MC 302EF - Atividade de Laboratório no. 7

Objetivos

- Uso de programação concorrente em Java.

Descrição da Atividade

Esta atividade consiste na implementação do algoritmo *quicksort* usando os recursos de programação concorrente disponíveis na linguagem Java. Em linhas gerais, o algoritmo *quicksort* para ordenação de um vetor v consiste nos seguintes passos:

1. *escolher um 'elemento pivô'*
2. *rearranjar o vetor de forma que este seja formado por três partes:*
    i. *sequência s1 formada pelos elementos que precedem(*) o pivô*
    ii. *o elemento pivô*
    iii. *sequência s2 formada pelos elementos que sucedem(*) o pivô.*
3. *ordenar s1 usando o próprio algoritmo quicksort.*
4. *ordenar s2 usando o próprio algoritmo quicksort.*

(*) *de acordo com o critério de ordenação utilizado*

A implementação tradicional do algoritmo *quicksort* é baseada numa função que é chamada recursivamente nos passos 3 e 4, acima.

Nesta atividade os passos 3 e 4 devem ser implementados como linhas de execução paralelas, ou 'threads', na terminologia usada em Java. Na implementação usando 'threads', os passos 3 e 4 devem ser alterados para

3. *Se tamanho(s1) > M ordenar s1 usando o próprio algoritmo quicksort senão ordenar s1 usando o algoritmo do bubblesort.*
4. *Se tamanho(s2) > M ordenar s2 usando o próprio algoritmo quicksort senão ordenar s2 usando o algoritmo do bubblesort.*

O valor de M deve ser uma constante da implementação e seu valor igual a 7.

A interface Comparator

A interface **`Comparator`**, fornecida como parte desta atividade, define o método **`precede(),`** responsável pela comparação de dois objetos. Esse método define o tipo de ordenação a ser usado por cada instância de **`MultithreadSorter`**.

Implementação

A classe MultithreadSorter, a ser implementada nesta atividade deverá

- Ter como parâmetro a classe dos objetos que devem ser ordenadas
- Implementar a interface Runnable (java.lang)
- Ter um construtor público da forma

```
public MultithreadSorter('Classe'[] lista, Comparator<'Classe'> comparator)
```

onde

- `'Classe'` se refere à classe dos objetos a serem ordenados
- O parâmetro **`comparator,`** é o responsável pela comparação utilizada na ordenação dos objetos.

- Publicar o método

```
public void sort(){ ... }
```

  - Esse método será o responsável pela ordenação do vetor de objetos da classe 'Classe', passado como parâmetro.
  - Deverá usar o objeto 'comp', passado como parâmetro, que implementa a interface 'Comparator', para a comparação dos objetos a serem ordenados.
  - A execução de sort() deverá por sua vez
    - Criar um objeto Thread, baseado numa nova instância da classe MultithreadSorter.
    - Iniciar a execução dessa thread através do método start().
    - Esperar pelo encerramento dessa thread.
    - Retornar ao chamador.

- Publicar o método run(), previsto na interface Runnable. Esse método será o responsável pela implementação do algoritmo 'quicksort concorrente', disparando as 'threads filhas' para as partes do vetor e esperando pela sua conclusão (através do método join()).

Programas de testes

Para os testes desta atividade serão usados dois programas de testes, disponibilizados como anexo. O primeiro deles é mostrado a seguir.

```
package concorrencia;

public class TestAtiv7_1 {

	/* lista de nomes a ser ordenada */
	static String[] listaDeNomes = { "Luiz", "Vinicius","Geraldo" ... }

	/** Escreve a lista de nomes na saída padrão */
	static void print(){
	   for(int i = 0; i < listaDeNomes.length; i++ )
	       System.out.println(i+": "+ listaDeNomes[i]);
	}

	public static void main(String[] args) throws InterruptedException {
	   MultithreadSorter <String> sorter =
	       new MultithreadSorter<String>(listaDeNomes, new Comparator7());
	       sorter.sort();
	       print();
	}

}

class Comparator7 implements Comparator<String> {

	@Override
	public boolean precede(String n1, String n2) {
	       return n1.compareTo(n2) < 0;
	}

}
```

A saída do programa de testes mostrado acima consiste na sequência de nomes ordenada em ordem crescente, com um nome por linha da saída padrão.

Data de entrega: 19/05/2016