一、規格要求，違反者以零分計！

(a) 以 Dev-C++或 Code::Blocks 編譯與成功執行的 C/C++程式碼(.cpp/.c/.h/.hpp)，要有註解。

(b) 任何一部分的程式碼都不得被偵測為抄襲。

(c) 檔名限以「**DS1ex4_組別_學號_學號**」開頭。

二、題目內容

整合下列任務在一個系統選單下，未整合、無法連續執行或沒有輸入防呆措施，都各扣 5 分。若影響任務執行，該任務以零分計。

前言：

- 佇列 Queue 可以應用在許多層面，諸如工作排程，飛機起降，點餐系統等，以印表機列印多個檔案為例，先送到的檔案先列印，其餘檔案則暫時在佇列中等候。數據機傳輸也是，網路堵塞時先暫存資料在佇列中，等網路暢通時再繼續傳送。
- 本次作業要以佇列模擬點餐系統，需排程的每筆訂單表示成四個欄位：「訂單編號」OID、「下單時刻」Arrival（第幾分鐘）、「製作耗時」Duration（多少分鐘）和「逾時時刻」Timeout（第幾分鐘），其中 Duration > 0 且 Arrival + Duration <= Timeout。
- 原始資料檔是文字檔，第一列由左至右依序為四個欄位名稱，以定位符號（'\t'）間隔，其餘的每一列各代表一筆訂單，四個欄位值均為正整數，也以定位符號（'\t'）間隔，預設是沒有排序的，檔名格式如 input401.txt、input402.txt。

（任務一）排序後另存新檔

輸入：原始資料檔。

步驟：自行撰寫**希爾排序**(shell sort)，依「下單時刻」遞增排序，多筆同時下單者則再以「訂單編號」遞增排序，排序後另存新檔，並分別測量讀檔，排序和寫檔的執行時間。

輸出：儲存排序後的新檔（檔名格式如 sort401.txt、sort402.txt），並顯示所測量的三個執行時間於螢幕上。

佇列模擬原則：（違反一項各扣 5 分）

1. 採用先進先出（FIFO）方式的佇列來紀錄一連串訂單，每條佇列的訂單由一位廚師負責製作餐點，每位廚師負責一條佇列，廚師並非閒置時（閒置時刻>「下單時刻」），訂單就先放入佇列。
2. 佇列不得使用現成的資料結構，必須自行實作成 C++ Class，而且限制每個佇列空間只能存放最多 3 筆訂單。假設：廚師處理之前無法預知（偷看）佇列內訂單的「製作耗時」。
3. 廚師閒置時（沒有新訂單或閒置時刻<=新訂單的「下單時刻」）立即取出佇列內的舊訂單處理。一旦佇列清空，廚師閒置時刻就移至新訂單的「下單時刻」或結束模擬。
4. 訂單一旦排入佇列後就不能取消或轉移到別處，其他訂單也不能插隊（not preemptive）。
5. 訂單進入佇列前，若發現佇列空間不足就予以取消，並立即寫入『取消清單』Abort List：三個欄位包括「訂單編號」、「延誤時間」Delay（多少分鐘）一律設為 0、「取消時刻」Abort

（第幾分鐘）為該訂單的「下單時刻」。

6. 廚師從佇列取出訂單時，若發現已經逾時（「逾時時刻」<=閒置時刻）就取消，也立即寫入『取消清單』:「取消時刻」為廚師從佇列取出該訂單的閒置時刻，「延誤時間」則為「取消時刻」減去該訂單的「下單時刻」(Delay=Abort - Arrival)。
7. 廚師處理完訂單之後才發現已經逾時（「逾時時刻」<閒置時刻+「製作耗時」)，就以「製作耗時」改變閒置時刻，並立即寫入『逾時清單』Timeout List：三個欄位包括「訂單編號」、「延誤時間」Delay(多少分鐘)為廚師從佇列取出訂單的閒置時刻減去「下單時刻」、「完成時刻」Departure(第幾分鐘)為「下單時刻」加上「延誤時間」和「製作耗時」(Departure = Arrival + Delay + Duration)。
8. 成功出餐且未逾時的訂單也以「製作耗時」改變閒置時刻，但無須紀錄。

（任務二）單一佇列模擬

輸入：讀入任務一已排序的訂單資料檔，存放於一個動態陣列中。

步驟：

(1) 從第一筆訂單開始模擬單一佇列排程的等候狀態(waiting state)及處理狀態(running state)，先檢查佇列空間是否不足，若是就立即寫入『取消清單』，否則就模擬出餐（以「製作耗時」改變閒置時刻）或放入佇列。

(2) 模擬所有訂單後，要再依序處理仍在佇列內等候的訂單。最後，依『取消清單』和『逾時清單』計算『總延誤時間』Total Delay 以及『失敗比例』Failure Percentage（兩份清單筆數加總佔所有訂單數的百分比），一律四捨五入至小數點後兩位。

輸出：依序將『取消清單』、『逾時清單』、『總延誤時間』及『失敗比例』寫成一個文字檔（檔名格式如 one401.txt、one402.txt）。

附加的佇列模擬原則：(違反一項各扣 5 分)

*1. 多位廚師的佇列分別編號為 1,2,...，並且擁有各自獨立的「閒置時刻」，均預設為 0，每個佇列的空間上限同樣只能存放最多 3 筆訂單。

*2. 『取消清單』和『逾時清單』的欄位都要加上一個欄位名為「廚師編號」CID，用來記載每筆訂單所對應的廚師/佇列。

（任務三）雙重佇列模擬

輸入：同任務二。

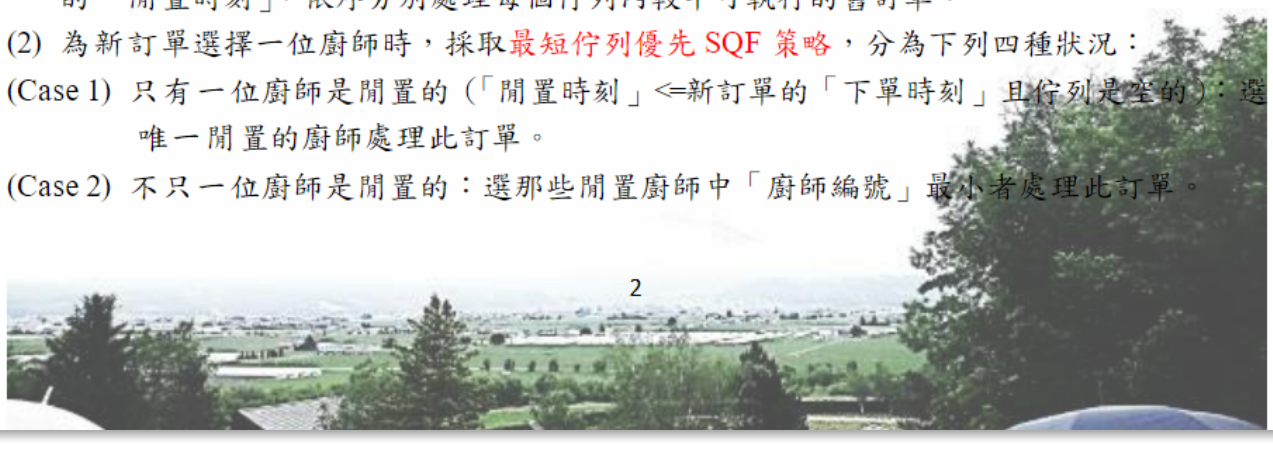
步驟：

(1) 從第一筆訂單開始模擬兩個佇列排程的等候及執行狀態，先比較「下單時刻」和各廚師的「閒置時刻」，依序分別處理每個佇列內較早可執行的舊訂單。

(2) 為新訂單選擇一位廚師時，採取最短佇列優先 SQF 策略，分為下列四種狀況：

(Case 1) 只有一位廚師是閒置的（「閒置時刻」<=新訂單的「下單時刻」且佇列是空的）：選唯一閒置的廚師處理此訂單。

(Case 2) 不只一位廚師是閒置的：選那些閒置廚師中「廚師編號」最小者處理此訂單。



(Case 3) 每位廚師都並非閒置且至少一個佇列並非全滿：選佇列長度（存放訂單筆數）最短的；若最短佇列不只一個，則選其中「廚師編號」最小者。

(Case 4) 每位廚師都並非閒置且佇列全滿：立即取消此訂單，『取消清單』的「廚師編號」記成 0 號，代表未進入佇列就被取消。

(3) 同任務二，模擬所有訂單後依序處理佇列內的訂單。最後，依『取消清單』和『逾時清單』計算『總延誤時間』以及『失敗比例』，一律四捨五入至小數點後兩位。

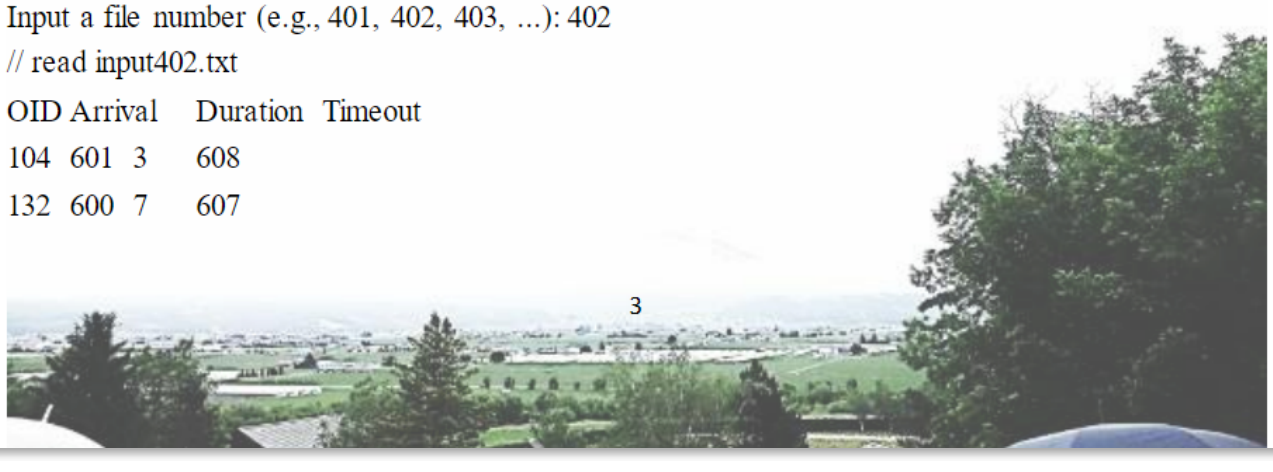
輸出：同任務二，但修改檔名格式如 two401.txt、two402.txt。

三、參考範例，格式自訂，內容不可欠缺！

（任務一）排序後另存新檔

```
Input a file number (e.g., 401, 402, 403, ...): 401
// read input401.txt
OID Arrival   Duration  Timeout   //「訂單編號」、「下單時刻」、「製作耗時」、「逾時時刻」
103 6     7     15
104 11    9     22
112 5     6     13
101 3     9     12
106 6     9     17
108 6     8     18
105 10    6     20
Reading data: 2 ms
Sorting data: 0 ms
Writing data: 3 ms
OID Arrival   Duration  Timeout   // write sort401.txt
101 3     9     12
112 5     6     13
103 6     7     15
106 6     9     17
108 6     8     18
105 10    6     20
104 11    9     22


Input a file number (e.g., 401, 402, 403, ...): 402
// read input402.txt
OID Arrival   Duration  Timeout
104 601 3     608
132 600 7     607
```

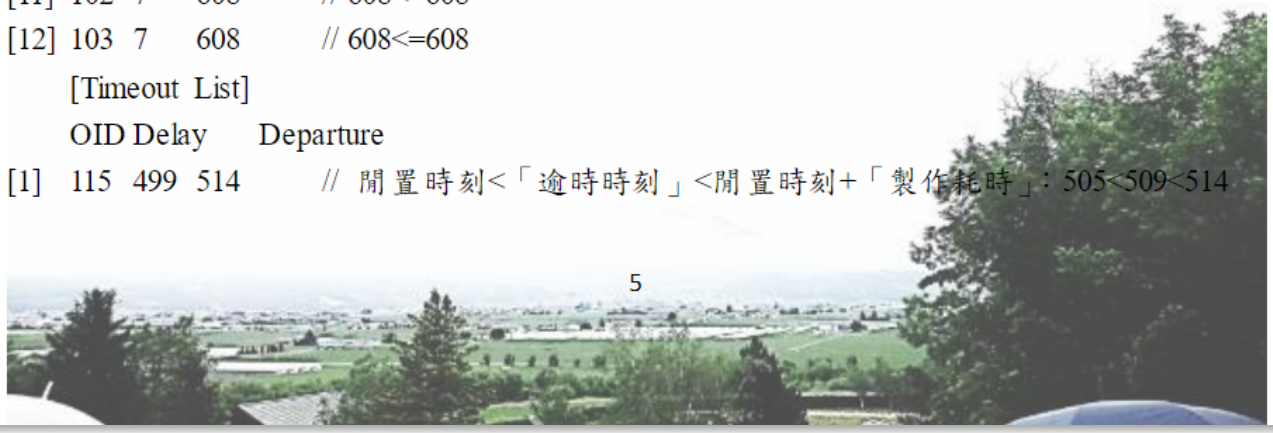


101 601 6 608
119 6 450 500
122 250 1 251
120 1 500 502
116 6 6 507
105 601 2 608
135 501 7 515
103 601 1 608
121 1 10 11
111 6 4 508
142 501 80 610
102 601 5 608
115 6 9 509
108 604 3 615
107 603 3 612
106 601 1 608
Reading data: 3 ms
Sorting data: 0 ms
Writing data: 5 ms
OID Arrival Duration Timeout // write sort402.txt
120 1 500 502
121 1 10 11
111 6 4 508
115 6 9 509
116 6 6 507
119 6 450 500
122 250 1 251
135 501 7 515
142 501 80 610
132 600 7 607
101 601 6 608
102 601 5 608
103 601 1 608
104 601 3 608
105 601 2 608
106 601 1 608
107 603 3 612
108 604 3 615

（任務二）單一佇列模擬

```
Input a file number (e.g., 401, 402, 403, ...): 401
// read sort401.txt, write one401.txt
      [Abort List]
      OID Delay     Abort          // 由左而右依序為「訂單編號」、「延誤時間」、「取消時刻」
[1]   108  0     6                 // 佇列已放滿 3 筆，故立即取消，「延誤時間」為 0
[2]   105  0     10                // 佇列已滿，立即取消
[3]   104  0     11                // 佇列已滿，立即取消
[4]   103  12    18                // 佇列的第 2 筆，發現已經逾時所以取消 (15<=18)
[5]   106  12    18                // 佇列的第 3 筆，發現已經逾時所以取消 (17<=18)
      [Timeout List]
      OID Delay     Departure      // 由左而右依序為「訂單編號」、「延誤時間」、「完成時刻」
[1]   112  7     18                // 佇列的第 1 筆，廚師處理後才發現已逾時 (12<13<18)
[Total Delay]
31 min.                           // 『總延誤時間』0+0+0+12+12+7
[Failure Percentage]
85.71 %                           // 『失敗比例』100 * 6 / 7 %

Input a file number (e.g., 401, 402, 403, ...): 402
// read sort402.txt, write one402.txt
      [Abort List]
      OID Delay     Abort
[1]   116  0     6          // 佇列已滿
[2]   119  0     6          // 佇列已滿
[3]   122  0     250        // 佇列已滿
[4]   121  500   501        // 佇列的第 1 筆，11<501
[5]   104  0     601        // 佇列已滿
[6]   105  0     601        // 佇列已滿
[7]   106  0     601        // 佇列已滿
[8]   107  0     603        // 佇列已滿
[9]   108  0     604        // 佇列已滿
[10]  101  7     608        // 「逾時時刻」<=閒置時刻：608<=608
[11]  102  7     608        // 608<=608
[12]  103  7     608        // 608<=608
      [Timeout List]
      OID Delay     Departure
[1]   115  499   514        // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：505<509<514
```



[2] 135 13 521 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：514<515<521

[3] 132 1 608 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：601<607<608

[Total Delay]

1034 min. // 0+0+0+500+0+0+0+0+0+7+7+7+499+13+1

[Failure Percentage]

83.33 % // 100 * 15 / 18 %

（任務三）雙重佇列模擬

Input a file number (e.g., 401, 402, 403, ...): 401

// read sort401.txt, write two401.txt

[Abort List]

OID CID Delay Abort //「訂單編號」、「廚師編號」、「延誤時間」、「取消時刻」

[1] 108 1 13 19 // 「逾時時刻」<=閒置時刻：18<=19

[2] 105 2 10 20 // 「逾時時刻」<=閒置時刻：20<=20

[Timeout List]

OID CID Delay Departure //「訂單編號」、「廚師編號」、「延誤時間」、「完成時刻」

[1] 106 2 5 20 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：11<17<20

[2] 103 1 6 19 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：12<15<19

[3] 104 2 9 29 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：20<22<29

[Total Delay]

43 min. // 『總延誤時間』13+10+5+6+9

[Failure Percentage]

71.43 % // 『失敗比例』100 * 5 / 7 %

Input a file number (e.g., 401, 402, 403, ...): 402

// read sort402.txt, write two402.txt

[Abort List]

OID CID Delay Abort

[1] 122 2 220 470 // 「逾時時刻」<=閒置時刻：251<=470

[2] 108 0 0 604 // 兩個佇列皆滿，「廚師編號」設為 0 號

[3] 105 2 7 608 // 「逾時時刻」<=閒置時刻：608<=608

[4] 104 1 11 612 // 608<=612

[5] 106 1 11 612 // 608<=612

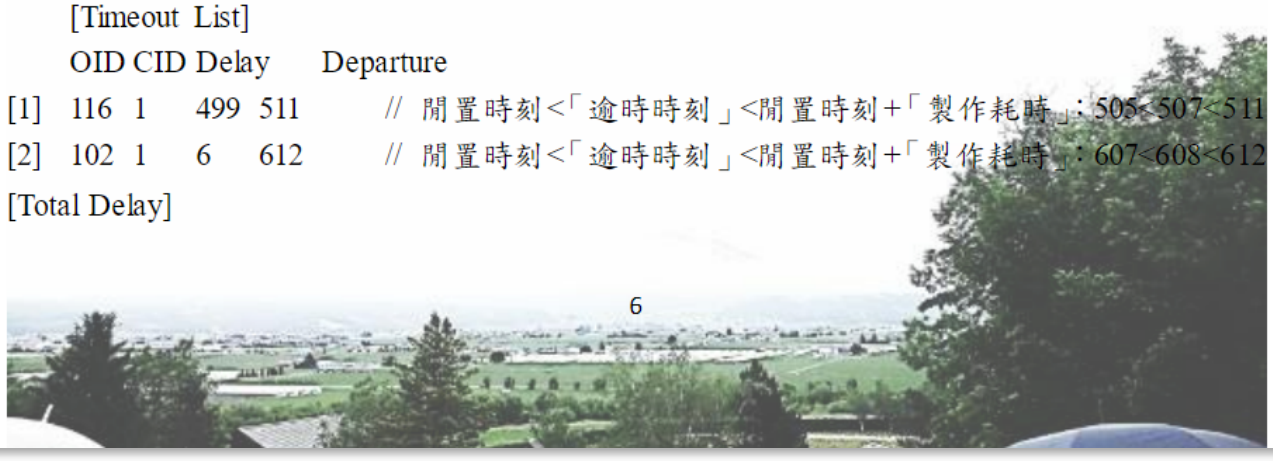
[Timeout List]

OID CID Delay Departure

[1] 116 1 499 511 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：505<507<511

[2] 102 1 6 612 // 閒置時刻<「逾時時刻」<閒置時刻+「製作耗時」：607<608<612

[Total Delay]



754 min. // 220+7+11+11+499+6

[Failure Percentage]

38.89 % // 100 * 7 / 18 %

四、繳交項目

(a) 作業程式碼：上機前一晚上傳至/評量區/作業/作業#4 程式碼/，缺交者於二日內補繳。

(b) 作業流程圖：上機結束前上傳至/評量區/同儕互評/作業#4 流程圖/，不接受補繳！

五、評分配置 60%

(a) 作業程式碼：每項任務的各佔 15 分，一個錯扣 5 分，更多錯就以零分計。

(b) 作業流程圖：每項任務的的各佔 5 分，一個錯就以零分計。

六、偵測抄襲

(a) 嚴禁抄襲網路上或相關課程的舊程式碼，老師提供或重修生自己以前寫的程式碼除外。

(b) 一旦偵測程式、助教、和老師均認定抄襲，即使是一小部分的程式碼，一律以零分計。