Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Algoritmos e Estruturas de Dados

Semestre de Inverno 2009/10

Terceira série de exercícios

Observações:

- Data de entrega: **4 de Janeiro de 2010**.

1. Realize a classe genérica `AedHashTable<E>` para a representação e manipulação de tabelas de dispersão, contendo os seguintes métodos de instância públicos:

   - `void put(E e)`, que insere o elemento e na tabela.
   - `boolean contains(E e)`, que retorna `true` se e só se a tabela contiver um elemento igual a e.
   - `void removeAllDuplicates()`, que remove todos os elementos duplicados, de forma a que a instância só contenha elementos distintos.
   - `E putUnique(E e)`, que insere o elemento e na tabela se e só se a tabela não contiver outro elemento igual. Caso já exista um elemento igual, retorna esse elemento. Caso contrário, retorna `null`.

   Esta classe deve implementar a interface `Iterable<E>`. O iterador retornado pelo método `iterator()` deve suportar remoção.

   Utilize o método `equals`, declarado na classe`Object`, para verificar se dois elementos são iguais.

2. Realize a classe contendo os seguintes métodos estáticos para a manipulação de árvores. Assuma que cada objecto do tipo `Node<E>` tem 3 campos: um `value` do tipo `E` e duas referências, `left` e `right`, para os descendentes respectivos.

   2.1. `public static <E> int copyToArray(Node<E> root, E[] v);`

   que copia para o *array* `v` os elementos da árvore binária de pesquisa referenciada por `root`. Os elementos devem ficar organizados no *array* por ordem decrescente, com o maior elemento na primeira posição. Caso a dimensão da árvore exceda a dimensão do *array*, apenas são copiados os últimos `v.length` elementos.

   2.2. `public static Node<Integer> createBSTFromArithmeticProgression(int a, int d, int n)`

   que retorna a referência para o nó raíz duma árvore binária de pesquisa contendo os inteiros da progressão aritmética com início em a, dimensão n e factor d (diferença entre dois elementos consecutivos $d = a_i - a_{i-1}$). Assuma que o factor d é maior do que zero. A árvore resultante deve estar balanceada.

   2.3. `public static <E extends Comparable<E>> int rangeCount(Node<E> root, E l, E r)`

   que retorna o número de elementos da árvore binária de pesquisa com raíz `root` pertencentes ao intervalo $[l, r]$. Considere que o critério de organização da árvore binária de pesquisa é a *ordenação natural* do tipo `E`.

3. Acrescente, à classe `Iterables` realizada na segunda série de exercícios, o método

   ```
   public static <E> Iterable<E> distinct(Iterable<E> iter)
   ```

   que retorna um objecto com a interface `Iterable<E>`, representando a sequência `iter` sem os elementos duplicados.

   Não é necessário implementar o método `remove`, pertencente à interface genérica `Iterator<E>`.

   A implementação deste método deve minimizar o espaço ocupado pelo iterador.

4. Esquematize a inserção da seguinte sequência de elementos numa *B-Tree* com $M = 2T - 1 = 5$ (número máximo de chaves por nó/página):

   $\{10, 20, 30, 40, 50, 4, 5, 6, 7, 31, 32, 33, 34\}$.

ISEL, 16 de Novembro de 2009.