Listas

1/2024
kenia.carolina@ifmg.edu.br

Programação III

Kênia Carolina

Listas

- Forma mais simples de interligar elementos de um conjunto

- Tipo Estruturado de Dados(TAD):
  - Estática
  - Dinâmica

- Operações: inserir, retirar e localizar

Listas: Quando usar?

- Aplicações nas quais não é possível prever a demanda por memória

- E os dados possuem uma organização lógica para estarem no mesmo conjunto(lista)

Listas Estáticas

- **Listas com Vetores: Desvantagens**
  - Tamanho máximo fixo
  - Mesmo vazias ocupam espaço grande de memória •
    Mesmo que utilizemos um vetor de ponteiros, se quisermos prever uma lista de 10.000 elementos, teremos 40.000 bytes desperdiçados.
  - Operações podem envolver muitos deslocamentos de dados:
    - Inclusão em uma posição ou no início
    - Exclusão em uma posição ou no início

Lista Dinâmicas

- Alocação em tempo de execução
- C++: ponteiros, new e delete
- C: ponteiros, malloc e free

Listas Encadeadas

- Simplesmente
  - Linear
  - Circular
- Duplamente
  - Linear
  - Circular

Listas Encadeadas

- **Simplesmente**
  - **Linear**
  - Circular
- Duplamente
  - Linear
  - Circular

Operações

1. Criar uma lista vazia
2. Pesquisar um item na lista a partir do valor chave
3. Inserir o item x: ao final, início ou ordenado 4.

Retirar um item da lista a partir do valor chave, o primeiro, o último ou no meio

5. Retornar uma referência do primeiro, último ou do item a partir de sua chave
6. Verificar se a lista está vazia
7. Imprimir os itens da lista

Item/Célula/Elemento/...

```
class Celula{

    int valor; //variável
    Celula prox; // referência
};
```

*Isso vai para o arquivo Celula.java

Lista Simplesmente Encadeada Linear

```
Celula primeiro;
Lista();
boolean vazia();
void inserirAoFinal(Celula c);
void inserirNoInicio(Celula c);
Celula pesquisar(int v);
boolean removerInicio();
boolean removerFinal();
void imprimir();
```

*Isso vai para o arquivo ListaSE.java

Lista Simplesmente Encadeada Linear

```
ListaSE(){ //Construtor
    primeiro = null;
}
```

primeiro

```
public boolean vazia(){
    return (primeiro == null);
}
```

*Isso vai para ListaSE.java

Lista Simplesmente Encadeada

Linear void inserirNoInicio(Celula c){

```
c.prox = primeiro;
primeiro = c;
}
```

*Isso vai para ListaSE.java

12

primeiro

c

Lista Simplesmente Encadeada

Linear

```
void inserirAoFinal(Celula c){
  if(vazia()){
  primeiro = c; }
  else{
    Celula aux =
    primeiro;
```

primeiro

```
    while(aux.prox != NULL) { aux = aux.prox; }
```

c

```
    aux.prox = c;
  }//fim do else
}
```

*Isso vai para ListaSE.java

Lista Simplesmente Encadeada Linear

```
Celula pesquisar(int v){
   if(vazia())
      return null;
   else{
      Celula aux = primeiro;
      while(aux != null && aux.valor != v)
      {
         aux = aux.prox;
      }
      return aux;
```

```
    }
}
```
*Isso vai para ListaSE.java

Lista Simplesmente Encadeada Linear

```
bool removerInicio(){
    if(vazia()){ return false; }
    else{
        Celula aux = primeiro;
        if(aux.prox == null){ primeiro = null; }
        else{
            primeiro = aux.prox;
            aux.prox = null;
        }
        //delete aux; - Não faço isso em linguagens que possuem garbage collection
        return true;
```

```
    }
}
```

*Isso vai para ListaSE.java

Lista Simplesmente Encadeada Linear

```
bool removerFinal(){
   if(vazia()){ return false; }
   else{
      Celula ant = primeiro;
      Celula atual = primeiro.prox;
      if(atual != NULL){
         while(atual.prox != NULL){
            ant = atual;
            atual = atual.prox;
         }
         ant.prox = NULL;
      }
      else{
         atual = primeiro;
         primeiro = NULL;
```

```
      }
      //delete atual; - Não preciso disso em linguagem com garbage collection
      return true;
   }
}*Isso vai para ListaSE.java
```


Lista Simplesmente Encadeada Linear

```
void imprimir()
{
   Celula aux = primeiro;
   while(aux != null)
   {
      SOP("Valor = "+aux.valor);
      aux = aux.prox;
   }
}
```

*Isso vai para ListaSE.java

Lista Simplesmente Encadeada Linear

```
public static void main(String[] args)
{
        ListaSE numeros = new ListaSE();
        Celula n1 = new Celula(10);
        numeros.inserirNoInicio(n1);
        n1 = new Celula(5);
        numeros.inserirAoFinal(n1);
        n1 = new Celula(15);
        numeros.inserirNoInicio(n1);
                numeros.imprimir();
}
```

*Isso vai para Principal.java

10 5

15 10 5

Exercício

- Implemente e teste tudo que foi apresentado na aula de hoje.
- Implemente a inserção ordenada das Células inteiras. Sua lista deverá iniciar vazia e ir inserindo aos poucos. Imprima a lista no final para verificar.
- Implemente a remoção no fim usando apenas uma referência auxiliar.

- Implemente a remoção específica de um elemento. Procure por uma célula com valor x antes de removê la.