PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
ESTRUTURA DE DADOS ORIENTADA A OBJETOS
ADS1232
PROF. MSC. ANIBAL SANTOS JUKEMURA

PROGRAMAÇÃO ORIENTADA A OBJETOS [JAVA]

Agenda

- Listas
- Exemplos
- Exercícios

Listas

- Em DEITEL (2005, pág. 673), uma coleção é uma estrutura de dados, na realidade um objeto, que pode armazenar ou agrupar referências a outros objetos (um contêiner). As classes e interfaces da estrutura de coleções são membros do pacote java.util e a **Figura 1** apresenta a hierarquia de algumas destas interfaces oferecidas pelo Java.

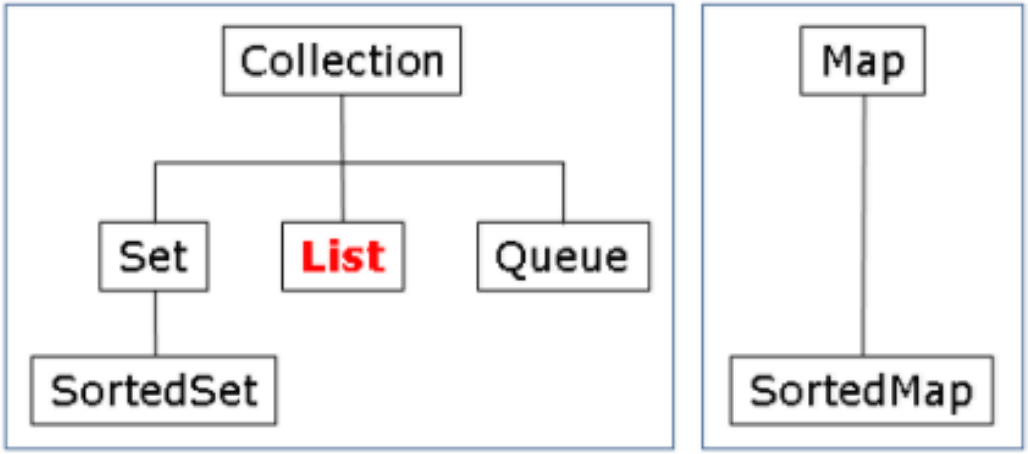

Figura 1. Hierarquia das interfaces da estrutura de coleções

Listas

Na lista de interfaces da estrutura de coleções destacam-se os conjuntos, listas, filas e mapas. Vamos começar pelas listas:

Lista (List):

- É uma coleção indexada de objetos às vezes chamada de sequência.
- Como nos vetores, índices de **List** são baseados em zero, isto é, o índice do primeiro elemento é zero.
- Além dos métodos herdados de **Collection**, **List** fornece métodos para manipular elementos baseados na sua posição (ou índice) numérica na lista, remover determinado elemento, procurar as ocorrências de um dado elemento e percorrer sequencialmente (**ListIterator**) todos os elementos da lista.
- A interface **List** é implementada por várias classe, incluídas as classes **ArrayList** (implementada como vetor), **LinkedList** e **Vector**.

Listas

ArrayList

- Usaremos esta classe na implementação de vetores dinâmicos (ou redimensionáveis).
- Principais métodos:

- boolean **add**(**Object** element): Adiciona o elemento especificado no final da lista.

- void **add**(**int** index, **Object** element): Insere o elemento especificado na posição indicada da lista.

- void **clear**(): Remove todos os elementos da lista.

- boolean **contains**(**Object** element): Retorna verdadeiro se a lista contém o elemento especificado e falso caso contrário.

- Object **get**(**int** index): Retorna o i-ésimo elemento da lista.

- int **indexOf**(**Object** element): Retorna a posição da primeira ocorrência do elemento especificado na lista.

Listas

ArrayList

- Usaremos esta classe na implementação de vetores dinâmicos (ou redimensionáveis).
- Principais métodos:
- boolean **isEmpty**(): Retorna verdadeiro se a lista estiver vazia e falso caso contrário.
- int **lastIndexOf**(Object element): Retorna a posição da última ocorrência do elemento especificado na lista.
- Object **remove**(int index): Remove o i-ésimo elemento da lista.
- Object **set**(int index, Object element): Substitui o i-ésimo elemento da lista pelo elemento especificado.
- int **size**(): Retorna o número de elementos da lista.

Listas

ArrayList: exemplo 1 – Lista de Inteiros

Neste exemplo em sala realizaremos as seguintes operações:

1 – Criar uma lista de Inteiros:

```
ArrayList<Integer> numeros = new ArrayList();
```

2 - Usar o método add() para gravar números digitados pelo usuário:

```
numeros.add(ler.nextInt());
```

3 - Mostrar os "n" números da lista (usando o índice) e o método método size() que retorna o número de elementos da lista (tamanho da lista):

```
int n = numeros.size();
numeros.get(i);
```

4- Vamos remover o i-ésimo número da lista:

```
numeros.remove(i);
```

5- Mostar os "n" elementos da lista (usando for-each)

```
for (int num: numeros)
```


```java
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

public class Lista01Numeros {

    public static void main(String[] args) {
        Scanner dado = new Scanner(System.in);
        int i;
        ArrayList<Integer> numeros = new ArrayList();

        for (i = 0; i < 10; i++)
        {
            System.out.print("Digite o " + (i+1) + "o numero: ");
            numeros.add(dado.nextInt());
        }
        System.out.println("Percorrendo o ArrayList (usando o índice)");
        int tamanho = numeros.size();
        for (i = 0; i < tamanho; i++)
        {
          System.out.printf("Posição %d: %s\n", i, numeros.get(i));
        }

        System.out.printf("\nInforme a posição a ser excluída:\n");
        i = dado.nextInt();

        try
        {
            numeros.remove(i);
        }
        catch (IndexOutOfBoundsException e)
        {
            System.out.printf("\nErro: posição inválida (%s).", e.getMessage());
        }

        System.out.printf("\nPercorrendo o ArrayList (usando for-each)\n");
        i = 0;
        for (int num: numeros) {
          System.out.printf("Posição %d: %s\n", i, num);
          i++;
        }
    }
}
```

Listas

ArrayList: exemplo 2 - Agenda

Neste exemplo em sala realizaremos as seguintes operações:

1 – Criar uma lista de Strings:

```
ArrayList<object> agenda = new ArrayList();
```

2 - Usar o método add() para gravar contatos na agenda:

```
agenda.add("Darth Vader;62 9999-1111");
```

3 - Mostrar os "n" contatos da agenda (usando o índice) e o método método size() que retorna o número de elementos da agenda (tamanho da lista):

```
int n = agenda.size();
agenda.get(i);
```

4- Vamos remover o i-ésimo contato da agenda:

```
agenda.remove(i);
```

5- Mostar os "n" contatos da agenda (usando for-each)

```
for (agenda contato: agenda)
```

6- mostrando os "n" contatos da agenda (com iterator)

```
Iterator<object> iterator = agenda.iterator();
iterator.hasNext()
iterator.next()
```

PUC GOIÁS

```java
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

public class Lista02Agenda {

    public static void main(String[] args) {
        Scanner dado = new Scanner(System.in);

        // [ A ] declarando e instanciando uma agenda de contatos
        ArrayList<String> agenda = new ArrayList<String>();

        // [ B ] usando o método add() para gravar 4 contatos na agenda
        agenda.add("Darth Vader;62 9999-1111");
        agenda.add("Bruna Ribeiro;64 98888-0001");
        agenda.add("Maria Rita;65 97777-0000");
        agenda.add("Mestre Yoda;44 4444-4444");

        int i;

        // [ C ] mostrando os "n" contatos da agenda (usando o índice)
        // número de elementos da agenda: método size()
        System.out.printf("Percorrendo o ArrayList (usando o índice)\n");

        for (i = 0; i < agenda.size(); i++) {
          System.out.printf("Posição %d: %s\n", i, agenda.get(i));
        }

        System.out.printf("\nInforme a posição a ser excluída:\n");
        i = dado.nextInt();
```


```
31
32          try {
33              // [ D ] remove o i-ésimo contato da agenda
34              agenda.remove(i);
35          } catch (IndexOutOfBoundsException e) {
36              // exceção lançada para indicar que um índice (i)
37              // está fora do intervalo válido (de 0 até agenda.size()-1)
38              System.out.printf("\nErro: posição inválida (%s).",
39                  e.getMessage());
40          }
41
42          // [ E ] mostrando os "n" contatos da agenda (usando for-each)
43          System.out.printf("\nPercorrendo o ArrayList (usando for-each)\n");
44          i = 0;
45          for (String contato: agenda) {
46              System.out.printf("Posição %d: %s\n", i, contato);
47              i++;
48          }
49
50          // [ F ] mostrando os "n" contatos da agenda (com iterator)
51          System.out.printf("\nPercorrendo o ArrayList (usando iterator)\n");
52          i = 0;
53          Iterator<String> iterator = agenda.iterator();
54          while (iterator.hasNext()) {
55              System.out.printf("Posição %d- %s\n", i, iterator.next());
56              i++;
57          }
58      }
59  }
```

PUC GOIÁS

Exercícios:

Exercício 01 – Crie uma lista com a classe ArrayList de 10 números inteiros. Após isso, liste apenas os números pares. Após isso, liste apenas os números ímpares. Para listagem, use os métodos size() e get(i).

Exercício 02 – Crie uma lista com a classe ArrayList de 10 números double. Após isso, imprima a lista usando o método iterator. Após, substitua o elemento na posição de índice 5 por "9,7" (use o método set()). Liste novamente a lista. Após isso, mostre a média desses números. Após isso, liste o maior número digitado. Remova todos os elementos da lista e teste se está vazia.

Listas

Referência Bibliográfica Principal

• DEVMEDIA. Disponível em https://www.devmedia.com.br . Acessado em Fevereiro de 2018.