Project 2 Michalski Kacper 293464

Templated class Ring with typename <Key>. It is implemented by Node structures pointing at other Node structures, until the final one points at the first one. Node structures each have 3 variables: key of type Key, prev of type Node* and next of type Node*. They also have a constructor for creating nodes. Ring points at the first Node with a private Node* head. Ring has 3 constructors – blank constructor, copy constructor and constructor from arrays.

Method:	**Arguments:**	**Description:**	**Returns:**
getsize	void	Returns number of elements	Number of elements
print/revprint	void	Prints list to cout	void
putFront, putBack	Key k, Data d	Puts a new node either at the front, or at the back	void
remove	Key k, Node* nodeptr	Recursive function that removes all elements with a given key.	Number of elements removed
clear	void	Clears the list	void
insert	int I, Key k, Data d	Inserts at index i element with k and d. If it's outside of boundaries, it does nothing.	void
multinsert	Key k, Key where	Insert elements k after elements where.	True if any inserted, else false.
search	Key k	Searches for elements with Key k.	Number of elements
popFront, popBack	void	Deletes either the first, or last element	void
isempty	void	Returns true if no elements, else false.	True if no elements on the list, else false
front/back	void	Returns data either at the front or at the back	Data at the front, or at the back

Operator:	**Description:**	**Returns:**
=	Assignment operator, copies the "right" list to the "left" one (deep copy).	The "right" list
+	Concatenates two lists with the first one as the beginning of the new list, and second one as the end	Product of concatenation
+=	Puts the "right" list at the end of the "left" one	Product of concatenation
[]	Returns data at index, if out of bounds returns 0	Data at index
==	Comparison operator.	True if rings are equal, else false.