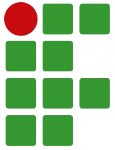


INSTITUTO FEDERAL
Sertão Pernambucano
Campus Salgueiro

Estrutura de Dados e Algoritmos com Java

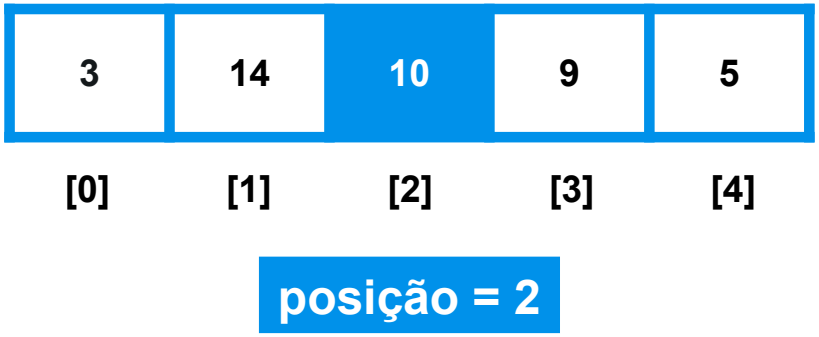
Prof. Heraldo Gonçalves Lima Junior
heraldo.junior@ifsertao-pe.edu.br

1. Listas Lineares Sequenciais

CONTINUAÇÃO...

1.1. Obtendo elemento de uma determinada posição

- Agora vamos criar o nosso método de busca. Para isso, precisamos passar a posição desejada para o nosso método.



1.1. Obtendo elemento de uma determinada posição

```
public String busca(int posicao){
    return this.elementos[posicao];
}
```

- Mas e se a posição informada for maior que o número de índices do vetor?
- E se for menor que zero?

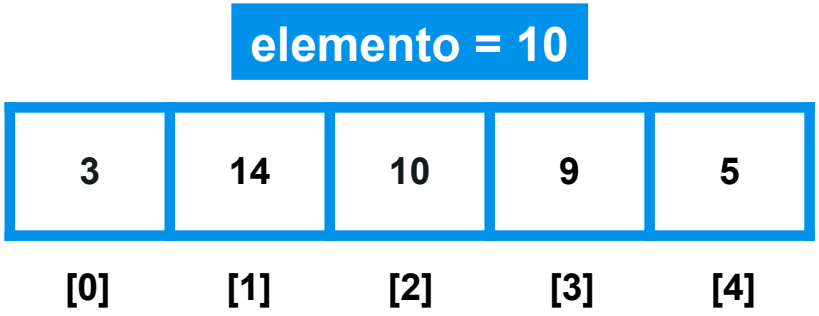
1.1. Obtendo elemento de uma determinada posição

```
public String busca(int posicao) {
    try {
        return this.elementos[posicao];
    } catch (Exception e) {
        return e.getMessage();
    }

}
```

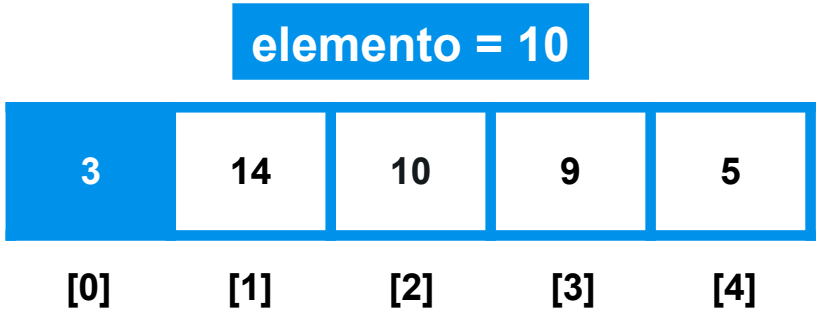
1.2. Verificando se um elemento existe

- Agora vamos verificar se um determinado elemento está armazenado no vetor.



1.2. Verificando se um elemento existe

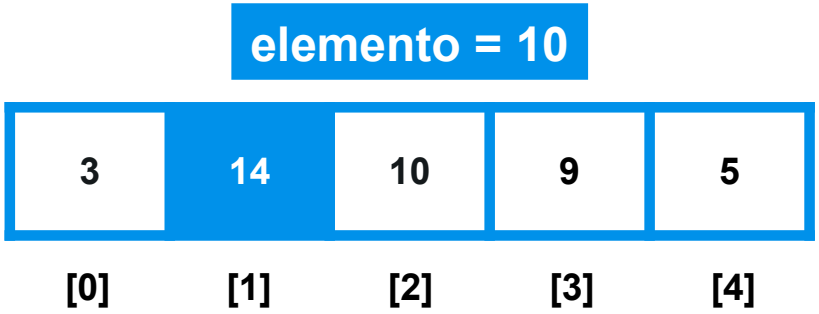
- Busca Sequencial:



- O elemento é igual ao armazenado neste índice?

1.2. Verificando se um elemento existe

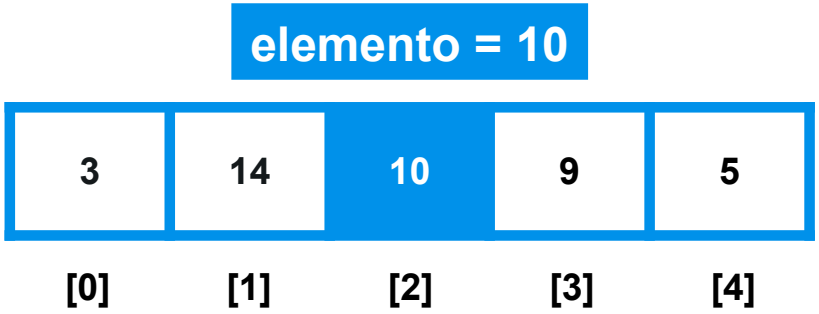
- Busca Sequencial:



- O elemento é igual ao armazenado neste índice?

1.2. Verificando se um elemento existe

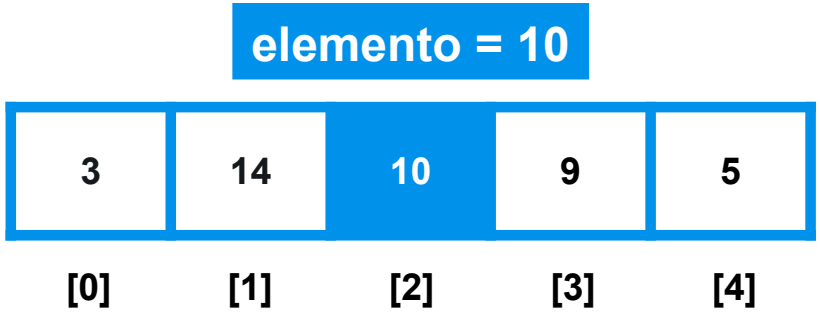
- Busca Sequencial:



- O elemento é igual ao armazenado neste índice?

1.2. Verificando se um elemento existe

- Busca Sequencial:



- O elemento é igual ao armazenado neste índice? **SIM!**

return true

1.2. Verificando se um elemento existe

```
public boolean verifica(String elemento){
    for(int i=0;i<this.tamanho;i++){
        if(this.elementos[i].equals(elemento)){
            return true;
        }
    }
    return false;
}
```

Da pra melhorar? A busca considera maiúsculas e minúsculas?

1.2. Verificando se um elemento existe

```
public boolean verifica(String elemento){
    for(int i=0;i<this.tamanho;i++){
        if(this.elementos[i].equalsIgnoreCase(elemento)){
            return true;
        }
    }
    return false;
}
```

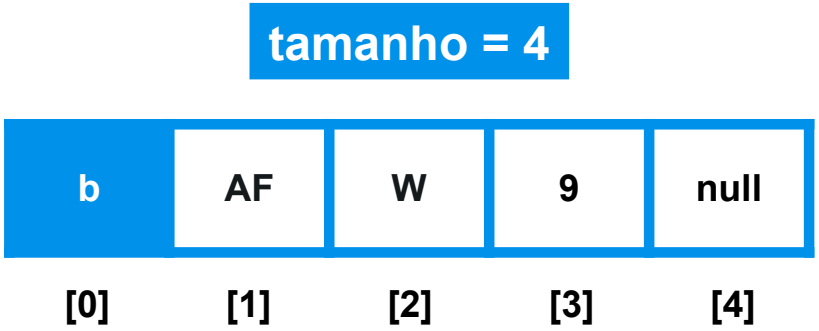
- Melhorou bastante... Mas e seu eu quiser retornar a posição do elemento, caso ele seja encontrado?

1.2. Verificando se um elemento existe

```
public int verifica(String elemento){
    for(int i=0;i<this.tamanho;i++){
        if(this.elementos[i].equalsIgnoreCase(elemento)){
            return i;
        }
    }
    return -1;
}
```

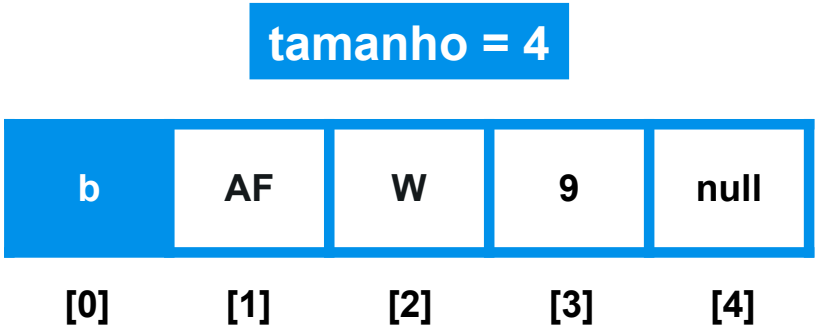
- Agora eu retorno o número do índice onde o valor está armazenado. Se não for encontrado, retorno -1.

1.3. Adicionando elementos em qualquer posição



- E se eu quiser inserir o valor **“X”** na posição 0 do vetor?

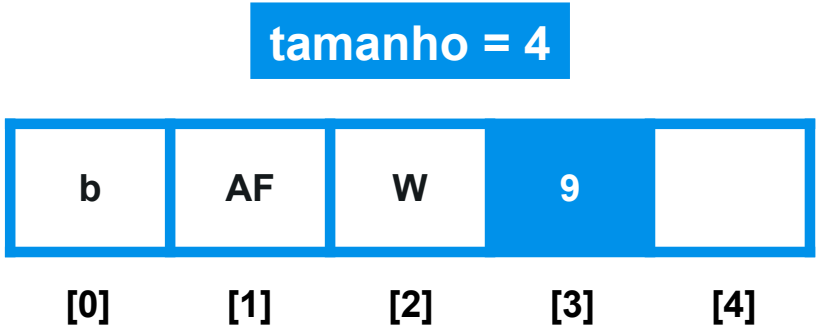
1.3. Adicionando elementos em qualquer posição



- A posição 0 já está ocupada com o valor “b”.
- Perderemos o valor “b”?

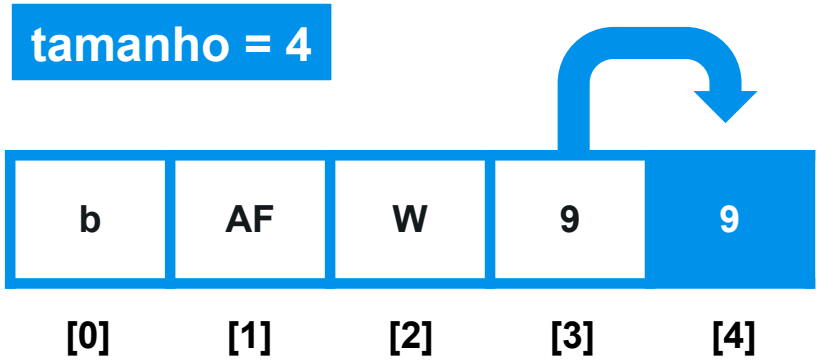
1.3. Adicionando elementos em qualquer posição

- Vamos simplificar esse exemplo... E se eu quisesse inserir o elemento **“X”** na posição 3?



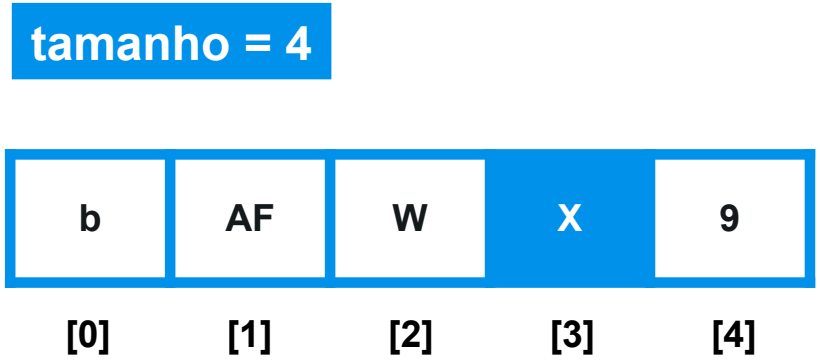
1.3. Adicionando elementos em qualquer posição

- Vamos simplificar esse exemplo... E se eu quisesse inserir o elemento “X” na posição 3?
- **Bastaria deslocar o “9” para a posição 4!**



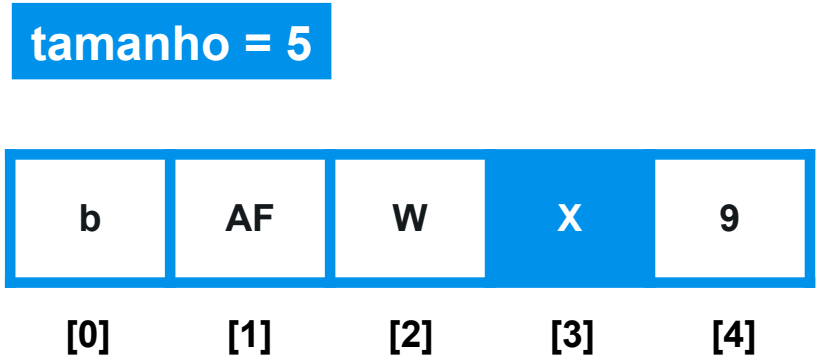
1.3. Adicionando elementos em qualquer posição

- Agora que o valor “9” foi copiado para a posição 4, posso **atribuir o valor “X” à posição 3**.



1.3. Adicionando elementos em qualquer posição

- Por último, incremento o valor de **tamanho**, que agora vale 5.



- Fácil né?

1.3. Adicionando elementos em qualquer posição

- Nosso código fica assim:

```
public boolean adiciona(int posicao, String elemento){
    if(!((posicao>=0) && (posicao <= this.tamanho))){
        throw new IllegalArgumentException("Posição inválida!");
    }
    for(int i=this.tamanho-1; i>=posicao; i--){
        this.elementos[i+1] = this.elementos[i];
    }
    this.elementos[posicao]=elemento;
    this.tamanho++;
    return true;
}
```

1.4. Aumentando a Capacidade do Vetor

- No método construtor do vetor, ao criarmos a nossa lista vazia, definimos a capacidade deste vetor.

```
//CRIAR LISTA VAZIA
public Vetor(int capacidade){
    this.elementos = new String[capacidade];
    this.tamanho = 0;
}
```

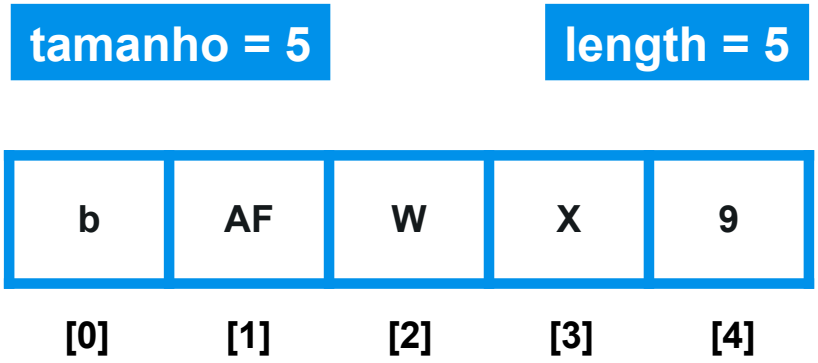
- Porém, na vida real esse limite de elementos pode ser complicado. Precisamos aumentar essa capacidade de acordo com a necessidade.

1.4. Aumentando a Capacidade do Vetor

- Precisamos então adicionar o método **aumentaCapacidade()** no nosso método adiciona. Assim, caso a capacidade do nosso vetor não seja o suficiente no momento da inserção, **será criado outro vetor**, agora com capacidade maior. **Todos os elementos do antigo vetor serão transferidos para o novo**, e agora sim o novo elemento será inserido.

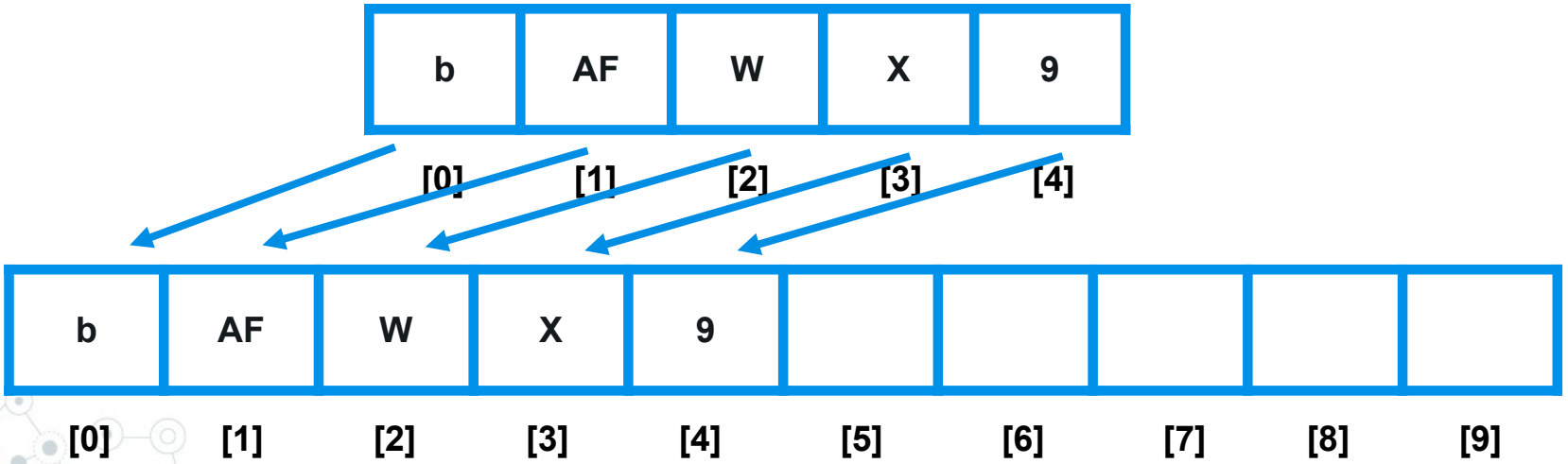
1.4. Aumentando a Capacidade do Vetor

- No nosso código, quando será preciso aumentar a capacidade do vetor?
  - **Sempre que tamanho == this.elementos.length**



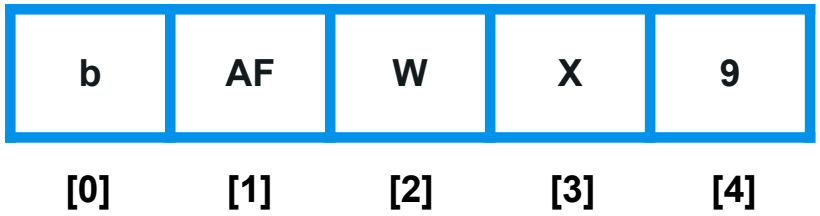
1.4. Aumentando a Capacidade do Vetor

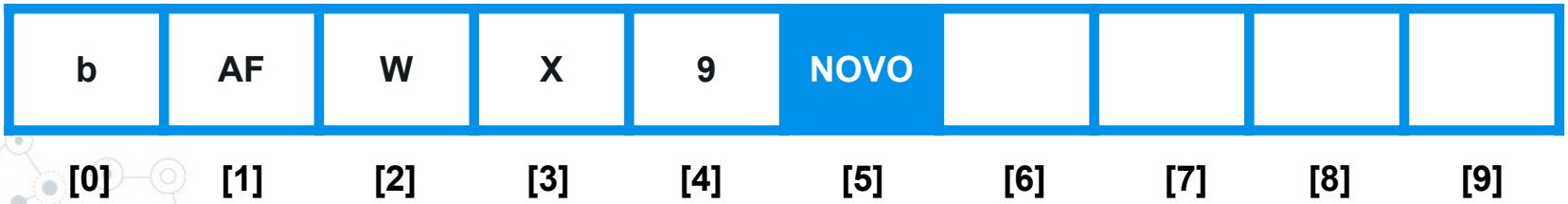
- Dessa forma crio outro vetor e transfiro todos os elementos para este novo, com capacidade maior:



1.4. Aumentando a Capacidade do Vetor

- Após criar e transferir todos os elementos, eu faço a inserção do novo elemento.





1.4. Aumentando a Capacidade do Vetor

- Nosso código fica assim:

```
public void aumentaCapacidade(){
    if(this.tamanho == this.elementos.length){
        String[] elementosNovos = new String[this.elementos.length*2];
        for (int i=0; i<this.elementos.length; i++){
            elementosNovos[i] = this.elementos[i];
        }
        this.elementos = elementosNovos;
    }
}
```

1.4. Aumentando a Capacidade do Vetor

- Agora é só alterar o método **adiciona().** Agora ele não precisa mais retornar um booleano, pois a operação sempre irá dar certo.

```
public void adiciona(String elemento) {
    if(this.tamanho<this.elementos.length) {
        this.elementos[this.tamanho] = elemento;
    }else {
        this.aumentaCapacidade();
        this.elementos[this.tamanho] = elemento;
    }
    this.tamanho++;
}
```

1.4. Aumentando a Capacidade do Vetor

- Agora é só alterar o método **adicionaPos()**:

```
public boolean adicionaPos(int posicao, String elemento) {

    if(!((posicao >= 0) && (posicao <= this.tamanho))) {
        throw new IllegalArgumentException("Posição inválida!");
    }
    if(this.tamanho==this.elementos.length) {
        this.aumentaCapacidade();
    }
    for(int i=this.tamanho-1; i>=posicao; i--) {
        this.elementos[i+1] = this.elementos[i];
    }
    this.elementos[posicao] = elemento;
    this.tamanho++;
    return true;
}
```

RES PIRA !

Obrigado!

Perguntas?

 heraldo.junior@ifsertao-pe.edu.br

 heraldolimajr.com.br