Estruturas de Dados

Sumário

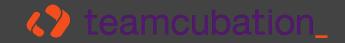
- . O que são estruturas de dados?
- . Estruturas de dados em Java:
 - . Arrays, Lists, Sets
 - . Maps
 - . Pilhas, Filas



O que são estruturas de dados?

Visto que os dados normalmente são utilizadas de forma conjunta, precisamos organizá-los de certas formas que permitam a utilização destes dados de forma eficiente.

Estas formas de organizar podemos chamar de estruturas de dados.



Arrays

É uma coleção ordenada de dados. Em Java, os dados dos arrays devem possuir o mesmo tipo, bem como também tem tamanho fixo.



Arrays - Inicialização

```
// Inicializacao direta
String[] linguagens = {"Java", "PHP", "Python"};
// Inicializacao com tamanho
String[] linguagens = new String[3];
linguagens[0] = "Java";
linguagens[1] = "PHP";
linguagens[2] = "Python";
// Inicializacao com tamanho definido por variável
int tamanho = 3;
String[] linguagens = new String[tamanho];
// Inicializacao direta utilizando operador new
String[] linguagens = new String[]{"Java", "PHP", "Python"};
```

Arrays - Conceitos Importantes

- . Índice começa do zero e vai até o tamanho 1;
- . O tamanho é fixo;
- . Precisam ter o mesmo tipo;
- . O acesso dos valores é pelo índice:

```
System.out.println(linguagens[2]); // Python
System.out.println(linguagens[3]); // ERRO!
```



Arrays - Conceitos Importantes

- . Consultar tamanho
- . Acesso aos itens
- . Atribuir valores aos itens

```
String[] array = {"A", "B", "C", "D"};
System.out.println(array.length); // 4
System.out.println(array[0]); // A
array[0] = "novo valor";
System.out.println(array[0]); // novo valor
```

teamcubation

Lists e Sets

. São coleções/conjuntos de itens, com tamanho dinâmico.

. Interface List: coleção ordenada de itens. Permite elementos duplicados.

. Interface Set: conjunto não ordenado de itens, e que não permite elementos duplicados.



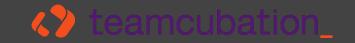
Lists e Sets

```
import java.util.*;
public class Main
   public static void main(String[] args) {
       List<String> lista = new ArrayList<>();
        lista.add("Banana");
        lista.add("Maçã");
        lista.add("Maçã"); // Elemento duplicado
       System.out.println(lista); // Saída: [Banana, Maçã, Maçã]
       Set<String> conjunto = new HashSet<>();
       conjunto.add("Banana");
       conjunto.add("Maçã");
       conjunto.add("Maçã"); // Elemento duplicado, não será adicionado
        System.out.println(conjunto); // Saída: [Maçã, Banana]
```



Lists - Métodos Principais

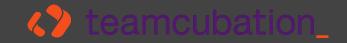
- . Consultar tamanho método size()
- . Verificar se a Lista está vazia método isEmpty()
- . Adicionar item método add()
- . Buscar item método get()
- . Remover item método remove()



Set - Métodos Principais

- . Consultar tamanho método size()
- . Verificar se o Set está vazio método isEmpty()
- . Adicionar item método add()
- . Remover item método remove()

Por não ser ordenado, o acesso aos itens do Set se indiretamente. Por exemplo, através de iteração.



Maps

. Um map é uma coleção de pares chave-valor, onde cada chave é única e mapeada para um valor.

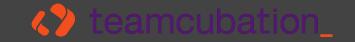
. Os maps são úteis quando você precisa recuperar rapidamente valores através de chaves atribuídas previamente.

. Em Java podemos utilizar as implementações da interface Map.



Maps

```
Map<String, String> mapa = new HashMap<>();
mapa.put("chave 1", "melancia");
mapa.put("chave 2", "banana");
mapa.put("chave 3", "abacaxi");
System.out.println(mapa);
// {chave 3=abacaxi, chave 1=melancia, chave 2=banana}
```



Map - Métodos Principais

- . Consultar tamanho método size()
- . Verificar se o mapa está vazio método isEmpty()
- . Adicionar item método put()
- . Buscar item método get()
- . Remover item método remove()



Pilhas (Stack)

. Pilhas são estruturas de dados seguindo o princípio de que o último elemento a ser adicionado será o primeiro a ser removido.

. Este princípio é conhecido como LIFO - Last In, First Out.

. Em Java podemos utilizar a classe Stack.



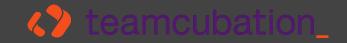
Pilhas (Stack)

```
Stack<String> stack = new Stack<>();
stack.push("1");
stack.push("2");
stack.push("3");
while (!stack.isEmpty()) {
    System.out.println(stack.pop()); // 3 2 1
```



Stack - Métodos Principais

- . Consultar tamanho método size()
- . Verificar se a pilha está vazia método isEmpty()
- . Adicionar item ao topo da pilha método push()
- . Remover item do topo da pilha método pop()
- . Verificar item do topo da pilha método peek()

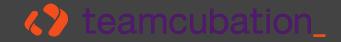


Filas (Queue)

. Filas são estruturas de dados seguindo o princípio de que o primeiro elemento a ser adicionado será o primeiro a ser removido.

. Este princípio é conhecido como FIFO - First In, First Out.

. Em Java podemos utilizar as implementações de Queue.



Filas (Queue)

```
Queue<String> queue = new ArrayDeque<>();
queue.add("1");
queue.add("2");
queue.add("3");
while (!queue.isEmpty()) {
    System.out.println(queue.poll()); // 1 2 3
```

Queue - Métodos Principais

- . Consultar tamanho método size()
- . Verificar se a fila está vazia método isEmpty()
- . Adicionar item ao final da fila método add()
- . Remover item do início da fila método poll()
- . Verificar item do início da fila método peek()



Perguntas?