MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto
Departamento de Computação e Sistemas
Campus João Monlevade

Curso: Engenharia de Computação — 1º semestre de 2019
Disciplina: Algoritmos e Estruturas de Dados I — Data: 20/06/19
Professor: Alexandre Magno de Sousa

CONJUNTO LISTA DE QUESTÕES DE PROVAS ANTERIORES

1 Questões sobre Listas

1. Escreva uma função que verifica se duas listas simplesmente encadeadas são iguais.
2. Faça uma função que insere elementos em ordem crescente de nomes em uma lista simplesmente encadeada. Para isso, pense nos três casos a seguir: **(a)** quando a lista estiver vazia, utilize a função `Insere` padrão (que sempre insere um elemento no final da lista), construída em sala de aula, para inserir o elemento, veja que se ela está vazia basta inserir o primeiro elemento sem se importar com a ordem da lista; **(b)** se o elemento a ser inserido na lista for maior do que o último elemento existente na lista, utilize a função `Insere` padrão, pois, se ele já é maior do que o último, significa que ele deverá entrar no final da lista; **(c)** este caso somente acontece quando os casos (a) e (b) não acontecem, assim, você deve percorrer a lista comparando o elemento a ser inserido com os que já existem na lista, enquanto o elemento a ser inserido for maior do que o elemento comparado, continue percorrendo a lista, caso ele seja menor ou igual, então você encontrou o ponto de inserção do novo elemento.
3. Faça uma função que receba como parâmetro uma lista simplesmente encadeada e imprima os dados da i-ésima célula dessa lista. Tenha certeza de que a célula existe. Por exemplo, se a lista for $L1 = (A, B, C, D, E)$ e $i = 3$, você deverá realizar uma varredura na lista de modo que avance apenas i vezes e consiga acessar a terceira célula e imprimir o elemento C.
4. Baseado na lógica da questão anterior, faça uma função que receba como parâmetro uma lista simplesmente encadeada e remova a i-ésima célula dessa lista.
5. Crie uma função que receba como parâmetro duas listas simplesmente encadeadas e ordenadas pelo nome e uma terceira lista duplamente encadeada. A função deverá intercalar os elementos das duas listas simplesmente encadeada na lista duplamente encadeada. Por exemplo, sejam as duas listas simplismente encadeadas $L1 = (A, B, C, D)$ e $L2 = (K, L, M)$, intercalando os elementos a lista duplamente encadeada deverá ser $D = (A, K, B, L, C, M, D)$.
6. Faça uma função que remova da lista duplamente encadeada LD as células que ocupam as posições indicadas em uma lista simplesmente encadeada de inteiros L. Por exemplo, se $LD = (A, B, C, D, E)$ e $L = (2, 4, 8)$, então, depois da remoção $L = (A, C, E)$. Veja que o terceiro elemento da lista L é 8, porém, a lista LD possui apenas 5 elementos, nesse caso, não há como remover a 8^a célula porque ela não existe.

2 Questões sobre Filas

1. Faça uma função que receba como parâmetro uma fila simplesmente encadeada e um parâmetro n e remova o n-ésimo item dessa fila.
2. Faça uma função que receba como parâmetro três filas simplesmente encadeadas e que realiza a interseção dos elementos das duas primeiras filas na terceira.
3. Construa uma função que desloca uma determinada célula p n posições à frente em uma fila duplamente encadeada.
4. Coloque em ordem crescente os elementos de uma fila utilizando obrigatoriamente duas filas adicionais.
5. Faça uma função que receba como parâmetro uma fila de inteiros e retorne a mediana dessa lista de números, para isso, coloque a fila em ordem crescente.
6. Faça uma função que receba como parâmetro três filas e faça com que o resultado da diferença (operação de diferença entre conjuntos) dos elementos da primeira com a segunda seja colocado na terceira fila.

3 Questões sobre Pilhas

1. Crie uma função que inverta a ordem dos elementos em uma pilha: (a) usando duas pilhas auxiliares; (b) usando uma fila auxiliar; (c) usando uma pilha auxiliar e variáveis auxiliares.

2. Faça uma função que coloque os elementos de uma pilha passada via parâmetro em ordem decrescente utilizando uma pilha auxiliar e algumas variáveis auxiliares.

3. Crie uma função que transfere os elementos da primeira pilha para a segunda sem utilizar uma pilha auxiliar.

4. Faça uma função que receba como parâmetro uma pilha e organize seus elementos da seguinte forma: o primeiro elemento deve ser o menor, o segundo deverá ser o maior, o terceiro deve ser o segundo menor e o quarto deverá ser o segundo maior, e assim sucessivamente. Utilize apenas uma pilha auxiliar e variáveis auxiliares.

5. Descreva como seria implementar o TAD Fila utilizando duas pilhas, para isso, construa o TAD Fila a partir do TAD Pilha e crie as funções de enfileirar e desenfileirar.

6. (*8 pontos*) Um exemplo de utilização de pilha consiste em avaliar uma expressão aritmética segundo a utilização dos parênteses ( ), colchetes [ ] e chaves { }. O problema consiste em criar uma pilha de caracteres. O programa recebe uma sequência de caracteres que representa uma expressão aritmética genérica, por exemplo: $\{A * (A + B)\}$. A expressão é lida caractere a caractere da esquerda para direita. Quando um caractere de abertura é encontrado (, [, ou { ele é empilhado. Quando um caractere de fechamento é encontrado),] ou }, o elemento do topo da pilha é comparado com ele. Se o elemento do topo da pilha representa a abertura do respectivo fechamento, então, o elemento é desempilhado. Quando o final da expressão é alcançado, se a pilha está vazia, então a expressão está correta, caso contrário, a expressão está incorreta. Crie uma função que implementa este problema. Esta função deverá receber como parâmetro um vetor de caracteres, que contém a expressão e uma pilha. Considere que o TAD e as operações da pilha já existem.

4 Questões Avançadas

1. Crie uma função para enfileirar e desenfileirar um elemento em um *TAD Fila de Prioriedade* de acordo com sua prioridade que é dada no campo prioridade da estrutura. As funções deverão ter o seguinte cabeçalho

```
void EnQueue(FilaPrioridade *fila, TipoItem x)

void DeQueue(FilaPrioridade *fila, TipoItem x)
```

   A função irá trabalhar em um contexto em que existem apenas três prioridades: alta (3), média (2) e baixa (1). Por exemplo, sejam dados a seguir os seguintes elementos e suas respectivas prioridades:

Elemento	Prioridade
A	1
B	3
C	3
D	2

   A inserção desses elementos em uma fila de prioridades resultaria em uma fila

$$F = \{B, C, D, A\}$$

   Encontre a estratégia mais eficiente possível para construir esse *TAD de Filas de Prioridade*.

2. Um dos exemplos de aplicação de pilhas é para adição de números muito grandes. Números inteiros são limitados e assim não é possível somar 18×10^{30} com 8×10^{15}. O problema pode ser resolvido se tratarmos esses números como cadeias de numerais, pode-se armazenar os números correspondentes

a esses numerais em duas pilhas e então realizar a adição extraindo-se os números das pilhas. A Figura 1 apresenta um exemplo desse tipo de soma. Nesse exemplo, os números 592 e 3784 são somados da seguinte forma:

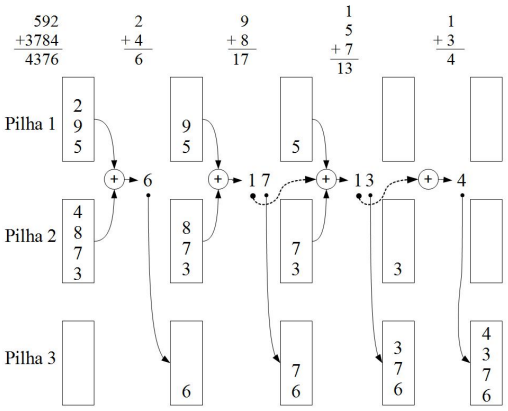

Figura 1: Um exemplo para somar os números 592 e 3784 usando pilhas.

1. Os números que correspondem aos dígitos que compõem o primeiro número são colocados na Pilha 1, e os que correspondem aos dígitos do segundo número são colocados na Pilha 2, observe a ordem dos dígitos na pilha.
2. Os números 2 e 4 são extraídos das pilhas e o resultado da soma, 6, é colocado na Pilha 3.
3. Os números 9 e 8 são extraídos das pilhas e a parte unitária da soma, 7, é colocada na Pilha 3, a parte da dezena do resultado, o número 1, é retida como vai-um na variável `resultado` para uma soma posterior.
4. Os números 5 e 7 são extraídos das pilhas e somados ao vai-um, a parte unitária do resultado, 3, é colocada na Pilha 3 e o vai-um, 1, se torna um valor da variável `resultado`.
5. Uma pilha está vazia, assim, um número é extraído da pilha não vazia e somado ao vai-um. O resultado é colocado na Pilha 3.
6. Ambas as pilhas de operando estão vazias, assim, os números da Pilha 3 são extraídos e impressos como resultado final.

3. Torre de Hanói é um jogo que consiste em uma base contendo três pinos, dos quais em um deles estão dispostos três discos, uns sobre os outros, em ordem crescente de diâmetro de cima para baixo. O problema consiste em passar todos os discos de um pino para outro utilizando um dos pinos como auxiliar, de maneira que um disco maior nunca fique em cima de outro menor em nenhuma situação. O número de discos pode variar. É possível simular a torre de Hanói de forma que os pinos sejam representados por um TAD pilha e os discos por números inteiros. Em uma das pilhas os números estão dispostos de forma que o menor esteja no topo e, de maneira crescente de cima para baixo, o maior esteja na base. Faça uma função que receba com o parâmetro três pilhas de forma que na primeira pilha estão dispostos três discos, transfira os discos para a terceira pilha obedecendo as regras da Torre de Hanói. DICA: simule a Torre de Hanói por meio de desenhos antes de escrever o código.