#ifndef PQ\_lib

#define PQ\_lib

struct HeapType ///Heap\_t

{

void \*elemts; // element array

int numElemts; // number of elements，可以用來得知陣列是滿的或是空的

};

typedef struct HeapType Heap\_t;

struct PQ ///PQ\_t

{

H\_class pqClass;　///辨別是ＭaxHeap或MinHeap

Heap\_t heap; ///heap的頭，即Priority的頭

int maxSize, eleSize; ///maxSize：陣列最大可存放的大小。eleSize：使用者使用結構的大小。

int (\*cp)(void \*eleA, void \*eleB);//結構裡面可以有函式指標，用來比較使用者想比較的數值。

};

typedef struct PQ PQ\_t;

**以上為使用者可使用 Heap\_t 、 PQ\_t 的結構。**

**若使用此library，需先呼叫此函式建立Priority Queue，做Priority Queue的初始化**

void createPQ( PQ\_t \*pq , H\_class pqClass, int eleSize, int maxSize, int (\*cp)(void \*eleA, void \*eleB));

傳入值：  
第一個參數：型態為PQ\_t，Priority Queue頭的位址。

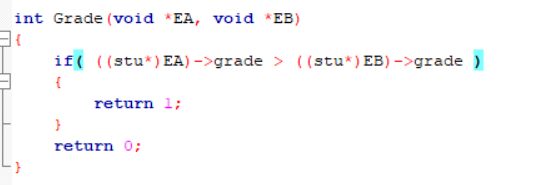
第二個參數：型態為H\_class，若想建構maxHeap、傳入MAXHEAP；建構minaxHeap、傳入MINHEAP。

第三個參數：型態為int，使用者使用基本元素的大小。

第四個參數：型態為int，使用者想用多長的priority queue的陣列大小。

第五個參數：函式指標。

　　　　　　需使用者寫一個可以比較兩個node的函式並傳入，如：



（以上為欲比較學生成績的函式）(stu為該程式所使用的結構)  
 而需要其傳回的結果符合以下幾點

(1)當EA的值 > EB的值，回傳1  
 (2)當EA的值 <= EB的值，回傳0

回傳值：無回傳值

**可使用的功能：**

（１）辨別此Priority Queue是不是空的

int IsEmpty(PQ\_t \*pq); /\* return 0: not empty, 1: empty\*/

傳入值：型態為PQ\_t，Priority Queue頭的位址。

回傳值：空的、回傳１；不是空的、回傳０。

（２）刪除BST中，與第一個傳入參數相等的值

int IsFull(PQ\_t \*pq); /\* return 0: not full, 1:full \*/

傳入值：型態為PQ\_t，Priority Queue頭的位址。

回傳值：滿的、回傳１；不是滿的、回傳０。

（３）加入一個element到PQ

int Enqueue(PQ\_t \*pq, void\* eleA); /\* add an element into PQ \*/

第一個傳入值：型態為PQ\_t，Priority Queue頭的位址。

第二個傳入值：型態為void，想要加入PQ的element的位址。

回傳值：型態為int，加入成功，回傳１；加入失敗（Priority Queue滿的狀態），回傳０。

（４）從PQ中刪除一個element

void \*Dequeue(PQ\_t \*pq); /\*delete an element from PQ \*/

傳入值：型態為PQ\_t，Priority Queue頭的位址。

回傳值：型態為void指標，該指標指向刪除掉的節點。

#endif