

# *Linguagens de Programação 1*

**Francisco Sant'Anna**  
**Sala 6020-B**

`francisco@ime.uerj.br`

`http://github.com/fsantanna-uerj/LP1`

# Listas Encadeadas

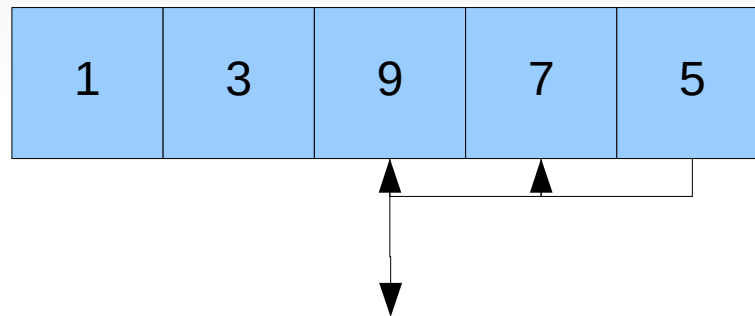
# Exemplo 9.1

- Criar um vetor `vet` de 5 posições
- Ler 5 números e guardá-los em `vet`
- Exibir todos os números de `vet`
- Ler um outro número `I`
- Remover o valor de `vet` no índice `I`
  - Manter o vetor sem buracos

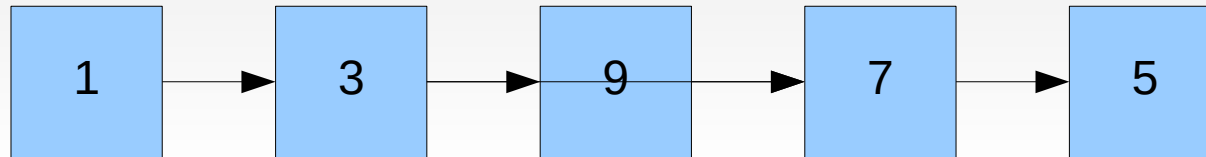
# Exemplo 9.1 - Problemas?

- Precisamos re-ajustar as posições o tempo todo.
- E se eu precisar de mais um elemento?

# Vetor - Rígido



# Lista Encadeada - Flexível



```
struct Caixa {  
    int valor;  
    struct Caixa* prox;  
};
```

## Exemplo 9.2

- Crie um programa que reproduza o estado do slide anterior antes de remover o índice 2
- Cada caixa é um `struct Caixa`
- O `valor` é o número dentro da caixa
- O `prox` é o endereço da próxima caixa
- Teste o programa, ex.:
  - `printf("%d->%d->...\n", c0.valor,  
*(c0.prox).valor, ...)`

## Exemplo 9.3

- Qual é o `prox` da última caixa?
  - Tem que ser um endereço que “não existe”
  - `NULL == 0`
- Crie uma função que receba um ponteiro para uma caixa e percorra todas as caixas (até o `NULL`), exibindo todos os valores
- `void exhibe (struct Caixa* caixa);`



## Exemplo 9.4

- Altere o Ex. 9.2 para remover a caixa no índice 2
- Exiba o valor das caixas (usando `exibe`)
- Insira a caixa removida, no início da lista
- Exiba o valor das caixas (usando `exibe`)
- Mantenha um novo ponteiro `cabeca` que sempre aponta para a caixa inicial
  - `struct Caixa* cabeca =`