一、規格要求，違反者以零分計！

(1) **課程指定環境**成功編譯與執行的 C++程式原始碼，力求和**範例程式**的輸入輸出**一致**！。

(2) 任何一部分程式碼不得被判定為疑似抄襲，程式碼第一列要清楚註解二人的**學號姓名**。

(3) 檔名限以「**DS1HW4_組號__學號__學號**」開頭，同組限一人繳交單一檔案的.cpp 原始碼。

- 同組只限其中一位同學繳交一份程式碼，建議採用結對程式設計與 GitHub 進行協作。
- 同組只限其中一位同學可以使用 DC 帳號進入程式檢測系統，下次作業將換成另一人。

二、作業內容

整合下列任務於**單一程式**，並遵循範例程式的輸入輸出介面，未整合、無法連續執行或沒有完整的輸入防呆措施，都至少扣 5 分。若導致程式檢測系統無法正常運作，該任務以零分計。

必須(而且只限)出現在任務選單第一列：***** (^_^) Data Structure (^o^) *****

主題：以自行設計的佇列類別(C++ class)撰寫一套模擬點餐系統。

- 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。

必須遵守的原則：(每個任務違反一項各扣 5 分)

(1) 須採用動態配置空間的動態陣列儲存從檔案讀入的資料，此處禁用靜態陣列。

(2) 須"自行實作"佇列類別(C++ class)，禁用 std 標準函式庫提供類似功能的現成資料結構。

(3) 任務二成功讀入排序檔後，該動態陣列會被反覆用於任務三和任務四，不得重新讀檔。

(任務一) 以指定方法另建排序檔

資料格式：

1. 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，**訂單數值的假設**為 Duration > 0 且 Arrival + Duration <= Timeout。
2. 訂單資料是文字檔，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，四個欄位值都是正整數，也以定位符號('/t')間隔，預設沒有排序，檔名格式如 input401.txt、input402.txt。

輸入：以一個動態陣列讀入指定編號的原始資料檔。

步驟：

(1) 依照讀入次序將存放在動態陣列的每筆原始資料輸出到螢幕上。

(2) 採用插入式作法(insertion method)自行實作**希爾排序**(shell sort)，依「下單時刻」遞增排序，多筆下單時刻相同再以「訂單編號」遞增排序，排序後另存新檔，分別測量讀檔，排序和寫檔的執行時間，時間採用微秒(microsecond, 縮寫 us)為單位，輸出到螢幕上。

- 希爾排序限用插入式作法，禁用原始交換式作法(original exchange method)！

輸出：

(1) 依照檔案原始次序輸出資料到螢幕上。

(2) 排序後的資料另存新檔，檔名如 sorted401.txt、sorted402.txt。若重複執行同一個指定編號，覆寫同一個檔案。

(3) 依序顯示以微秒(us)為單位測量的三個執行時間於螢幕上。

（任務二）單一佇列模擬

佇列模擬原則：（違反一項各扣 5 分）

- 違反**訂單數值的假設**的資料只需要忽略不處理即可！

1. 一位廚師擁有一條先進先出（FIFO）的佇列，廚師製作餐點時（閒置時刻>「下單時刻」）訂單依序暫存在佇列，不得變換至其他廚師的佇列，也不允許插隊（non-preemptive）。
2. 一條佇列可存放至多 3 筆訂單，廚師完成餐點之前無法預知（偷看）佇列內訂單的「製作耗時」，廚師完成前一個餐點（閒置時刻<=「下單時刻」）方可取出佇列下一筆訂單。
3. 一旦佇列清空，廚師閒置時刻就移至下一筆訂單的「下單時刻」，無訂單則可結束模擬。
4. 無法執行的訂單要記載於『取消清單』Abort List，三個欄位包括「訂單編號」、「取消時刻」Abort（第幾分鐘）、「延誤時間」Delay（多少分鐘），符合下列條件即取消之：
   (1) 訂單剛抵達時佇列已滿：「取消時刻」為該訂單的「下單時刻」(Abort = Arrival)，「延誤時間」一律設為 0。
   (2) 從佇列取出訂單時發現逾時（**「逾時時刻」<閒置時刻**）：「取消時刻」為取出該訂單的閒置時刻，「延誤時間」為「取消時刻」減去該訂單的「下單時刻」(Delay = Abort - Arrival)。
5. 廚師完成餐點時才發現逾時（**「逾時時刻」<閒置時刻+「製作耗時」**）：
   (1) 將閒置時刻加上「製作耗時」，即為新的閒置時刻。
   (2) 寫入『逾時清單』Timeout List：三個欄位包括「訂單編號」、「完成時刻」Departure（第幾分鐘）為新的閒置時刻、「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」。
6. 成功出餐而且未逾時的訂單也以「製作耗時」更新閒置時刻，但無須紀錄下來。
7. 舊訂單的餐點完成時刻或逾時時刻恰好等於新訂單的「下單時刻」時，一律先移除舊訂單並且假設處理耗時為 0，以確保新訂單不會因空間不足而遭到取消。

輸入：讀入指定編號的排序檔如 sorted401.txt，內容存放於一個動態配置空間的動態陣列。

步驟：

(1) 從動態陣列的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) 未被取消的訂單就模擬被處理或暫存於佇列等候，完成餐點時才發現逾時的訂單寫入『逾時清單』。模擬完所有訂單後，再依序處理仍在佇列內等候的訂單。

(3) 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，一律四捨五入至小數點後兩位。

- 『總延誤時間』為兩份清單內所有訂單的「延誤時間」加總(單位：分鐘)。
- 『失敗比例』為兩份清單的訂單數加總佔所有訂單數的百分比(單位：%)。

輸出：

(1) 依照排序檔的次序輸出資料到螢幕上。

(2) 依序將『取消清單』、『逾時清單』、『總延誤時間』及『失敗比例』紀錄成文字檔，另存新檔的名稱前綴改為 one，例如 one401.txt、one402.txt。重複執行同一個指定編號，覆寫同一個檔案。

（任務三）雙重佇列模擬

附加的佇列模擬原則：(違反一項各扣 5 分)

8. 二位廚師的佇列分別賦予 1 號及 2 號，各自的「閒置時刻」均預設為 0，二個佇列的空間上限同樣只能各自存放最多 3 筆訂單。
9. 『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。

輸入：**(重複使用)**任務二讀入排序檔所建立的動態陣列。

步驟：

(1) 從動態陣列首筆訂單開始模擬二個佇列的等候狀態及二位廚師的執行狀態，比較「下單時刻」和二位廚師的「閒置時刻」，依序分別處理每個佇列內比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) 只有一位廚師閒置（「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) 二位廚師都是閒置：選 1 號廚師處理此訂單。

(Case 3) 二位廚師都並非閒置且一個佇列並非全滿：選佇列長度（存放訂單筆數）比較短的；若二個佇列長度相等，則選 1 號廚師的佇列。

(Case 4) 二位廚師都並非閒置且二個佇列全滿：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) 遵循任務二的步驟，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：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，另存新檔的名稱前綴改為 two，例如 two401.txt、two402.txt。若重複執行同一個指定編號，覆寫同一個檔案。

（任務四）多重佇列模擬

附加的佇列模擬原則：(違反一項各扣 5 分)

10. 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。

輸入：**(重複使用)**任務二讀入排序檔所建立的動態陣列、額外輸入的一個正整數 N。

步驟：

(1) 從動態陣列首筆訂單開始模擬 N 個佇列的等候狀態及 N 位廚師的執行狀態，比較「下單時刻」和 N 位廚師的「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) 遵循任務二的步驟，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：

(1) 若 N>2，依序將『取消清單』、『逾時清單』、『總延誤時間』及『失敗比例』存入文字檔，新檔名的前綴改為 any，例如 any401.txt、any402.txt。

(2) 若 N=1，檔名前綴沿用任務二的 one，例如 one401.txt；若 N=2，檔名前綴沿用任務三的 two，例如 two401.txt。若重複執行同一指定編號及相同 N 值，覆寫同一個檔案。

程式碼：期限前每組上傳一份單一檔案的.cpp 原始碼至 **i-learning 2.0** 系統，不得多繳。

貼文：期限前至 **i-learning 2.0** 系統張貼流程圖與簡報網址的一篇貼文，須保持公開到學期末。

三、評分項目

繳交項目：(A) + (B) + (C) = 15*4 + 20 + 20 = 100 分

(A) 程式碼：**四項任務**每項各佔 15 分，1 個錯扣 5 分，更多錯就以零分計。

(B) 貼文：流**程圖與簡報**每項各佔 10 分，1 個錯扣 5 分，更多錯以零分計。

(C) 機測：時限內正確回答**方法原理**或**程式寫法**的提問，共佔 20 分。

四、評分流程

(1) 每項任務以公開和隱藏測資檢測正確性與效率，必須嚴格遵循**範例**程式的輸入輸出！

(2) 機測節次安排在機測前一天公告，**機測缺席視為放棄**，該次作業成績以**零分**計！

五、偵測抄襲

(1) 嚴禁抄襲網路上或相關課程的程式碼，老師教材提供或重修生自己寫的程式碼除外。

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