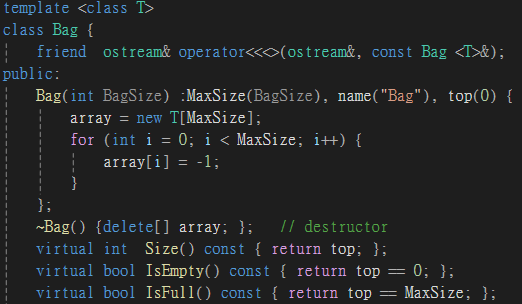
Data Structure Homework 4

B1100731 潘永牧

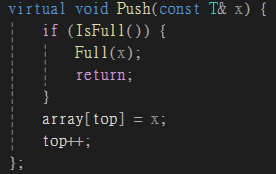
本次作業目標是實現bag、stack、queue和double-ended queue，class以template的方式來增加可重複利用性，另外stack和queue用了繼承bag的方式來省去重複的程式碼。

1. Bag

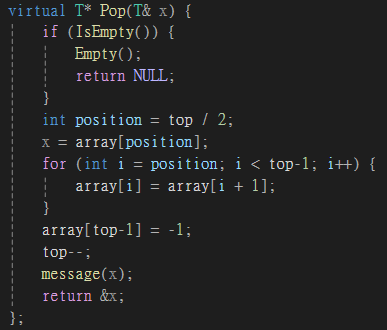
* 在建構的指令當中，先建構了一個MaxSize的Array，然後將裡面每一項定義為-1。而Size函數做的是回傳目前有效大小。IsEmpty函數判斷有效大小是否為0。IsFull函數判斷有效大小是否已達最大值。



* Push是用來插入元素的，所以當有效大小達最大值時，就顯示無法再插入，並且結束函式。如果還有空間那就插入到陣列最後一項，然後top加一。



* 在Bag中固定Pop出去的是陣列中間的元素，如果陣列是空的那麼就顯示訊息並結束。如果有東西，就找到列中間的元素存起來，然後將所有元素向前移，覆蓋被Pop掉的元素。

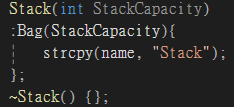


* ReadMe的輸出定義message的訊息，表示哪一個元素被Pop出來了。

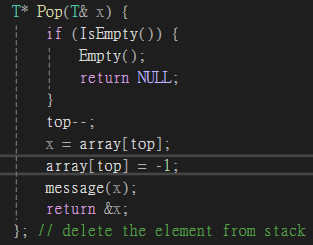


1. Stack

* 建構函數中呼叫Bag的建構函數，傳入有效最大值，並改寫name。

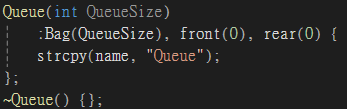


* 改寫Bag的Pop，(因為Push方式都是一樣的，所以不須改寫)，Stack永遠都是Pop最後一個元素，top先減一，就能找最後一項的index，存取Pop的數字，並改為-1。

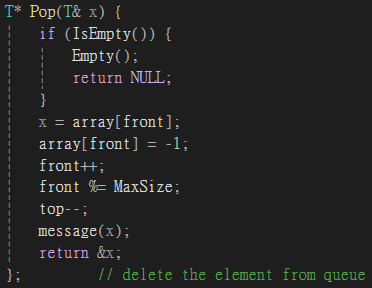


1. Queue

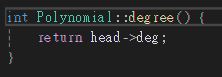
* 建構函數呼叫Bag的建構函數，改寫name，並多兩個初始值為0變數front與rear。我改善Queue只能存n-1問題的方式是，讓top儲存元素的數量，然後front指到第一個元素的位置，rear指到下一次Push要插入的位置。



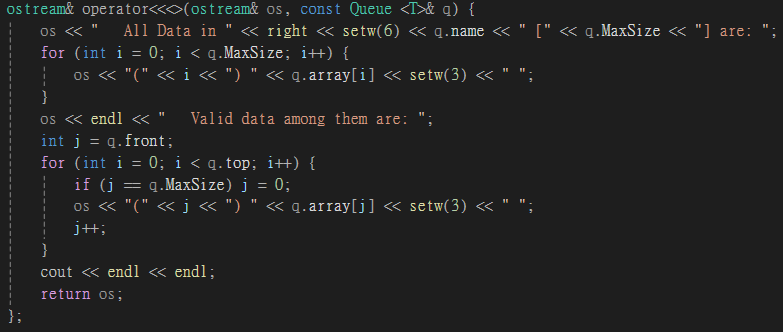
* 改寫Pop，Queue永遠Pop陣列最前面的元素，將[front]的元素儲存起來，並改為-1，front往後指一格，如果front已經是最後一個那就指回0。



* 改寫Push，Queue永遠Push在陣列最後面，所以插入在rear的位置，rear加一，如果rear為最後一個那就指回0。



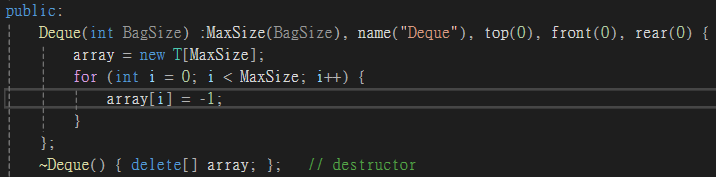
* 改寫 <<，從front開始印top個元素。



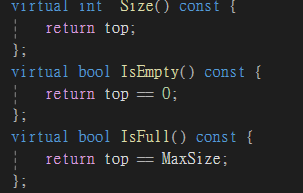
1. Deque

沒有繼承Bag因為Deque沒有pop和push這兩個函式。

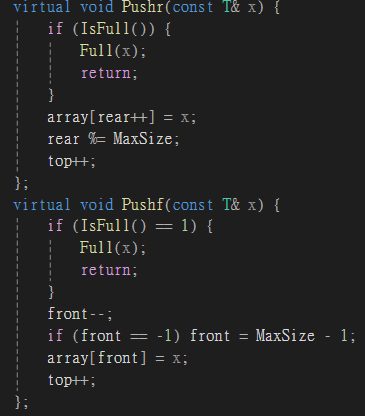
* 建構函式與bag相同只是多了front和rear兩個變數。



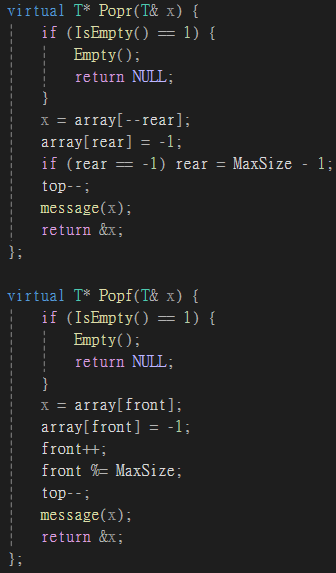
* Size、IsEmpty和IsFull也與Bag一樣。



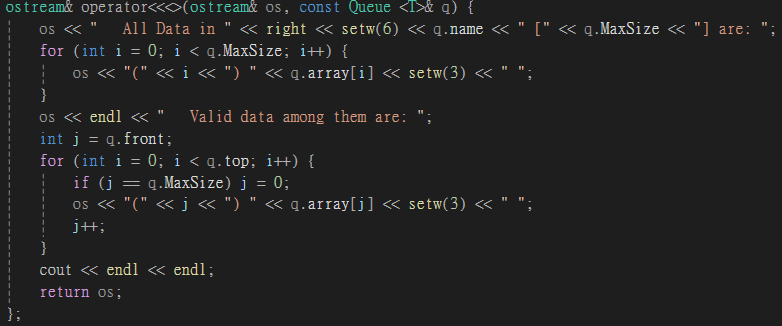
* Pushr與Queue的Push一樣，Pushf則是在反方向新增一個元素。



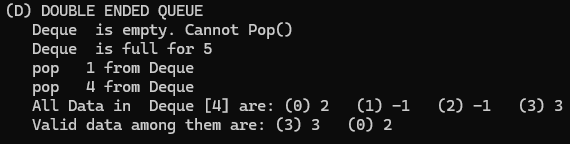
* Popf與Queue的Pop一樣，Popr則是刪除最後一個元素。



* 改寫 <<，與queue相同。



* 輸出結果與老師提供的ReadMe有些不同，我的程式設計Pushf是方向相反的Push，而現在這個queue是一個頭接尾的圓形，所以我的第一個元素插在了(3)的位置，而不是向老師解答的(0)。如果第一個指令是Pushr那麼會插在(0)的位置，但如果是Pushf的話我覺得應該插在最後一個位置，我這樣想不知道有沒有錯。



心得:

此次作業也沒有時麼比較突出的優化，但在寫程式的過程中有發現一些自己打的判斷式有些多餘，像是IsEmpty()我原本打成{return top==0?1:0;}，後來發現直接{retrun top == 0;}的意思跟根本一樣效率也比較好；在改寫queue的<<時，我原本是想說用一個變數儲存front的值，然後cout完一個元素就加一，一直到變數等於rear的時候停止，但後來想到還有top這個變數可以用，後來就改成了從front開始cout top個元素，這樣可以減少程式的出錯。