**RESUMO DA AULA 10 - GENERICS E ESTRUTURA DE DADOS**

ALUNO: Denilson José do Bom Jesus Silva de Lima

**- ESTRUTURA DE DADOS**

Forma de organizar operações em um conjunto de dados eficientemente, como com o uso de: Array, Listas, Pilhas, Fila, HashTable e Árvores.

1. A Lista Encadeada [Linked List]

Uma estrutura e dados sequenciais, ou seja, não se acessa um elemento de forma aleatória, como nos Arrays. É composta por nós (*Nodes*) que armazena o dado e um ponteiro para o próximo elemento (sendo possível adicionar um novo nó e remover um nó preexistente). Pode ser duplamente encadeada e possui como ponto positivo a facilidade de crescer a estrutura e o fato de a memória ser alocada conforme o necessário. Já um ponto negativo é que para chegar a um elemento, pode ser necessário percorrer a lista inteira.

2. Fila [Queue]

Um tipo de lista encadeada. possui um estrutura do tipo FIFO (primeiro que entra, primeiro que sai). em sua implementação existem pelos menos dois métodos:

- *Enqueue*: insere um novo elemento no final da fila

- *Dequeue*: remove o primeiro elemento da fila

(Ex. Método para adicionar elementos na Fila:)

public class Informacao {

private String nome;

private int idade;

public Informacao(String nome, int indade){ ... }

}

public classe Fila {

private class Node{

Informacao info;

Node next;

public Node(Informacao info) {

this.info = info;

this.next = null;

}

}

private Node primeiro;

private Node ultimo;

public Fila() {

this.primeiro = null;

this.ultimo = null;

}

public void enqueue(Informacao info) {

Node newNode = new Node()

if(this.ultimo == null) {

this.primeiro = this.ultimo = newNode;

} else {

this.ultimo.next = newNode;

this.ultimo = newNode;

}

}

public Informacao dequeue() {

if(this.primeiro == null) {

return null;

}

Informacao info = this.primeiro.info;

this.primeiro = this.primeiro.next;

return info;

}

}

**- GENERICS**

Caso queira armazenar outro tipo de dado na fila, pode-se usar uma classe Genérica, que cria um Tipo genérico, que ficou disponível a partir do JDK 5.0. (Ex:)

public class FilaGenerica <T> {

private class Node{

T info;

Node next;

public Node(T info) {

this.info = info;

this.next = null;

}

[...]

}

*// implementacao*

public static void main(String[] args){

FilaGenerica<Informacao> fila = new FilaGenerica();

[...]

}

}

Também é possível criar Métodos Genéricos. (Ex:)

public static <T> void imprimeVetor(T v[]){

for(T e : v) System.out.printf(e + " ");

System.out.println();

}

- EXCEÇÕES DO GENERICS

Apenas tipo referenciados (objetos) podem ser utilizados. Não há Generics para tipos primitivo e é necessário Usar as classes *Byte, Short, Integer, Long, Float,*

*Double, Boolean* e *Character*.

Não há diferença entre Generics e o Objects, mas, é possível restringir os tipos que podem ser utilizados em uma classe/método genérico.

3. PILHA [STACK]

Tipo de lista encadeada que trabalha com a ideia do conceito de organização LIFO: Last In First out (Último que Entra, Primeiro que Sai).

Existem pelo menos dois métodos para a implementação de uma Pilha:

*- Push:* Insere um novo elemento no topo da pilha.

*- Pop:* Remove o elemento do topo da pilha.

4. HASH TABLES

Tipo de estrutura de dados que associa um *value* (dado) a uma *key* (chave). Tem como premissa ser uma forma simples de encontrar um elemento, usando de uma chave para localizar um determinado dado a ela associado.

**- JAVA COLLECTIONS**

Um *framework* de interfaces e classes para representar diversas estruturas de dados.

1. Implementações da Interface List

Define uma coleção ordenada, podendo conter elementos duplicados. Em geral, o usuário tem controle total sobre a posição onde cada elemento é inserido e pode recuperá-los através de seus índices.

- ArrayList – um array que pode aumentar (alocação dinâmica de memória). Busca é rápida, mas inserir/remover elementos aumentam com o tamanho da estrutura.

- LinkedList – implementa lista ligada (encadeada). Como característica de uma lista encadeada, a busca é linear e inserções e exclusões são rápidas.

2. Implementações da Interface Set

Interface que define uma coleção que não permite elementos duplicados.

- HashSet

- TreeSet

- LinkedHashSet

3. Implementações da Interface Queue

Um tipo de coleção para manter uma lista de prioridades, onde a ordem dos seus elementos, definida pela implementação de *Comparable* ou *Comparator*, determina essa prioridade. Com a interface fila pode-se criar filas e pilhas;

4. Implementações da Interface Map

Mapeia chaves para valores. Cada elemento tem na verdade dois objetos: uma chave e um valor. Os valores podem ser duplicados, mas chaves não.

- HashMap

- TreeMap

- LinkedHashMap