**Project: DU (Developer Utility) like STL**

**namespace { util }**

**dev\_util**

**.hpp**

**Class (Container)**

**ADT (Abstract Data Type)**

* **Dynamic Arrays**
* **Linked List**
* **Doubly Linked List**
* **Circular Linked List**
* **Circular Doubly Linked List**
* **STACK**
* **Two Way STACK**
* **QUEUE**
* **Tree**

**.cpp**

**dev\_algo**

**dev\_util**

**.lib**

**.lib**

**.lib**

**.lib**

**.lib**

**.lib**

**.lib**

**.lib**

**Array (ArrayList)**

* ArrayList::createArray (capacity);
* ArrayList::indexOf(item, start, end); //start, and end optional
* ArrayList::bin\_searcht(X);
* ArrayList::bubble\_sort(); //ASC or DESC as an argument
* ArrayList::revList();
* ArrayList::insert\_front(item);
* ArrayList::insert\_back(item);
* ArrayList::insertAt(indexPos, item);
* ArrayList::fillItem(item);
* ArrayList::extend(anotherList);
* ArrayList::update(indexPos, item);
* ArrayList::[indexPos];
* ArrayList::indexAt(indexPos);
* ArrayList::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* ArrayList::endItem(); //editable also like subscript operator arr[0]=5; same like endItem()=5;
* ArrayList::isListEmpty();
* ArrayList:: isListFull();
* ArrayList::del(start, end); //start Index Position is necessary
* ArrayList::pop(indexPos);
* ArrayList::removeItem(item);
* ArrayList::remove\_front();
* ArrayList::remove\_back();
* ArrayList::removeAll(item); //will remove all the duplicate items
* ArrayList::release(); //release the current memory which you have created
* ArrayList::countItems();
* ArrayList::isEqualTo(anotherList); //compares two List returns true(1) if equal otherwise false(0)
* ArrayList::totalMemCapacity();
* ArrayList::viewList();

**List (LinkedList)**

* LinkedList::indexOf(item, start, end); //start, and end optional
* LinkedList::bin\_searcht(X);
* LinkedList::revList();
* LinkedList::insert\_front(item);
* LinkedList::insert\_back(item);
* LinkedList::insertAt(indexPos, item);
* LinkedList::del(start, end); //start Index Position is necessary
* LinkedList::pop(indexPos);
* LinkedList::removeItem(item);
* LinkedList::remove\_front();
* LinkedList::remove\_back();
* LinkedList::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* LinkedList::endItem(); //editable also like subscript operator arr[0]=5; same like endItem()=5;
* LinkedList::isListEmpty(); //returns true(1) if empty otherwise false(0)
* LinkedList::countItems();
* LinkedList::fillItem(item);
* LinkedList::extend(anotherList);
* LinkedList::[indexPos];
* LinkedList::viewList();

**List (DLL)**

* DLL::indexOf(item, start, end); //start, and end optional
* DLL::bin\_searcht(X);
* DLL::revList();
* DLL::insert\_front(item);
* DLL::insert\_back(item);
* DLL::insertAt(indexPos, item);
* DLL::del(start, end); //start Index Position is necessary
* DLL::pop(indexPos);
* DLL::removeItem(item);
* DLL::remove\_front();
* DLL::remove\_back();
* DLL::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* DLL::endItem(); //editable also like subscript operator arr[0]=5; same like endItem()=5;
* DLL::isListEmpty(); //returns true(1) if empty otherwise false(0)
* DLL::countItems();
* DLL::fillItem(item);
* DLL::extend(anotherList);
* DLL::[indexPos];
* DLL::viewList();

**List (CLL)**

* CLL::indexOf(item, start, end); //start, and end optional
* CLL::bin\_searcht(X);
* CLL::insert\_front(item);
* CLL::insert\_back(item);
* CLL::insertAt(indexPos, item);
* CLL::del(start, end); //start Index Position is necessary
* CLL::pop(indexPos);
* CLL::removeItem(item);
* CLL::remove\_front();
* CLL::remove\_back();
* CLL::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* CLL::endItem(); //editable also like subscript operator arr[0]=5; same like endItem()=5;
* CLL::isListEmpty(); //returns true(1) if empty otherwise false(0)
* CLL::countItems();
* CLL::fillItem(item);
* CLL::extend(anotherList);
* CLL::[indexPos];
* CLL::viewList();

**List (CDLL)**

* CDLL::indexOf(item, start, end); //start, and end optional
* CDLL::bin\_searcht(X);
* CDLL::insert\_front(item);
* CDLL::insert\_back(item);
* CDLL::insertAt(indexPos, item);
* CDLL::del(start, end); //start Index Position is necessary
* CDLL::pop(indexPos);
* CDLL::removeItem(item);
* CDLL::remove\_front();
* CDLL::remove\_back();
* CDLL::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* CDLL::endItem(); //editable also like subscript operator arr[0]=5; same like endItem()=5;
* CDLL::isListEmpty(); //returns true(1) if empty otherwise false(0)
* CDLL::countItems();
* CDLL::fillItem(item);
* CDLL::extend(anotherList);
* CDLL::[indexPos];
* CDLL::viewList();

**List (pairList)**

* pairList::append(item);
* pairList::remove\_front()
* pairList::remove\_back ();
* pairList::removeFrom();
* pairList::first(); //editable also like subscript operator arr[0]=5; same like first()=5;
* pairList::second(); //editable also like subscript operator arr[0]=5; same like second="Hello";
* pairList::extend(anotherList);
* pairList::countItems();

**List (StackArray)**

* StackArray::createStack (capacity);
* StackArray::push(item);
* StackArray::pop();
* StackArray::peek(); //editable also like subscript operator arr[0]=5; same like peek()=5;
* StackArray::extend(anotherStackArray);
* StackArray::release();
* StackArray::isStackEmpty();
* StackArray::isStackFull();
* StackArray::release();
* StackArray::countItems();
* StackArray::viewList();

**List (StackList)**

* StackList::push(item);
* StackList::pop();
* StackList::peek(); //editable also like subscript operator arr[0]=5; same like
* StackList::extend(anotherStackList);
* StackList::isStackEmpty();
* StackList::countItems();
* StackList::viewList();

**List (TwoWayStack)**

* TwoWayStack::createStack (capacity);
* TwoWayStack::push\_left(item);
* TwoWayStack::push\_right(item);
* TwoWayStack::pop\_left();
* TwoWayStack::pop\_right();
* TwoWayStack::peek\_left(); //editable also like subscript operator arr[0]=5; same like peek\_left()=5;
* TwoWayStack::peek\_right(); //editable also like subscript operator arr[0]=5; same like peek\_right()=5;
* TwoWayStack::extendToLeft(anotherStackList);
* TwoWayStack::extendToRight(anotherStackList);
* TwoWayStack::release();
* TwoWayStack::isEmptyLeft();
* TwoWayStack::isEmptyRight();
* TwoWayStack::isStackFull();
* TwoWayStack::countItems();
* TwoWayStack::viewList();

**List (QueueArray)**

* QueueArray::createQueue (capacity);
* QueueArray::insertion(item);
* QueueArray::deletion();
* QueueArray::isQueueEmpty();
* QueueArray::isQueueFull();
* QueueArray::peek(); //editable also like subscript operator arr[0]=5; same like peek()=5;
* QueueArray::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* QueueArray::endItem(); //editable also like subscript operator arr[0]=5; same like backItem()=5;
* QueueArray::extend(anotherQueueArray);
* QueueArray::release();
* QueueArray::countItems();

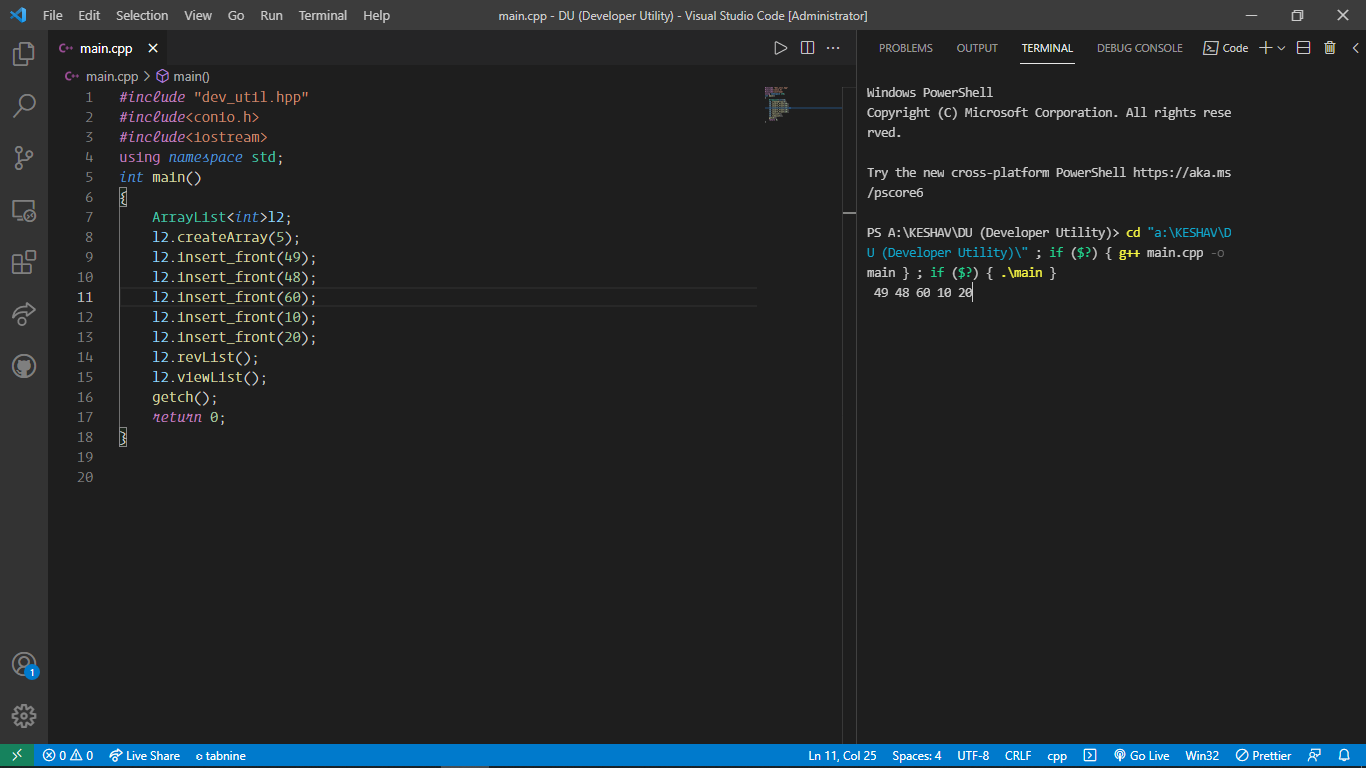
**List (QueueList)**

* QueueList::insertion(item);
* QueueList::deletion();
* QueueList::isQueueEmpty();
* QueueList::peek(); //editable also like subscript operator arr[0]=5; same like peek()=5;
* QueueList::frontItem(); //editable also like subscript operator arr[0]=5; same like frontItem()=5;
* QueueList::endItem(); //editable also like subscript operator arr[0]=5; same like backItem()=5;
* QueueList::extend(anotherList);
* QueueLinkedList::countItems();

**How to use?**

First of all one of the given files is a file named ***dev\_util.hpp***, you install it in a directory and then in the CPP program, while doing it, you just include ***dev\_util.hpp*** by putting double quotes like this #include "dev\_util.hpp"

***DEMO:-***

******