Rafaela Lomboy
CISP 430
Prof. Dixon

A2 Linked List Class Designs

(*Note: These class designs are a rough draft, but still show the most important members. Updated members are shown in UML diagrams.*)

```
INCLUDE iostream

class DataClass
   Typedef
    +   typedef dataType : [data type]
   Constructors
    +   DataClass()
    +   DataClass(dataType data)
    +   DataClass(const DataClass& dc)    // copy
   Modification methods
    +   set(dataType data) : void
   Constant methods
    +   get() const : dataType
   Overloaded operators
    +   operator=(const DataClass& dc) : DataClass&     // copy assignment!
   Friend functions
    +   operator==(const DataClass&, const DataClass&) : bool
    +   operator<(const DataClass&, const DataClass&) : bool       // for SORTING
    +   operator>(const DataClass&, const DataClass&) : bool
   Member Variables
    -   dataType data
endclass
```

```
INCLUDE DataClass header and iostream

class ListItem
   Typedef
    +   typedef listDataType : DataClass      // type of data/class list item holds
   Constructors
    +   ListItem()
    +   ListItem(listDataType)
    +   ListItem(listDataType, ListItem* nextItem)     // may remove…
    +   ListItem(const ListItem&)                       // copy - pointers!
   Destructor
    +   ~ListItem()
   Modification methods
    +   setData(const listDataType&) : void
    +   setNext(ListItem*) : void
    +   setPrev(ListItem*) : void
    +   getNext() : ListItem*          // note: getter methods for pointers can modify ListItems
    +   getPrev() : ListItem*          // may need to add const version if const ListItem* is used
   Constant methods
    +   getData() const : listDataType
   Overloaded operators
    +   operator=(const ListItem&) : ListItem&        // copy assignment - do NOT copy pointers
    +   operator==(const ListItem&) : ListItem&       // may remove...
   Friend functions
    +   operator<<(ostream&, const ListItem&) : ostream&          // for displaying list
   Member variables
    -   data : listDataType
    -   nextPtr : ListItem*
    -   prevPtr : ListItem*
endclass
```

Association: ListItem ----- 1 : 1 ------ includes>>DataClass

(Note: NEVER return ListItem from public List method - it’s hidden!
ListItem is hidden and internal to the List class!*)*

INCLUDE **ListItem** header and iostream

```
class List
   Typedef
    +   typedef iterator : Iterator
    +   typedef listDataType : ListItem::listDataType      // List depends on data type of ListItem
   Constructors
    +   List()
    +   List(listDataType)
    +   List(ListItem)              // may remove...
    +   List(const List&)           // use deep copy - do NOT copy pointers!
   Destructor
    +   ~List()
   Modification methods
    +   insert(listDataType) : void               // default inserts new ListItem to tail
    +   insertToHead(listDataType) : void
    +   insertToMid(listDataType) : void
    +   delete(listDataType) : void
    +   search(listDataType) : bool
    +   sortAsc() : void                          // selection sort
    +   sortDesc() : void
    +   getNext() : listDataType
    +   hasNext() : bool
   Constant methods
    +   start() : listDataType
   Overloaded Operators
    +   operator=(const List&)    // use deep copy - do NOT copy pointers! also may remove...
   Member variables
    -   headPtr : ListItem*
    -   tailPtr : ListItem*
    -   size : int
    -   isSorted : bool
endclass
```

Association: List ----- 1 : m ------ contains>>ListItem

```
INCLUDE List header

class Stack
   Typedef
    +   typedef stackDataType : List::listDataType  // depends on data type of List
   Constructors
    +   Stack()
    +   Stack(stackDataType )
    +   Stack(const Stack&)              // use deep copy - do NOT copy pointers!
   Destructor
    +   (auto call base?)
   Modification methods
    +   push(stackDataType)
    +   pop() : stackDataType
   Constant methods
    +   isEmpty() : bool
    +   showTop() : stackDataType
endclass
```

Association: Stack----- 1 : 1 ------ inherits from>>List

```
INCLUDE List header

class Queue
   Typedef
    +   typedef queueDataType : List::listDataType  // depends on data type of List
   Constructors
    +   Queue()
    +   Queue(queueDataType)
    +   Queue(const Queue&)                    // use deep copy - do NOT copy pointers!
   Destructor
    +   (auto call base?)
   Modification methods
    +   enqueue(queueDataType )
    +   dequeue() : queueDataType
    +   sortAsc() : void
    +   sortDesc() : void
   Constant methods
    +   isEmpty() : bool
    +   search(queueDataType) : bool
endclass
```

Association: Queue----- 1 : 1 ------ inherits from>>List