

Algoritmos e Programação II

Prof. MSc. Samuel Benjaino Ferraz Aquino



Módulo 1 – Noções de gerenciamento de memória e listas lineares

Prof. MSc. Samuel Benjoino Ferraz Aquino



Roteiro

- Unidade 1: Noções de Gerenciamento de Memória e Listas Lineares Sequenciais
 - Variável simples
 - Variável composta
 - Alocação de memória
 - Listas lineares sequenciais
- Unidade 2: Listas Lineares Encadeadas
- Resumo

Unidade I

Noções de Gerenciamento de Memória e Listas Lineares Sequenciais



Variável

- Região de memória capaz de armazenar dados de um programa

- Exemplo:

num1

num2

char1 = "a"

		Endereço	Conteúdo	
	<i>num1</i>	0	10	
=		1		10
=	<i>num2</i>	2	13.5	13.5
		3		
	<i>char1</i>	4	"a"	
		

Variável Simples

- Variável capaz de armazenar uma **única informação**
- Podem ser de dois tipos:
 - ***Homogênea***
 - ***Heterogênea***

Variável Simples Homogênea

- Variável capaz de armazenar uma **única informação** de um **único tipo**
- São as variáveis mais básicas de um programa
- Exemplo:

<i>num1</i>	=	10
<i>num2</i>	=	13.5
<i>char1</i> = "a"		

Variável Simples Heterogênea

- Variável capaz de armazenar uma **única informação**, que pode ser dividida em “pedaços” de **diferentes tipos**.
- Antes de declarar uma variável simples heterogênea, é necessário definir quais serão os “pedaços” dessa variável.

Variável Simples Heterogênea

- Exemplo: uma variável n que precisa armazenar o nome, a nota 1 e a nota 2 de um aluno.

$n = \text{Nota}(\text{"Samuel"}, 7, 8)$

	<i>Endereço</i>	<i>Conteúdo</i>
n	0	"Samuel"
	1	7
	2	8

Variável Composta

- Variável capaz de armazenar um **conjunto** de informações
- Exemplo: vetores de inteiros

v =

v.append(10)

v.append(15)

v.append(8)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>v</i>	0	10
	1	15
	2	8

...

Variável Composta Homogênea

- Variável capaz de armazenar um **conjunto** de informações **simples**

- Exemplo: vetores de inteiros

v =

`v.append(10)`

`v.append(15)`

`v.append(8)`

	Endereço	Conteúdo
v [0	10
	1	15]
	2	8

...

Variável Composta Heterogênea

- Variável capaz de armazenar um **conjunto** de informações **heterogêneas**

- Exemplo: vetores de notas

```
v =  
v.append(Nota("Samuel",6,  
v.append(Nota("Wallace",7, 8))
```

	Endereço	Conteúdo
v	0	"Samuel"
	1	6
	2	7
	3	"Wallace"
	4	7
	5	8

Alocação de memória

- Toda variável usada **precisa** ser armazenada em uma região de memória
- O momento em que uma variável recebe uma nova região de memória é chamado de **alocação de memória**
- O Interpretador/Compilador aloca uma região de memória **inicialmente sequencial** para as variáveis

Alocação de memória

- Exemplo 1: declaração de três variáveis

a

a = 10

b = 15.5

n = Nota("Samuel", 7, 8)

<i>Endereço</i>	<i>Conteúdo</i>
0	10
1	
2	
3	
4	
5	
6	

Alocação de memória

- Exemplo 1: declaração de três variáveis

$a = 10$

$b = 15.5$

$n = \text{Nota}(\text{"Samuel"}, 7, 8)$

a

b

Endereço	Conteúdo
0	10
1	
2	15.5
3	
4	
5	
6	

Alocação de memória

- Exemplo 1: declaração de três variáveis

$a = 10$

$b = 15.5$

$n = \text{Nota}(\text{"Samuel"}, 7, 8)$

	Endereço	Conteúdo
a	0	10
	1	
	2	15.5
b	3	
	4	"Samuel"
	5	7
n	6	8

Alocação de memória

- Exemplo 2: vetor

$V = [10, 20, 30, 40]$

	<i>Endereço</i>	<i>Conteúdo</i>
V	0	
	1	10
	2	20
	3	30
	4	40
	5	
	6	

Listas lineares

- Uma **lista linear** é um conjunto de $n \geq 0$ **nós/elementos** organizados de acordo com as suas **posições** dentro da lista.

"Alberto"	10	10.5	João	45
0	1	2	3	4

Listas lineares

- Principais operações em uma lista linear
 - **Busca**
 - **Inserção**
 - **Remoção**
- O custo dessas operações depende da implementação da lista linear
 - Lista linear com **alocação sequencial**
 - Lista linear com **alocação encadeada**

Lista linear com alocação sequencial

- Alocar todas as posições da lista de maneira **sequencial** na memória física.
- Ou seja, dada uma lista linear **L** com alocação sequencial, todos os seus elementos estão **contíguos** na memória física.

Lista linear com alocação sequencial

- Exemplo: Lista **L** com quantidade fixa de elementos

$L = [10, 20, 30, 40]$

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	
	1	10
	2	20
	3	30
	4	40
	5	
	6	

Lista linear com alocação sequencial

- Geralmente, listas lineares com alocação sequencial fazem uma **alocação prévia** de posições contíguas
- Exemplo: lista linear **L** inicialmente vazia, implementada com alocação sequencial

L = []

L.append(10)

...

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	

Lista linear com alocação sequencial

- Considere as seguintes operações em uma lista linear **L** com alocação sequencial:
 - **Buscar** um elemento **x**
 - **Buscar** o conteúdo de uma posição **pos**
 - **Inserir** um elemento **x** em uma posição **pos**
 - **Remover** um elemento **x**

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo
<i>L</i>	0	10
	1	20
	2	30
	3	None
	4	None
	5	None

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo	
<i>L</i>	0	10	← x
	1	20	
	2	30	
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	<i>Endereço</i>	<i>Conteúdo</i>	
<i>L</i>	0	10	
	1	20	← <i>x</i>
	2	30	
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo	
<i>L</i>	0	10	
	1	20	
	2	30	← x
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo	
<i>L</i>	0	10	
	1	20	
	2	30	
	3	None	← x
	4	None	
	5	None	

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo
<i>L</i>	0	10
	1	20
	2	30
	3	None
	4	None
	5	None

← x

Lista linear com alocação sequencial

- **Buscar um elemento x**

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 40
buscar(L, x)
```

	Endereço	Conteúdo
<i>L</i>	0	10
	1	20
	2	30
	3	None
	4	None
	5	None

← x

Lista linear com alocação sequencial

- **Buscar** o conteúdo de uma posição **pos**

```
L = []
L.append(10)
L.append(20)
L.append(30)
pos = 1
buscar(L, pos)
```

	Endereço	Conteúdo
<i>L</i>	0	10
	1	20
	2	30
	3	None
	4	None
	5	None

Lista linear com alocação sequencial

- **Buscar** o conteúdo de uma posição **pos**

```
L = []
L.append(10)
L.append(20)
L.append(30)
pos = 1
buscar(L, pos)
```

		Endereço	Conteúdo
L	pos →	0	10
		1	20
		2	30
		3	None
		4	None
		5	None

Lista linear com alocação sequencial

- **Inserir** um elemento **x** em uma posição **pos**

```
L = []
L.append(10)
L.append(20)
L.append(30)
pos = 1
x = 40
insere(L, x, pos)
```

	Endereço	Conteúdo
<i>L</i>	0	10
	1	20
	2	30
	3	None
	4	None
	5	None

Lista linear com alocação sequencial

- **Inserir** um elemento **x** em uma posição **pos**

```
L = []
L.append(10)
L.append(20)
L.append(30)
pos = 1
x = 40
insere(L, x, pos)
```

		Endereço	Conteúdo
L	pos →	0	10
		1	40
		2	30
		3	None
		4	None
		5	None

Lista linear com alocação sequencial

- Remove um elemento x

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 30
remove(L, x)
```

	Endereço	Conteúdo	
L	0	10	← x
	1	20	
	2	30	
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- Remove um elemento x

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 30
remove(L, x)
```

	Endereço	Conteúdo	
L	0	10	
	1	20	← x
	2	30	
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- Remove um elemento x

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 30
remove(L, x)
```

	Endereço	Conteúdo	
L	0	10	
	1	20	
	2	30	← x
	3	None	
	4	None	
	5	None	

Lista linear com alocação sequencial

- Remove um elemento x

```
L = []
L.append(10)
L.append(20)
L.append(30)
x = 30
remove(L, x)
```

	Endereço	Conteúdo
L	0	10
	1	20
	2	None ← x
	3	None
	4	None
	5	None

Unidade 2

Listas Lineares Encadeadas



Lista linear com alocação encadeada

- Alocar as posições da lista **sob demanda** e de maneira **esparsa** na memória física.
- Ou seja, dada uma lista linear **L** com alocação encadeada, os seus elementos estão **espalhados** na memória física.

Lista linear com alocação encadeada

- Uma posição de uma lista linear com alocação encadeada é chamada de **nó**
- Cada **nó** contém:
 - Conteúdo (*chave*)
 - Endereço da próxima posição (*prox*)
- Um **nó** é uma **variável simples heterogênea**

Lista linear com alocação encadeada

- Exemplo: lista linear encadeada **L** com 3 elementos

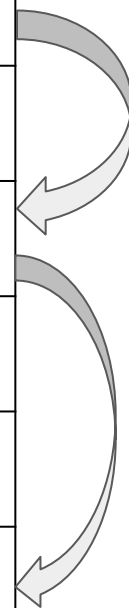
L = []

L.append(10)

L.append(20)

L.append(30)

	Endereço	Conteúdo
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- Considere as seguintes operações em uma lista linear **L** com alocação encadeada:
 - **Buscar** um elemento **x**
 - **Buscar** o conteúdo de uma posição **pos**
 - **Inserir** um elemento **x** em uma posição **pos**
 - **Remover** um elemento **x**

Lista linear com alocação encadeada

- **Buscar um elemento x**

L = []

L.append(10)

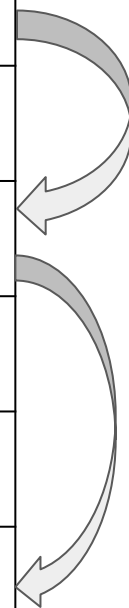
L.append(20)

L.append(30)

x = 30

buscar(L, x)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Buscar um elemento x**

$L = []$

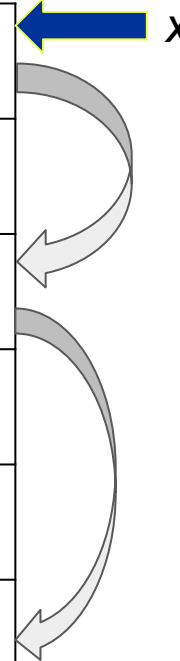
$L.append(10)$

$L.append(20)$

$L.append(30)$

$x = 30$

$buscar(L, x)$

	Endereço	Conteúdo	
L	0	10	
	1		
	2	20	
	3		
	4		
	5	30	

Lista linear com alocação encadeada

- **Buscar um elemento x**

L = []

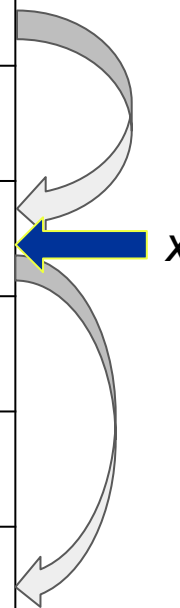
L.append(10)

L.append(20)

L.append(30)

x = 30

buscar(L, x)

	<i>Endereço</i>	<i>Conteúdo</i>	
<i>L</i>	0	10	
	1		
	2	20	
	3		
	4		
	5	30	

Lista linear com alocação encadeada

- **Buscar um elemento x**

L = []

L.append(10)

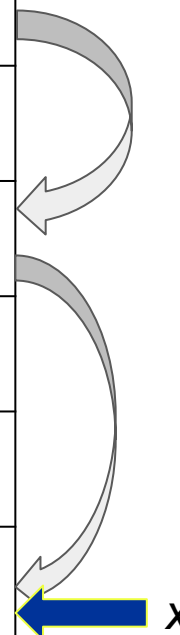
L.append(20)

L.append(30)

x = 30

buscar(L, x)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Buscar** o conteúdo de uma posição **pos**

L = []

L.append(10)

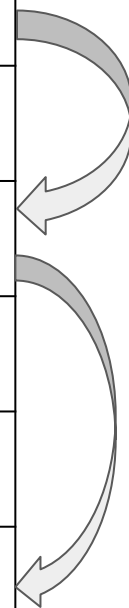
L.append(20)

L.append(30)

pos = 2

buscar(*L*, *pos*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Buscar** o conteúdo de uma posição **pos**

L = []

L.append(10)

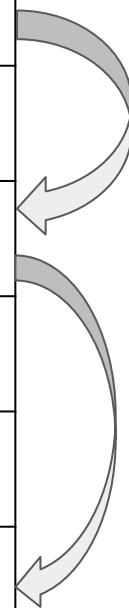
L.append(20)

L.append(30)

pos = 2

buscar(*L*, *pos*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	→ 0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Buscar** o conteúdo de uma posição **pos**

L = []

L.append(10)

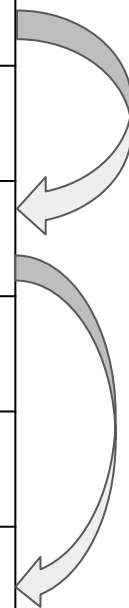
L.append(20)

L.append(30)

pos = 2

buscar(*L*, *pos*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	→ 2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Buscar** o conteúdo de uma posição **pos**

L = []

L.append(10)

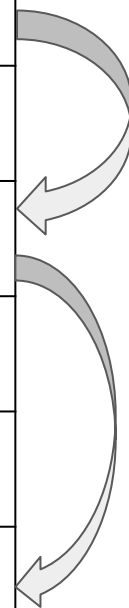
L.append(20)

L.append(30)

pos = 2

buscar(*L*, *pos*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	→ 5	30



Lista linear com alocação encadeada

- **Inserir** um elemento **x** em uma posição **pos**

L = []

L.append(10)

L.append(20)

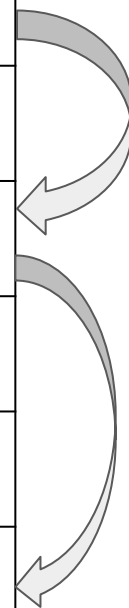
L.append(30)

pos = 1

x = 40

insere(*L*, *pos*, *x*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Inserir** um elemento **x** em uma posição **pos**

L = []

L.append(10)

L.append(20)

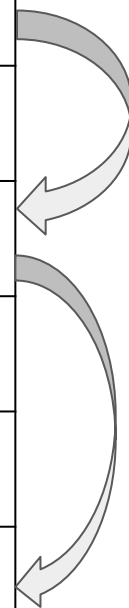
L.append(30)

pos = 1

x = 40

insere(*L*, *pos*, *x*)

	Endereço	Conteúdo
<i>L</i>	→ 0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Inserir** um elemento **x** em uma posição **pos**

L = []

L.append(10)

L.append(20)

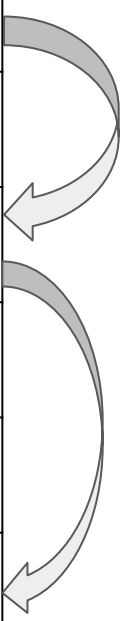
L.append(30)

pos = 1

x = 40

insere(*L*, *pos*, *x*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	→ 2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- **Inserir** um elemento **x** em uma posição **pos**

L = []

L.append(10)

L.append(20)

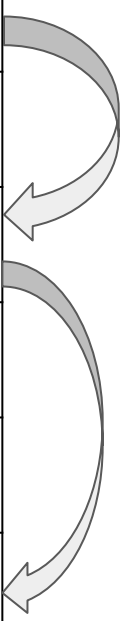
L.append(30)

pos = 1

x = 40

insere(*L*, *pos*, *x*)

	<i>Endereço</i>	<i>Conteúdo</i>
<i>L</i>	0	10
	1	
	→ 2	40
	3	
	4	
	5	30



Lista linear com alocação encadeada

- Remove um elemento x

$L = []$

$L.append(10)$

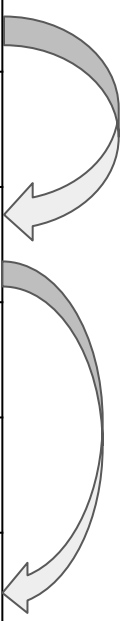
$L.append(20)$

$L.append(30)$

$x = 20$

$remove(L, x)$

	Endereço	Conteúdo
L	0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- Remove um elemento x

$L = []$


$L.append(10)$

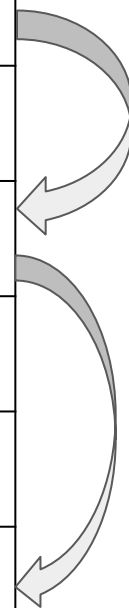
$L.append(20)$

$L.append(30)$

$x = 20$

$remove(L, x)$

	Endereço	Conteúdo
L	 0	10
	1	
	2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- Remove um elemento x

$L = []$

$L.append(10)$

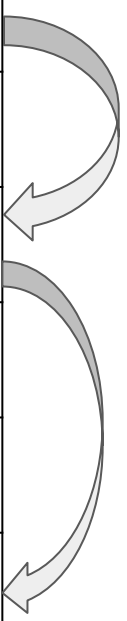
$L.append(20)$

$L.append(30)$

$x = 20$

$remove(L, x)$

	Endereço	Conteúdo
L	0	10
	1	
	→ 2	20
	3	
	4	
	5	30



Lista linear com alocação encadeada

- Remove um elemento x

$L = []$

$L.append(10)$

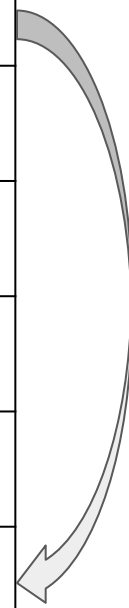
$L.append(20)$

$L.append(30)$

$x = 20$

$remove(L, x)$

	Endereço	Conteúdo
L	0	10
	1	
	2	
	3	
	4	
	5	30



Resumo

- Unidade 1: Noções de Gerenciamento de Memória e Listas Lineares Sequenciais
 - Variável simples
 - Variável composta
 - Alocação de memória
 - Listas lineares sequenciais
- Unidade 2: Listas Lineares Encadeadas