UENF

Universidade Estadual do Norte Fluminense Darcy Ribeiro

Curso: Ciência de Computação **Data:** 29./.05./2023
Prova: Teste1 **Período:** 3º **Disciplina:** Estrutura de dados I
Professor: Fermín Alfredo Tang **Turno:** Diurno
Nome do Aluno: Gabriel Viana de Almeida.Matrícula: 20211000143

1. **[1,5 Pontos]** Explique o conceito de Tipo Abstrato de Dados (TAD). Qual a vantagem de definir uma estrutura de dados como TAD? Porque motivo a definição do TAD é realizada em arquivos diferentes do programa de aplicação?.

Tipo Abstrato de dados (TAD) é um conjunto de valores cujo seu comportamento é definido por operações implementadas como funções. Ele é constituído a partir de tipos básicos (int, char, float, double) ou estruturados (array, struct, enum, union), são entidades teóricas, usadas para simplificar algoritmos abstratos, avaliar e classificar as estruturas de dados e descrever certos programas

Possui diversas vantagens tais como:

- Abstração e encapsulamento: o usuário só precisa se preocupar com a interface definida pelo TAD, sem precisar saber os detalhes da implementação interna, isso torna mais fácil sua utilização.
- Segurança: Não permite acesso direto do usuário ao código de implementação, evitando possíveis erros causados pelo usuário
- Reutilização: O TAD é feito em arquivos diferentes, com isso, pode ser utilizado em outros projetos, economizando tempo e esforço.
- Flexibilidade: Permite alterar o TAD sem mexer nas aplicações que o utilizam.

Como dito anteriormente, é aplicado em outro arquivo justamente pelas características de reutilização e flexibilidade. Permitindo utilizá-lo para outros projetos, além de deixar o código mais organizado e limpo.

2. **[1,5 Pontos]** Descreva as características de uma Pilha. Quais são as principais operações que operam sobre a estrutura?. Como ela funciona? Ilustre com um exemplo.

Uma pilha é definida pelo conceito de LIFO (Last-in-first-out), ou seja, primeira a entrar, primeiro a sair.

As principais operações são:

- Criar pilha (CreateStack)
- Adicionar elemento ao topo da pilha (PushStack)
- Remover elemento ao topo da pilha (PopStack)

- Destruir pilha (DestroyStack)
- Acessar elemento do topo (PeekStack)
- Verificar se está vazia ou cheia (EmptyStack | FullStack)
- Verificar o tamanho (StackCount)

  O **CreateStack** é responsável por criar a pilha, e o **DestroyStack** deleta a pilha. Sempre é adicionado ao topo da pilha com a função **PushStack,** e o **PopStack** remove sempre o elemento do topo. É possível acessar o elemento do topo com o **PeekStack**, além de verificar se está cheia ou vazia e seu tamanho.

  Vamos simular um exemplo como uma pilha de pratos.

[] ->A pilha é criada com

[A] -> É adicionado 1 prato ao topo da pilha

[C B A] -> São adicionados mais dois pratos no topo da pilha

[B A] -> É removido o primeiro prato da pilha.

3. **[1,5 Pontos]** Descreva as características de uma Fila. Quais são as principais operações que operam sobre a estrutura?. Como ela funciona? Ilustre com um exemplo.

   É uma sequência de elementos do mesmo tipo que seguem a ordem FIFO(First-in-First-out) - primeiro a entrar, primeiro a sair.

   As principais operações:

   - Criação da fila
   - Destruição da fila
   - Inserção no final da fila
   - Remoção do elemento no início
   - Acesso do primeiro elemento
   - Verificar se está vazio ou cheia
   - Informações do tamanho

Ela é criada e é inserida um dado sempre ao fim da fila. O primeiro elemento que entrou será o primeiro a sair. É possível verificar se a fila está cheia ou vazia, e quantos elementos possui, além de ser possível visualizar o primeiro elemento da fila. E por fim, há a opção de destruí-la..

Vamos simular um exemplo de fila numa fila de cinema.

[ ] -> A fila é criada

[E D C B A] -> Uma fila com 5 pessoas aguardando a chamada para comprar o ingresso.

[E D C B] -> O primeiro da fila (A) é chamado para comprar, dito isso, restam apenas 4 pessoas na fila

[E D C] -> Agora o B é chamado e deixa a fila.

[P Q E D C] -> Duas pessoas entram no final da fila.

4. **[1,5 Pontos]** Descreva as características de uma Lista. Quais são as principais operações que operam sobre a estrutura? Como ela funciona? Ilustre com um exemplo.

   Uma lista é uma sequência de elementos do mesmo tipo, é utilizada para armazenar e organizar dados, comumente esses dados são ordenados. Diferente de fila e pilha, a lista permite acessar e adicionar elementos em qualquer posição.

   Sua principais operações são:

   - Criação da lista
   - Inserção de um elemento na lista
   - Remoção de um elemento na lista
   - Destruição da lista
   - Recuperar elemento
   - Verificar se está vazia ou cheia
   - Verificar o tamanho

Primeiro, cria-se a lista, e posteriormente adiciona os elementos que serão armazenados sendo eles podendo ser adicionados ao início, fim ou meio. É possível qualquer elemento, e também recuperar um elemento específico. Além do fato de verificar se está cheia ou vazio e sua quantidade.

Um exemplo de lista:

Vamos supor uma lista de compras com 3 elementos.

[P Q R] -> Lista com 3 elementos.

[P Q T R] -> Você adiciona o T no meio da lista.

[P T R] -> Agora é removido o Q da lista.

[A P T R B] -> Agora foram adicionados **A** no início e **B** ao final da lista respectivamente.

5.- **[2,0 Pontos]** As estruturas de Pilhas, Filas e Listas podem ser implementadas usando estruturas estáticas (vetores) ou estruturas dinâmicas (listas encadeadas). Cite as vantagens e desvantagens, de ambas as implementações em cada caso.

- Pilhas estáticas
  - Vantagens:
    - Facilidade na criação e destruição da pilha: Uma vez que a implementação com vetores é simples, a criação e destruição são simples de aplicar.
  - Desvantagens:
    - Capacidade limitada: Pois é necessário pre definir um vetor de tamanho X .
    - Desperdício de memória: Caso seja criado um vetor com grande capacidade que não seja utilizada totalmente, gera um desperdício de memória.
    - Stack Overflow caso a pilha esteja cheia, não terá espaço para novos elementos causando possíveis erros.
- Pilhas dinâmicas
  - Vantagens:
    - Tamanho flexível: Não possuem tamanho fixo, é acrescentada conforme vai adicionando ou removendo elementos.
    - Melhor utilização de recursos de memória: Aloca apenas memória necessária no momento, evitando perdas
  - Desvantagens
    - Percorrer a pilha inteira para destruí-la: Precisa remover nó por nó para destruir a fila.
    - Complexidade para criar comparada com a estática: É mais complicada de implementar do que a estática.
- Filas estáticas
  - Vantagens:
    - Facilidade na criação e destruição de pilha: Sua implementação, assim como a pilha, é bem simples.
  - Desvantagens:
    - Capacidade limitada: Precisa pré definir o tamanho do vetor
    - Desperdício de memória: Se o vetor tiver capacidade além do necessário, estará desperdiçando memória.
- Filas dinâmicas
  - Vantagens
    - Tamanho flexível: Não possuem tamanho fixo, é acrescentada conforme vai adicionando ou removendo elementos.
    - Melhor utilização de recursos de memória: Aloca apenas memória necessária no momento, evitando desperdício.
  - Desvantagens
    - Percorrer toda a fila para destruí-la: Precisa percorrer por cada nó da fila para destruí-la completamente.

        - Complexidade maior para implementação: A necessidade de realizar os encadeamento dos nós a torna mais complexa de implementar.
- Listas estáticas
    - Vantagens
        - Acesso rápido e direto aos elementos: Pode alcançar qualquer elemento pelo índice.
        - Tempo constante para acessar elementos: Como tem acesso direto aos elementos, não precisa percorrer toda a lista para achá-lo.
        - Facilidade de modificar suas informações: Basta acessar o elemento na sua posição e mudá-lo.
    - Desvantagens
        - Tamanho limitado: Precisa pré definir o tamanho da lista antes.
        - Dificuldade de inserir e remover elementos entre outros dois: É necessário mover os elementos para preencher o espaço vazio da remoção, isso deixa o código menos eficiente.
- Lista dinâmica
    - Vantagens
        - Tamanho variável: Não precisa pré definir o tamanho como a na lista estática
        - Melhor utilização de memória: Aloca apenas memória necessária no momento, evitando desperdício.
        - Facilidade de inserção e remoção: Não precisa deslocar os elementos ao remover ou inserir um novo.
    - Desvantagens
        - Acesso indireto aos elementos: Diferente da lista estática, precisa percorrer toda a lista a fim de encontrar o elemento desejado.
        - Complexidade de implementação: É necessário alocar memória para cada nó da lista e manipular os ponteiros de ligação.

6.- **[1,0 Pontos]** As listas encadeadas podem ser implementadas com estruturas cabeçalhos ou sem estruturas cabeçalhos. Cite as vantagens e desvantagens em cada caso.

Listas com cabeçalho:

- Vantagens
    - Contagem para quantidade de elemento
    - Ponteiro para início e final do elemento
- Desvantagens
    - Uso de memória adicional

Listas sem cabeçalho

- Vantagens
    - Menor gasto de memória
- Desvantagens
    - Precisa percorrer toda a lista pra chegar no último ao final
    - Sem contador de elementos da lista

7.- **[1,0 Pontos]** Explique as semelhanças e diferenças entre uma lista duplamente encadeada e uma lista multi-encadeada.

Semelhanças:

- Estruturas de nós que contêm dados e referências de ponteiros anteriores e posteriores.
- Permitem percorrer os elementos da lista

Diferenças

- Na duplamente encadeada há referência para o nó anterior e posterior, já na multi-encadeada cada nó pode conter referências para vários outros nós.
- Uma lista multi-encadeada é mais complexa e trabalhosa de implementar do que a duplamente encadeada, tendo em vista que ela pode possuir múltiplas conexões.
- A lista multiencadeada não é linear, ela permite ter conexões diferentes, formando estruturas de grafos, árvores e etc. Já a lista duplamente encadeada, tem apenas acesso linear dos dados.