Bag class prototypes

Bag classes maintain unbounded collections of items potentially containing duplicate elements.

These are currently implemented in several ways, differing in representation strategy, algorithmic efficiency, and appropriateness for various tasks. (Listed next to each are average (followed by worst-case, if different) time complexities for [a] adding, [f] finding (via seek, contains), [d] deleting elements).

XPBags implement unordered Bags via XPlexes. ([a O(1)], [f O(n)], [d O(n)]).

OXPBags implement ordered Bags via XPlexes. ([a O(n)], [f O(log n)], [d O(n)]).

SLBags implement unordered Bags via linked lists ([a O(1)], [f O(n)], [d O(n)]).

OSLBags implement ordered Bags via linked lists ([a O(n)], [f O(n)], [d O(n)]).

SplayBags implement ordered Bags via Sleater and Tarjan's (JACM 1985) splay trees. The algorithms use a version of ``simple top-down splaying'' (described on page 669 of the article). (Amortized: [a O(log n)], [f O(log n)], [d O(log n)]).

VHBags implement unordered Bags via hash tables. The tables are automatically resized when their capacity is exhausted. ([a O(1)/O(n)], [f O(1)/O(n)], [d O(1)/O(n)]).

CHBags implement unordered Bags via chained hash tables. ([a O(1)/O(n)], [f O(1)/O(n)], [d O(1)/O(n)]).

The implementations differ in whether their constructors require an argument to specify their initial capacity.

Initial capacities are required for plex and hash table based Bags. If none is given **DEFAULT_INITIAL_CAPACITY** (from **<T>defs.h**) is used.

Bags support the following operations, for some class **Bag**, instances **a** and **b**, **Pix ind**, and base element **x**. Since all implementations are virtual derived classes of the **<T>Bag** class, it is possible to mix and match operations across different implementations, although, as usual, operations are generally faster when the particular classes are specified in functions operating on Bags.

Pix-based operations are more fully described in the section on Pixes. See *Pseudo-indexes* in **/NextLibrary/Documentation/GNU/libg++/Intro.rtf**.

Bag a; or **Bag a(int initial_size)**	Declares a to be an empty Bag. The second version is allowed in Bag classes that require initial capacity or sizing specifications.
a.empty()	returns true if a is empty.
a.length()	returns the number of elements in a.
ind = a.add(x)	inserts x into a, returning its index.
a.del(x)	deletes one occurrence of x from a.
a.remove(x)	deletes all occurrences of x from a.
a.clear()	deletes all elements from a;
a.contains(x)	returns true if x is in a.
a.nof(x)	returns the number of occurrences of x in a.

a(ind)	returns a reference to the item indexed by ind.
int = a.first()	returns the Pix of first item in the Bag or 0 if the Bag is empty. For ordered Bags, this is the Pix of the least element.
a.next(ind)	advances ind to the Pix of next element, or 0 if there are no more.
ind = a.seek(x. Pix from = 0)	Sets ind to the Pix of the next occurrence x, or 0 if there are none. If from is 0, the first occurrence is returned, else the following from.