Pontifícia Universidade Católica de Minas Gerais
Curso de Ciência da Computação
Disciplina: Algoritmos e Estruturas de Dados II

Lista de Exercícios

Tipos Abstratos de Dados Lineares

Lista

1. Faça a implementação de uma lista linear em Java para armazenar números inteiros. Para isso, crie uma classe chamada ListaLinear. Sua classe deve possuir os métodos abaixo e tratar os casos de *overflow* e *underflow* nas operações de inserção e remoção.

   - public ListaLinear(int max): cria uma lista capaz de armazenar, no máximo, max elementos.
   - public void inserirInicio(int elem): insere o elemento no início da lista
   - public void inserirFim(int elem): insere o elemento no fim da lista
   - public void inserir(int elem, int pos): insere o elemento na posição pos da lista
   - public int removerInicio(): remove e retorna o elemento do início da lista
   - public int removerFim(): remove e retorna o elemento do fim da lista
   - public int remover(int elem, int pos): remove e retorna o elemento na posição pos da lista
   - public void mostrar(): imprime os elementos da lista, método iterativo
   - public boolean pesquisar(int elem): pesquisa se o elemento está presente na lista, retornando verdadeiro em caso afirmativo

   Teste sua classe, criando um programa que instancia uma lista e invoca os métodos implementados.

2. Faça a implementação de uma lista linear em C para armazenar números inteiros. Para isso, crie uma estrutura (*struct*) chamada ListaLinear e um conjunto de funções descritos a seguir. Trate os casos de *overflow* e *underflow* nas operações de inserção e remoção.

   - ListaLinear* newListaLinear(int max): aloca dinamicamente uma lista capaz de armazenar, no máximo, max elementos. Retorna o ponteiro para a lista
   - void delListaLinear(ListaLinear* lista): desaloca a lista da memória

- void inserirInicio(ListaLinear* lista, int elem): insere o elemento no início da lista
- void inserirFim(ListaLinear* lista, int elem): insere o elemento no fim da lista
- void inserir(ListaLinear* lista, int elem, int pos): insere o elemento na posição pos da lista
- int removerInicio(ListaLinear* lista): remove e retorna o elemento do início da lista
- int removerFim(ListaLinear* lista): remove e retorna o elemento do fim da lista
- int remover(ListaLinear* lista, int elem, int pos): remove e retorna o elemento na posição pos da lista
- void mostrar(ListaLinear* lista): imprime os elementos da lista, método iterativo
- int pesquisar(ListaLinear* lista, int elem): pesquisa se o elemento está presente na lista, retornando verdadeiro em caso afirmativo

Teste sua estrutura, criando um programa que instancia uma lista e invoca os métodos implementados.

3. Na implementação de lista linear fornecida em nosso material, você pode constatar que os métodos para inserir no início e inserir no meio da lista são muito parecidos. Proponha uma alteração na codificação dessas operações a fim de favorecer a reutilização de código.

4. Crie um método na classe ListaLinear que inverte a ordem dos elementos da lista. Implemente uma versão (a) iterativa e (b) uma recursiva.

5. Crie uma estrutura de dados chamada ListaLinearOrdenada que representa uma lista linear ordenada, isto é, os elementos são armazenados na lista em ordem crescente. As seguintes operações devem estar disponíveis para a lista:

   - Inserção: insere um elemento na lista em ordem
   - Remoções no início, no meio e no fim: remove e retorna o elemento
   - Mostrar: imprime a lista
   - Verificar ordenação: verifica se os elementos da lista estão ordenados de forma crescente - chame o método de isOrdenada(), retornando, verdadeiro em caso afirmativo.
   - Pesquisar: pesquisa por um elemento na lista de forma eficiente, retornando verdadeiro caso o elemento seja encontrado

Pilha

6. Faça a implementação de uma pilha linear em Java para armazenar números inteiros. Para isso, crie uma classe chamada PilhaLinear. Sua classe deve possuir os métodos abaixo e tratar os casos de *overflow* e *underflow* nas operações de inserção e remoção.

   - public PilhaLinear(int max): cria uma pilha capaz de armazenar, no máximo, max elementos.
   - public void empilhar(int elem): insere o elemento na pilha
   - public int desempilhar(): remove e retorna o elemento do topo da pilha
   - public void mostrar(): imprime os elementos da pilha, na ordem que serão removidos, método iterativo
   - public boolean pesquisar(int elem): pesquisa se o elemento está presente na pilha, retornando verdadeiro em caso afirmativo

   Teste sua classe, criando um programa que instancia uma pilha e invoca os métodos implementados.

7. Faça a implementação de uma pilha linear em C para armazenar números inteiros. Para isso, crie uma estrutura (*struct*) chamada PilhaLinear e um conjunto de funções descritos a seguir. Trate os casos de *overflow* e *underflow* nas operações de inserção e remoção.

   - PilhaLinear* newPilhaLinear(int max): aloca dinamicamente uma pilha capaz de armazenar, no máximo, max elementos. Retorna o ponteiro para a pilha
   - void delPilhaLinear(PilhaLinear* pilha): desaloca a pilha da memória
   - void empilhar(PilhaLinear* pilha, int elem): insere o elemento na pilha
   - int desempilhar(PilhaLinear* pilha): remove e retorna o elemento do topo da pilha
   - void mostrar(PilhaLinear* pilha): imprime os elementos da pilha, na ordem que serão removidos, método iterativo
   - int pesquisar(PilhaLinear* pilha, int elem): pesquisa se o elemento está presente na pilha, retornando verdadeiro em caso afirmativo

   Teste sua estrutura, criando um programa que instancia uma pilha e invoca os métodos implementados.

8. Crie um método na classe PilhaLinear que converte a pilha para uma lista linear, sendo o primeiro elemento da lista o elemento do tipo e assim sucessivamente. O método não altera a pilha e gera uma lista correspondente. Utilize o seguinte protótipo:

   ListaLinear toLista()

9. Crie um método recursivo na classe PilhaLinear que mostra os elementos da pilha na ordem que foram inseridos.

Fila

10. Faça a implementação de uma fila linear em Java para armazenar números inteiros. Para isso, crie uma classe chamada FilaLinear. Sua classe deve possuir os métodos abaixo e tratar os casos de *overflow* e *underflow* nas operações de inserção e remoção.

    - public FilaLinear(int max): cria uma fila capaz de armazenar, no máximo, max elementos.
    - public void enfileirar(int elem): insere o elemento na fila
    - public int desenfileirar(): remove e retorna o elemento da fila
    - public void mostrar(): imprime os elementos da fila, na ordem que foram inseridos, método iterativo
    - public boolean pesquisar(int elem): pesquisa se o elemento está presente na fila, retornando verdadeiro em caso afirmativo

    Teste sua classe, criando um programa que instancia uma fila e invoca os métodos implementados.

11. Faça a implementação de uma fila linear em C para armazenar números inteiros. Para isso, crie uma estrutura (*struct*) chamada FilaLinear e um conjunto de funções descritos a seguir. Trate os casos de *overflow* e *underflow* nas operações de inserção e remoção.

    - FilaLinear* newFilaLinear(int max): aloca dinamicamente uma fila capaz de armazenar, no máximo, max elementos. Retorna o ponteiro para a fila
    - void delFilaLinear(FilaLinear* fila): desaloca a fila da memória
    - void enfileirar(FilaLinear* fila, int elem): insere o elemento na fila
    - int desenfileirar(FilaLinear* fila): remove e retorna o elemento da fila
    - void mostrar(FilaLinear* fila): imprime os elementos da fila, na ordem que foram inseridos, método iterativo
    - int pesquisar(FilaLinear* fila, int elem): pesquisa se o elemento está presente na fila, retornando verdadeiro em caso afirmativo

    Teste sua estrutura, criando um programa que instancia uma fila e invoca os métodos implementados.

12. Descreva como inverter os elementos de uma fila utilizando uma pilha. Crie um programa para mostrar sua solução.

13. Crie um método recursivo na classe FilaLinear que mostra os elementos da fila na ordem que serão removidos.

Fila Circular

14. Faça a implementação de uma fila circular em Java para armazenar números inteiros. Para isso, crie uma classe chamada FilaCircular. Sua classe deve possuir os métodos abaixo e tratar os casos de *overflow* e *underflow* nas operações de inserção e remoção.

    - public FilaCircular(int max): cria uma fila capaz de armazenar, no máximo, max elementos.
    - public void enfileirar(int elem): insere o elemento na fila
    - public int desenfileirar(): remove e retorna o elemento da fila
    - public void mostrar(): imprime os elementos da fila, na ordem que foram inseridos, método iterativo
    - public boolean pesquisar(int elem): pesquisa se o elemento está presente na fila, retornando verdadeiro em caso afirmativo

    Teste sua classe, criando um programa que instancia uma fila circular e invoca os métodos implementados.

15. Faça a implementação de uma fila circular em C para armazenar números inteiros. Para isso, crie uma estrutura (*struct*) chamada FilaCircular e um conjunto de funções descritos a seguir. Trate os casos de *overflow* e *underflow* nas operações de inserção e remoção.

    - FilaCircular* newFilaCircular(int max): aloca dinamicamente uma fila capaz de armazenar, no máximo, max elementos. Retorna o ponteiro para a fila
    - void delFilaCircular(FilaCircular* fila): desaloca a fila da memória
    - void enfileirar(FilaCircular* fila, int elem): insere o elemento na fila
    - int desenfileirar(FilaCircular* fila): remove e retorna o elemento da fila
    - void mostrar(FilaCircular* fila): imprime os elementos da fila, na ordem que foram inseridos, método iterativo
    - int pesquisar(FilaCircular* fila, int elem): pesquisa se o elemento está presente na fila, retornando verdadeiro em caso afirmativo

    Teste sua estrutura, criando um programa que instancia uma fila e invoca os métodos implementados.

16. Crie um método isVazia() que retorna verdadeiro se a fila circular está vazia. Crie uma método isCheia() que retoena verdadeiro se a fila circular está cheia.

17. Crie um método recursivo para mostrar os elementos da fila circular na ordem que foram inseridos.

18. Implemente a fila circular onde o arranjo conterá uma quantidade de elementos igual à solicitada na construção da fila. Isso equivale a construir a fila sem o +1 no arranjo conforme apresentado no material.