partir de um valor passado por parâmetro, e retorne uma nova lista a partir deste nó. Veja o exemploa abaixo:

Lista passado por parâmetro:



Lista de retornod a função:



18- Escreva um algoritmo que recebe uma lista duplamente encadeada e inverte essa lista, alterando somente os campos dos ponteiros e sem usar estruturas auxiliares.

19- Implemente uma função na sua lista simplesmente encadeada que busca um determinado elemento e retorne o elemento no retorno da função. Se o elemento não for encontrado, soltar exceção, elemento não encontrado.