# 

**Curso:** Ciência de Computação **Período: 3**º **Turno:** Diurno

**Disciplina:** Estrutura de Dados **Professor:** Fermín Alfredo Tang Montané

**Atividade:** Avaliação AM1 **Data:** 25/05/2021

**Aluno:** João Vítor Fernandes Dias **Matrícula:** 00119110377

1. **[1,5 Pontos]** ilustre com um exemplo (diferente daquele usado em aula) a ideia do algoritmo que permite converter uma expressão infixa em posfixa usando uma pilha.

R:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | | | | |  |  | **2** | | | | |  |
|  | INFIXA |  | PILHA |  | POSFIXA |  |  | INFIXA |  | PILHA |  | POSFIXA |  |
|  | A+B\*C-D/E |  |  |  |  |  |  | B\*C-D/E |  |  |  | A |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | + |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **3** | | | | |  |  | **3** | | | | |  |
|  | INFIXA |  | PILHA |  | POSFIXA |  |  | INFIXA |  | PILHA |  | POSFIXA |  |
|  | C-D/E |  |  |  | AB |  |  | .-D/E |  |  |  | ABC |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | \* |  |  |  |  |  |  | \* |  |  |  |
|  |  |  | + |  |  |  |  |  |  | + |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **5** | | | | |  |  | **6** | | | | |  |
|  | INFIXA |  | PILHA |  | POSFIXA |  |  | INFIXA |  | PILHA |  | POSFIXA |  |
|  | /E |  |  |  | ABC\*+D |  |  | E |  |  |  | ABC\*+D |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | / |  |  |  |
|  |  |  | - |  |  |  |  |  |  | - |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **7** | | | | |  |  | **8** | | | | |  |
|  | INFIXA |  | PILHA |  | POSFIXA |  |  | INFIXA |  | PILHA |  | POSFIXA |  |
|  |  |  |  |  | ABC\*+DE |  |  | A+B\*C-D/E |  |  |  | ABC\*+DE/- |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | / |  |  |  |  |  |  |  |  |  |  |
|  |  |  | - |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **[2,5 Pontos]** duas pilhas são iguais se possuem os mesmos elementos, exatamente na mesma ordem. Considere que as pilhas P1 e P2 já existem e são passadas como parâmetro. Considere que as pilhas possuem elementos do tipo inteiro. Responda os seguintes pontos:
2. **[0,8 Ponto]** descreva em palavras a ideia de um algoritmo para testar se duas pilhas P1 e P2 são iguais, usando operações do TAD Pilha, e preservando as pilhas originais;

R:

Para testarmos se duas pilhas (A e B) são iguais, primeiro iremos assumir que eles são iguais criando uma variável “SaoIguais” (que definirá ao final o estado da pilha, podendo ele ser 0 – Diferente, ou 1 – Igual) e definiremos seu valor inicial como 1. Também iremos criar uma variável temporária do tipo inteiro chamada “Temp” (que guardará temporariamente os elementos que estiverem sendo movidos) e uma pilha auxiliar chamada “Aux” (que armazenará temporariamente os elementos iguais em ambas as pilhas);

Começaremos comparando os contadores de cada pilha. Se os contadores forem diferentes, definiremos o valor de SaoIguais como 0, pois elas não são iguais.

Enquanto as duas pilhas não estiverem vazias e elas não estejam diferentes até então, repete-se o seguinte passo:

Compararemos o topo da pilha de ambos, se forem diferentes, a variável SaoIguais obtém o valor 0, se não forem diferentes, remove-se o elemento do topo de cada pilha e insere esse valor em uma pilha auxiliar;

Terminada essa repetição, outra repetição começa:

Enquanto a pilha auxiliar não estiver vazia, remove-se o elemento de seu topo e o insere tanto na pilha A quanto na pilha B.

Por fim, analisaremos o valor de “SaoIguais”. Se “SaoIguais” for igual a 1 será impresso na tela a mensagem “Sao iguais”, se “SaoIguais” não for igual a 1, será impresso na tela a mensagem “Sao diferentes”.

1. **[1,0 Ponto]** escreva o algoritmo correspondente;

R:

void SaoIguais(\*Pilha A, \*Pilha B){

int Temp;

int SaoIguais=1;

Pilha\* Aux;

CriaPilha(Aux);

se (ContadorPilha(A) != ContadorPilha(B)){

SaoIguais=0;

}

enquanto (!(PilhaVazia(A)||PilhaVazia(B))&&SaoIguais){

se (TopoPilha(A)!=TopoPilha(B)) {

SaoIguais=0;

} senao {

Temp=RemovePilha(A);

Temp=RemovePilha(B);

InserirPilha(Aux,Temp);

}

}

enquanto (!PilhaVazia(Aux)){

Temp = RemovePilha(Aux);

InserirPilha(A, Temp);

InserirPilha(B, Temp);

}

se (SaoIguais) {

imprime("Sao Iguais\n");

} senao {

imprime("Sao diferentes\n");

}

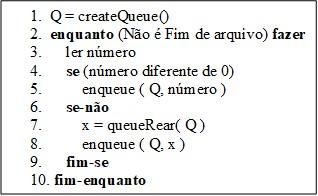
}

1. **[0,7 Ponto]** ilustre seu algoritmo com um exemplo.

R:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | | | | | | |  |  | **2** | | | | | | |  |
|  | 9 | 9 |  | Temp |  |  |  |  |  |  |  |  | Temp | 9 |  |  |  |
|  | 8 | 8 |  |  |  |  |  |  |  | 8 | 8 |  |  |  |  |  |  |
|  | 7 | 2 |  |  |  |  |  |  |  | 7 | 2 |  |  |  |  |  |  |
|  | 6 | 6 |  |  |  |  |  |  |  | 6 | 6 |  |  |  |  |  |  |
|  | A | B |  |  |  |  | Aux |  |  | A | B |  |  |  |  | Aux |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **3** | | | | | | |  |  | **4** | | | | | | |  |
|  |  |  |  | Temp |  |  |  |  |  |  |  |  | Temp | 8 |  |  |  |
|  | 8 | 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 7 | 2 |  |  |  |  |  |  |  | 7 | 2 |  |  |  |  |  |  |
|  | 6 | 6 |  |  |  |  | 9 |  |  | 6 | 6 |  |  |  |  | 9 |  |
|  | A | B |  |  |  |  | Aux |  |  | A | B |  |  |  |  | Aux |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **5** | | | | | | |  |  | **6** | | | | | | |  |
|  |  |  |  | Temp |  |  |  |  |  | Os próximos elementos são diferentes, então não serão removidos e elementos da pilha auxiliar serão enviados de volta | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  | 7 | 2 |  |  |  |  | 8 |  |  |  |
|  | 6 | 6 |  |  |  |  | 9 |  |  |  |
|  | A | B |  |  |  |  | Aux |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **7** | | | | | | |  |  | **8** | | | | | | |  |
|  |  |  |  | Temp | 8 |  |  |  |  |  |  |  | Temp |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | 8 | 8 |  |  |  |  |  |  |
|  | 7 | 2 |  |  |  |  |  |  |  | 7 | 2 |  |  |  |  |  |  |
|  | 6 | 6 |  |  |  |  | 9 |  |  | 6 | 6 |  |  |  |  | 9 |  |
|  | A | B |  |  |  |  | Aux |  |  | A | B |  |  |  |  | Aux |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **9** | | | | | | |  |  | **10** | | | | | | |  |
|  |  |  |  | Temp | 9 |  |  |  |  | 9 | 9 |  | Temp |  |  |  |  |
|  | 8 | 8 |  |  |  |  |  |  |  | 8 | 8 |  |  |  |  |  |  |
|  | 7 | 2 |  |  |  |  |  |  |  | 7 | 2 |  |  |  |  |  |  |
|  | 6 | 6 |  |  |  |  |  |  |  | 6 | 6 |  |  |  |  |  |  |
|  | A | B |  |  |  |  | Aux |  |  | A | B |  |  |  |  | Aux |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **[2,0 Pontos]** considerando que o seguinte algoritmo é executado:



Para processar a seguinte sequência de dados:

5, 7, 12, 4, 0, 4, 6, 8, 67, 34, 23, 5, 0, 44, 33, 22, 6, 0.

Responda:

1. **[1,0 Pontos]** qual seria o conteúdo da fila Q?

R: 5, 7, 12, 4, **4,** 4, 6, 8, 67, 34, 23, 5, **5**, 44, 33, 22, 6, **6**.

1. **[1,0 Pontos]** explique o que o algoritmo faz.

R:

Cria uma fila Q

Enquanto não for o final do arquivo irá

Ler o próximo número presente no arquivo

Se esse número lido for diferente de 0

Irá adicionar esse número à fila Q

Se esse número lido não for diferente de 0, ou seja, se esse número lido for igual a 0

A variável X receberá o valor do último número que foi adicionado à fila Q

E o valor da variável X será adicionado ao fim da fila

Resumidamente:

Lê os números de um arquivo e adiciona eles à fila. Se o valor lido for 0, irá adicionar novamente o último número adicionado à fila.

1. **[2,5 Pontos]** responda os seguintes pontos:
2. **[0,8 Ponto]** descreva em palavras a ideia de um algoritmo que misture duas listas ordenadas em uma única lista, usando operações do TAD Lista;

R:

Uma função receberá 3 listas: uma que será a lista mesclada e as outras duas são as listas que serão mescladas. Haverá uma repetição até que todos os valores de A e B sejam lidos, essa repetição seguirá os seguintes passos: Primeiro será comparado entre os valores das listas A e B, o que tiver menor valor terá o seu valor copiado para um nó na lista F e a lista da qual foi tirado o elemento avançará para o próximo nó. Se o valor do nó de A for igual ao de B, o valor de A é copiado para F e tanto A quanto B avançam para o próximo nó. Até que todos os nós das duas listas já tenham sido percorridos.

1. **[1,0 Ponto]** escreva o algoritmo correspondente;

R:

void mesclarListas (LIST\* Final, LIST\* A, LIST\* B) {

if(!(emptyList(A)&&emptyList(B))){

NODE\* Anode=A->head;

NODE\* Bnode=B->head;

NODE\* Fnode=Final->head;

while((Anode!=NULL)&&(Bnode!=NULL)){

if(Anode==NULL){

addNode(F,Bnode->dataPtr);

Bnode=Bnode->link

} else if(Bnode==NULL){

addNode(F,Anode->dataPtr);

Anode=Anode->link

} else {

if (Anode->dataPtr<Bnode->dataPtr){

addNode(F,Anode->dataPtr);

Anode=Anode->link

} else if(Bnode->dataPtr<Anode->dataPtr){

addNode(F,Bnode->dataPtr);

Bnode=Bnode->link

} else if (Anode->dataPtr==Bnode->dataPtr){

addNode(F,Anode->dataPtr);

Anode=Anode->link

Bnode=Bnode->link

}

}

}

}

}

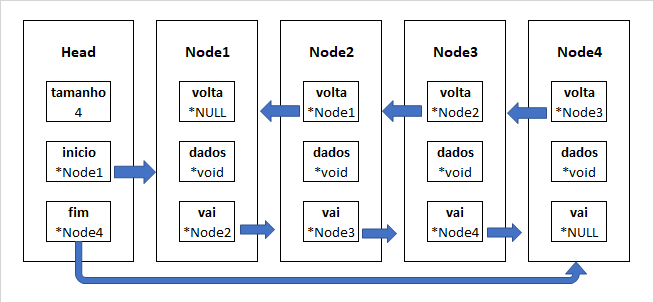
1. **[0,7 Ponto]** ilustre seu algoritmo com um exemplo.

R:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*NULL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*NULL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  | **Node1** | |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 1 |  |  |  | **vai** | \*NULL |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 2 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*NULL |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 3 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 4 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*Node4 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **Node4** | |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **vai** | \*NULL |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **data** | 6 |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA A** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |  | **data** | 6 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA B** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 3 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*NULL |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 5 |  |  |  | **data** | 7 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LISTA F** | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Head F** | |  |  |  | **Node1** | |  |  |  | **Node2** | |  |  |  | **Node3** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Tamanho** | 5 |  |  |  | **vai** | \*Node2 |  |  |  | **vai** | \*Node3 |  |  |  | **vai** | \*Node4 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **Inicio** | \*Node1 |  |  |  | **data** | 2 |  |  |  | **data** | 3 |  |  |  | **data** | 5 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **Node4** | |  |  |  | **Node5** | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **vai** | \*Node5 |  |  |  | **vai** | \*NULL |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **data** | 6 |  |  |  | **data** | 7 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **[1,5 Pontos]** defina o conceito de lista duplamente encadeada.
2. Ilustre graficamente.

R:



1. Cite uma vantagem e uma desvantagem do uso de listas duplamente encadeadas com relação as listas encadeadas simples.

R:

Uma das vantagens do uso de listas duplamente encadeadas em relação às listas encadeadas simples é a maior mobilidade de navegação interna, podendo ela avançar ou retroceder entre os seus elementos.

Uma das desvantagens do uso de listas duplamente encadeadas em relação às listas encadeadas simples o aumento da memória necessária por causa do aumento na quantidade de ponteiros presentes em cada elemento da lista.