# Programming Assignment 4

Implement the following specification of UnsortedType using a circular linked list as the implementation structure.

template <class ItemType>

struct NodeType;

/\* Assumption: ItemType is a type for which the operators

"<" and "==" are defined—either an appropriate built-in type or a class that overloads these operators. \*/

template <class ItemType>

class UnsortedType

{

public:

// Class constructor, destructor, and copy constructor

UnsortedType();

~UnsortedType();

UnsortedType(const UnsortedType<ItemType>&);

void operator=(UnsortedType<ItemType>);

bool IsFull() const;

// Determines whether list is full.

// Post: Function value = (list is full)

int GetLength() const;

// Determines the number of elements in list.

// Post: Function value = number of elements in list.

void RetrieveItem(ItemType& item, bool& found);

// Retrieves list element whose key matches item's key

// (if present).

// Pre: Key member of item is initialized.

// Post: If there is an element someItem whose key matches

// item's key, then found = true and item is a copy of

// someItem; otherwise found = false and item is

// unchanged.

// List is unchanged.

void InsertItem(ItemType item);

// Adds item to list.

// Pre: List is not full.

// item is not in list.

// Post: item is in list.

void DeleteItem(ItemType item);

// Deletes the element whose key matches item's key.

// Pre: Key member of item is initialized.

// One and only one element in list has a key matching

// item's key.

// Post: No element in list has a key matching item's key.

void ResetList();

// Initializes current position for an iteration through the

// list.

// Post: Current position is prior to list.

void GetNextItem(ItemType&);

// Gets the next element in list.

// Pre: Current position is defined.

// Element at current position is not last in list.

// Post: Current position is updated to next position.

// item is a copy of element at current position.

private:

NodeType<ItemType>\* listData;

int length;

NodeType<ItemType>\* currentPos;

};

## Deliverables

* A listing of the specification and implementation files for UnsortedType
* A listing of the driver program for your test plan
* A listing of the test plan as input to the driver
* A listing of the output from the driver