

# Estrutura de Dados Básicas I.

Aula 2 – Algoritmos de busca

**Prof. Eiji Adachi M. Barbosa**

# Objetivos

- Apresentar algoritmos de busca
  - Busca sequencial
  - Busca binária

**Por que é  
importante buscar?**

# O que é uma **busca?**

# O que é um algoritmo de busca?

# Algoritmos de busca

- Entrada:
  - Um argumento  $K$
  - Uma coleção de registros  $R_1, R_2, \dots R_n$ , cujas chaves são, respectivamente,  $K_1, K_2, \dots K_n$
- Problema:
  - Encontre o registro que possui chave igual a  $K$
- Saída:
  - Sucesso: localizou registro contendo chave igual a  $K$
  - Insucesso: não localizou registro contendo chave igual a  $K$

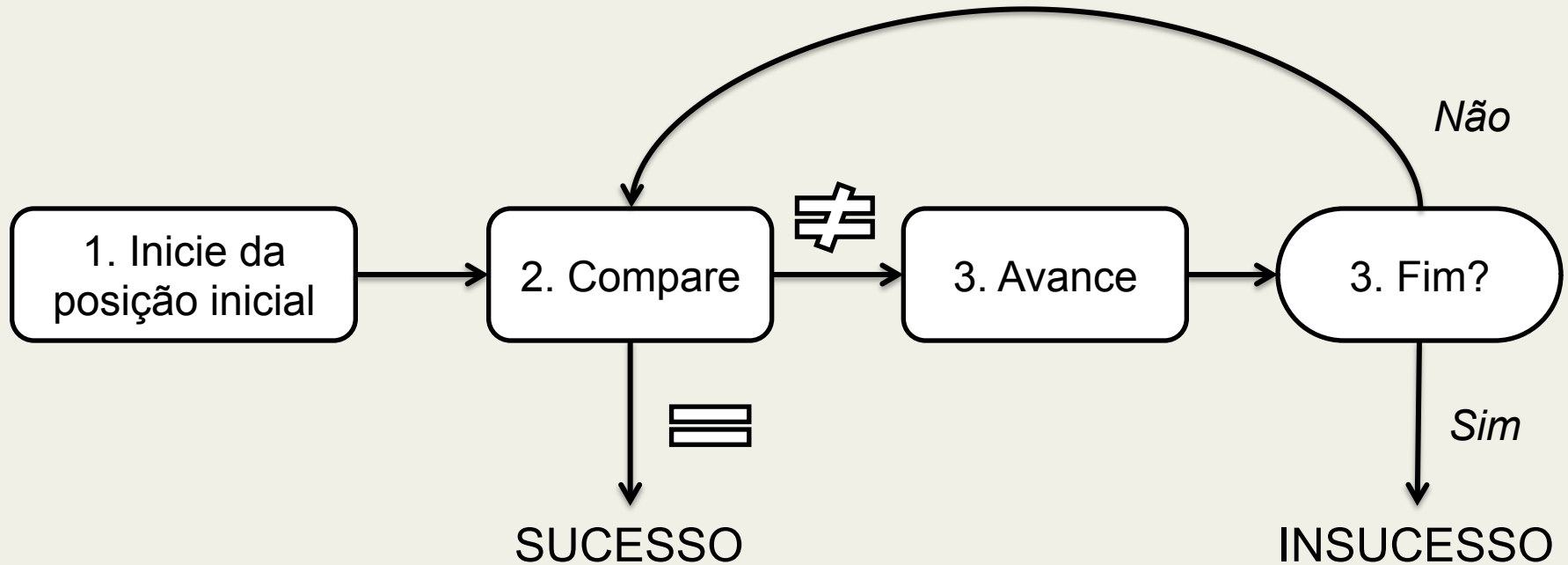
# Algoritmos de busca

- Busca sequencial
- Busca binária

# BUSCA SECUENCIAL



# Busca sequencial



# Busca sequencial

```
int busca_sequencial(int k, int colecao[], int tamanho)
{
    int posicao = 0;
    do {
        if( k == colecao[posicao] ){
            return posicao; // SUCESSO
        }
        posicao = posicao + 1;
    } while(posicao < tamanho);
    return -1; // INSUCESSO
}
```

# Busca sequencial – Questões

- Considere o vetor `int c[] = [14, 16, 15, 13, 18, 17]`, o inteiro `int n = 6` a função abaixo:

```
int busca_sequencial(int k, int colecao[], int tamanho)
{
    int posicao = 0;
    do {
        if( k == colecao[posicao] ){
            return posicao; // SUCESSO
        }
        posicao = posicao + 1;
    } while(posicao < tamanho);
    return -1; // INSUCESSO
}
```

- Quantos itens do vetor são examinados para as seguintes chamadas:
  - `busca_sequencial(13, c, n)`? `busca_sequencial(17, c, n)`? `busca_sequencial(12, c, n)`?  
`busca_sequencial(20, c, n)`?

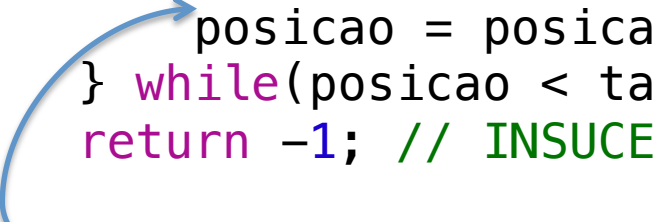
# Busca sequencial – Questões

- Qual o pior caso do algoritmo de busca sequencial?
- E o melhor caso?

**E se os dados de entrada  
estiverem ordenados,  
é possível melhorar a  
busca sequencial?**

# Busca sequencial

```
int busca_sequencial(int k, int colecao[], int tamanho)
{
    int posicao = 0;
    do {
        if( k == colecao[posicao] ){
            return posicao; // SUCESSO
        }
        posicao = posicao + 1;
    } while(posicao < tamanho);
    return -1; // INSUCESSO
}
```



```
else if( k < colecao[posicao] ){
    return -1; // INSUCESSO
}
```

# Busca sequencial

```
int busca_sequencial(int k, int colecao[], int tamanho)
{
    int posicao = 0;
    do {
        if( k == colecao[posicao] ){
            return posicao; // SUCESSO
        }
        else if( k < colecao[posicao] ){
            return -1; // INSUCESSO
        }
        posicao = posicao + 1;
    } while(posicao < tamanho);
    return -1; // INSUCESSO
}
```

# Busca sequencial – Questões

- Considere o vetor `int c[] = [13, 15, 17, 19, 21, 23]`, o inteiro `int n = 6` a função abaixo:

```
int busca_sequencial(int k, int colecao[], int tamanho)
{
    int posicao = 0;
    do {
        if( k == colecao[posicao] ){
            return posicao; // SUCESSO
        }
        else if( k < colecao[posicao] ){
            return -1; // INSUCESSO
        }
        posicao = posicao + 1;
    } while(posicao < tamanho);
    return -1; // INSUCESSO
}
```

- Quantos itens do vetor são examinados para as chamadas:
  - `busca_sequencial(19, c, n)`? `busca_sequencial(23, c, n)`? `busca_sequencial(12, c, n)`? `busca_sequencial(33, c, n)`?



# Busca sequencial – Questões

- Qual o melhor caso do algoritmo de busca sequencial para dados ordenados?
- E o pior caso?
- Quais melhorias em relação ao algoritmo de busca sequencial sem dados ordenados?

# BUSCA BINÁRIA

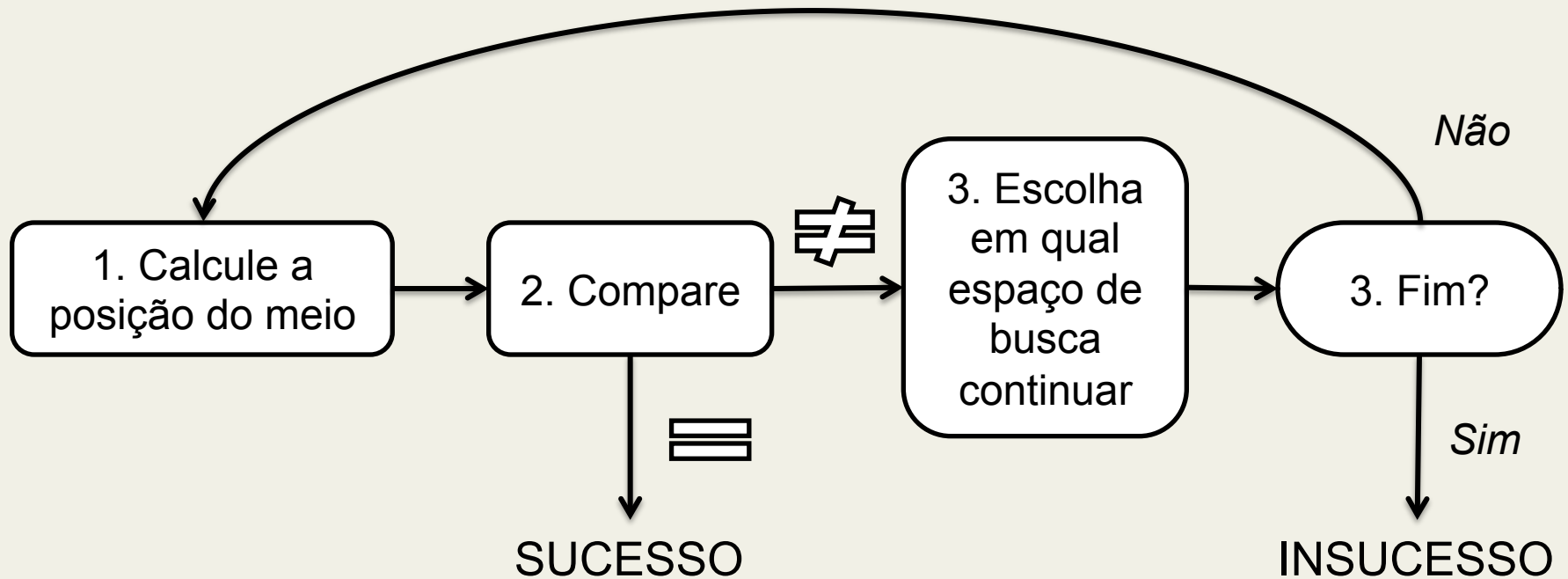
**E se os dados de entrada  
estiverem ordenados,  
é possível fazer melhor do  
que a busca sequencial?**

# Analogia:

## Buscar uma palavra em um dicionário

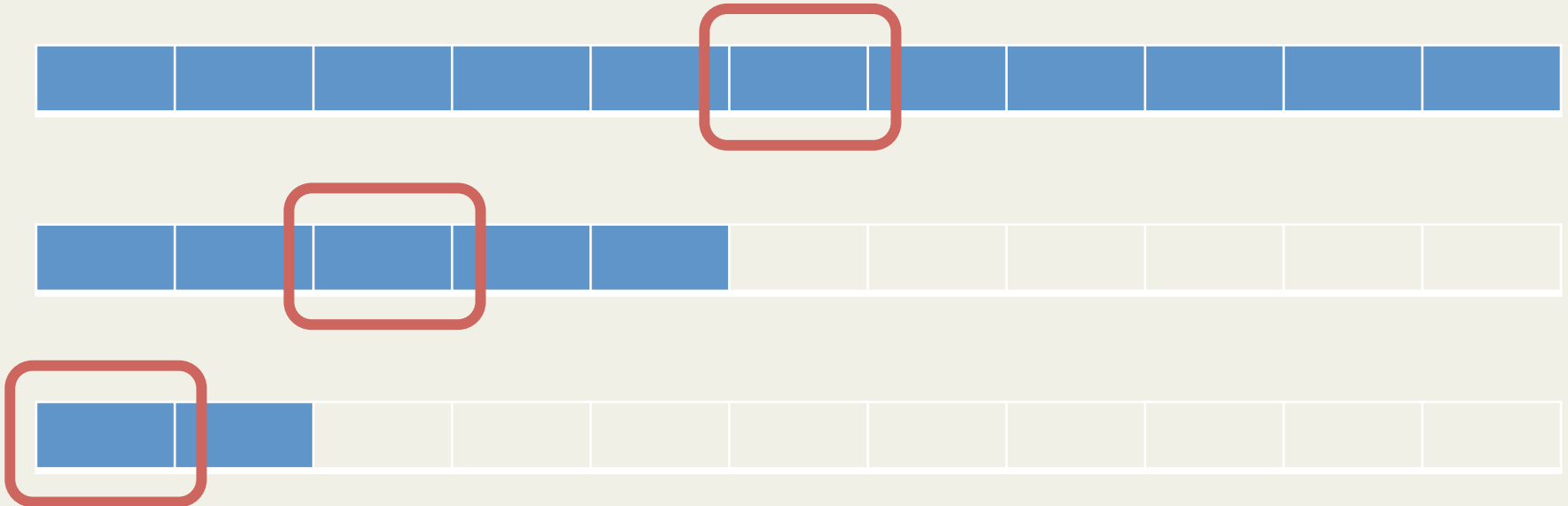


# Busca binária



# Busca binária

- Reduções sucessivas do espaço de busca



# Busca binária – Recursiva

```
int busca_binaria(int k, int colecao[], int inicio, int fim) {  
    if( inicio > fim ){  
        return -1;  
    }  
    int meio = (inicio+fim)/2;  
    if( k < colecao[meio] ){  
        return busca_binaria (k, colecao, inicio, meio-1);  
    }  
    else if( k > colecao[meio] ){  
        return busca_binaria (k, colecao, meio+1, fim);  
    }  
    else{  
        return meio;  
    }  
}
```

# Busca binária – Iterativa

```
int busca_binaria(int k, int colecao[], int tamanho){
    int inicio = 0, fim = tamanho-1, meio;
    while(inicio <= fim){
        meio = (inicio+fim)/2;
        if( k < colecao[meio] ){
            fim = meio -1;
        }
        else if( k > colecao[meio] ){
            inicio = meio + 1;
        }
        else {
            return meio;
        }
    }
    return -1;
}
```



# Busca binária – Questões

- Considere o vetor `int c[] = [13, 15, 17, 19, 21, 23]`, o inteiro `int n = 6` a função abaixo:

```
int busca_binaria(int k, int colecao[], int tamanho){
    int inicio = 0, fim = tamanho-1, meio;
    while(inicio <= fim){
        meio = (inicio+fim)/2;
        if( k < colecao[meio] )
            fim = meio -1;
        else if( k > colecao[meio] )
            inicio = meio + 1;
        else
            return meio;
    }
    return -1;
}
```

- Quantos itens do vetor são examinados para as chamadas:
  - `busca_binaria(19, c, n)`? `busca_binaria(23, c, n)`? `busca_binaria(12, c, n)`?  
`busca_binaria(33, c, n)`?

# Busca binária – Questões

- E se o vetor de entrada estiver em ordem decrescente? O que precisa mudar?

```
int busca_binaria(int k, int colecao[], int tamanho){
    int inicio = 0, fim = tamanho-1, meio;
    while(inicio <= fim){
        meio = (inicio+fim)/2;
        if( k < colecao[meio] )
            fim = meio -1;
        else if( k > colecao[meio] )
            inicio = meio + 1;
        else
            return meio;
    }
    return -1;
}
```

# Atividade prática

- Entre no SIGAA, baixe o .pdf contendo as instruções para a parte prática da aula de hoje

# Atividade prática

- Compile e execute o código.

# Estrutura de Dados Básicas I.

Aula 2 – Algoritmos de busca

**Prof. Eiji Adachi M. Barbosa**