CMPT 225

Lecture 33 – Dictionary Data Collection
Dictionary

- Value-oriented ADT
- Also called a map or table
- Data collection used when we need to manage data by value
  - <key, value> pair
Dictionary Operations

- **isEmpty( )**: Is the dictionary empty?
- **getElementCount( )**: Returns the number of elements in the dictionary
- **insert( element, key )**: Insert element into the dictionary
- **remove( key ) or remove( element* )**: Remove element with a given key from the dictionary
  * element object containing only a value in its attribute used as the key
- **removeAll( )**: Remove all element from the dictionary
- **retrieve( key ) or retrieve( element* )**: Get element with a given key from dictionary (but does not remove it)
  * element object containing only a value in its attribute used as the key
- **contains( key ) or contains( element* )**: Test whether a dictionary contains element with a given key
  * element object containing only a value in its attribute used as the key
- **traverse( visit )**: Traverse a dictionary in sorted key order and call the function **visit** (provided by the client code) when “visiting” each element

Assuming each element has a unique search key!
Various ways of implementing Dictionary data collection ADT class

Possible underlying “data structures”:
- Unsorted List (array-based or link-based)
- Sorted List (array-based or link-based)
- Binary Search Tree
- AVL
- Hashing
Selecting a Dictionary data collection ADT class implementation

- To make an intelligent choice among the various possible dictionary implementations, we must analyze the efficiency with which each of the implementations supports the dictionary operations.
Overview of value-oriented data collection ADT’s so far

<table>
<thead>
<tr>
<th></th>
<th>Unsorted List</th>
<th>Sorted List</th>
<th>BST</th>
<th>AVL</th>
<th>Hashing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>insert</strong></td>
<td>O(1)</td>
<td>O(n)</td>
<td>O(log₂ n)</td>
<td>O(log₂ n)</td>
<td></td>
</tr>
<tr>
<td><strong>remove</strong></td>
<td>O(n)</td>
<td>O(n)</td>
<td>O(log₂ n)</td>
<td>O(log₂ n)</td>
<td></td>
</tr>
<tr>
<td><strong>retrieve</strong></td>
<td>O(n)</td>
<td>O(log₂ n)</td>
<td>O(log₂ n)</td>
<td>O(log₂ n)</td>
<td></td>
</tr>
<tr>
<td><strong>traverse</strong></td>
<td>O(n)</td>
<td>O(n)</td>
<td>O(n)</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
</tbody>
</table>