# 410921202 資工三 林芷萱 DS HW3

## 函式清單及說明：

資料夾內容：

---

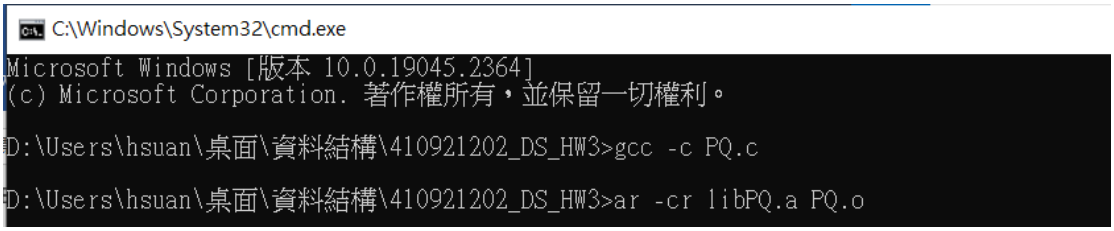
```
410921202_DS_HW3/
|---函式庫使用說明.pdf
|---bin
|     |---Debug
|           |---test.exe
|---include
|     |---PQ.h
|---lib
|     |---libPQ.a
|---obj
|     |---Debug
|           |---PQ.o
|---src
|     |---PQ.c
|     |---student.c
|---test
|     |---test.c
```

藍字為資料夾 **黑色粗體字為檔案**

---

在這個作業裡我寫了 library PQ、在 student.c 裡設計了一個用來存取 student 資料的節點，並在 test.c 裡使用 library 跟 student.c 的結構來做測試。

使用 gcc 靜態連結函式庫：



# library PQ 說明：

在這個作業裡我自己設計出的 library，用於實作 HW3 的 PQ，裡面共有 5 個 function，以下是對於設計出的 library 內 function 的介紹：

---

## 第一個 function createPQ：

他用於建立一個 PQ。使用者呼叫這個 function 的方式為：

**createPQ (PQ, PQ 的種類, 元素大小, 最大元素個數, 使用者自用 function)**

使用者要使用這個 function 時還需另外設計自己的 function 以完成他想做的工作。

**使用範例：**

我在 student.c 裡設計了一個用來存取 student 資料的節點，並在 test.c 中做測試：

```
student_t node[6]=
{
    {"C120308001", 70, 100},
    {"B220406001", 60, 90},
    {"D120306001", 80, 95},
    {"A220407001", 65, 90},
    {"D220506001", 10, 70},
    {"A120406001", 90, 90}
};

PQ_t maxPQ;
createPQ(&maxPQ, MAXHEAP, sizeof(student_t), 100, compareMath);
```

---

## 第二個 function 是 Enqueue：

他用於將元素加入 PQ 中。使用者呼叫這個 function 的方式為：
**Enqueue(PQ, 元素)**

**使用範例：**
我在 student.c 裡設計了一個用來存取 student 資料的節點，並在 test.c 中做測試：

```
printf("----------------------\n");
printf("add student_t node into maxPQ:\n");
int i;
for(i=0; i<6; i++)
    Enqueue(&maxPQ, &node[i]);

print(&maxPQ);
```

**測試結果：**

```
add student_t node into maxPQ:
index=0, ID=A120406001,math=90, eng=90
index=1, ID=A220407001,math=65, eng=90
index=2, ID=D120306001,math=80, eng=95
index=3, ID=B220406001,math=60, eng=90
index=4, ID=D220506001,math=10, eng=70
index=5, ID=C120308001,math=70, eng=100
```

## 第三個 function 是 IsEmpty：

他用於檢查 PQ 是否為空，是的話回傳 1，否則回傳 0。
使用者呼叫這個 function 的方式為：**IsEmpty (PQ)**

**使用範例：**

```
printf("IsEmpty: %d\n", IsEmpty(&maxPQ));
```

**測試結果:**

```
IsEmpty: 1
```

## 第四個 function 是：IsFull：

他用於檢查 PQ 是否為滿，是的話回傳 1，否則回傳 0。
使用者呼叫這個 function 的方式為：**IsFull (PQ)**

**使用範例：**

```
printf("IsFull: %d\n", IsFull(&maxPQ));
```

**測試結果：**

```
IsFull: 0
```

## 第五個 function 是 Dequeue：

他用於將元素從 PQ 中刪除，使用者呼叫這個 function 的方式為：**Dequeue(PQ)**

**使用範例：**

```
printf("-----------------------\n");
printf("after Dequeue node :\n");
Dequeue(&maxPQ);
print(&maxPQ);
```

**測試結果：**

```
-----------------------
after Dequeue node :
index=0, ID=D120306001,math=80, eng=95
index=1, ID=A220407001,math=65, eng=90
index=2, ID=C120308001,math=70, eng=100
index=3, ID=B220406001,math=60, eng=90
index=4, ID=D220506001,math=10, eng=70
```

## student.c 檔案說明：

在 main.c 檔裡有兩個自用 function、一個資料結構：

```
typedef struct myElement
{
    char ID[10];
    int math;
    int eng;
} student_t;

int compareMath(void *elementA, void *elementB)
{
    int mathA = ((student_t *)elementA)->math;
    int mathB = ((student_t *)elementB)->math;
    if(mathA > mathB)
    {
        return 1;
    }
    else if(mathA < mathB)
    {
        return -1;
    }
    return 0;
}

void print(PQ_t *pq)
{
    int i;
    student_t *temp;
    for (i=0; i<pq->heap.numElements; i++)
    {
        temp = (student_t *)(pq->heap.elements+i*sizeof(student_t));
        temp->ID[10] = '\0';
        printf("index=%d, ID=%s,math=%d, eng=%d\n",i,temp->ID, temp->math, temp->eng);
    }
}
```

student_t 是用來存取學生資料的節點

compareMath 是用來比較兩元素大小的 function

print 是用來將 PQ 印出的 function

---

## 測試檔(test.c)執行總結果：

```
D:\Users\hsuan\桌面\資料結構\410921202_DS_HW3>gcc test.c libPQ.a -o test.exe

D:\Users\hsuan\桌面\資料結構\410921202_DS_HW3>test.exe
IsEmpty: 1
IsFull: 0
----------------------
add student_t node into maxPQ:
index=0, ID=A120406001,math=90, eng=90
index=1, ID=A220407001,math=65, eng=90
index=2, ID=D120306001,math=80, eng=95
index=3, ID=B220406001,math=60, eng=90
index=4, ID=D220506001,math=10, eng=70
index=5, ID=C120308001,math=70, eng=100
IsEmpty: 0
IsFull: 0
----------------------
after Dequeue node :
index=0, ID=D120306001,math=80, eng=95
index=1, ID=A220407001,math=65, eng=90
index=2, ID=C120308001,math=70, eng=100
index=3, ID=B220406001,math=60, eng=90
index=4, ID=D220506001,math=10, eng=70
```