程式設計（112-1）

作業二

作業設計：孔令傑
國立臺灣大學資訊管理學系

繳交作業時，請至 PDOGS（`http://pdogs.ntu.im/`）爲第一、二、三、四題各上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳）。每位學生都要上傳自己寫的解答。不接受紙本繳交；不接受遲交。這份作業的截止時間是 ***9 月 19 日早上八點***。爲這份作業設計測試資料並且提供解答的助教是林元婷。

在你開始前，請閱讀課本的第 6.1–6.4 節（關於陣列）、第五章（關於函數）[1]。第 6.7 和 6.8 節也有幫助。

本次作業滿分爲 110 分，得幾分就算幾分。若整學期有 n 份作業，則學期的作業總成績即爲 n 份作業的總分除以 n（不論超過 100 與否）。

第一題

（20 分）接續作業一的「開心觀光巴士」相關問題，我們仍然給定了一條固定的路線，不包含起點站總共有 n 站。已知在起點站有 w 位乘客上車，而接下來的第 i 站有 y_i 位乘客下車，然後會有 x_i 位乘客上車，並且在終點站時，所有乘客都必須要下車。在每一站，都是所有要下車的乘客下車後，要上車的乘客才上車。跟作業一不同的是，這次我們多新增了巴士的人數上限爲 K 人。已知 $w \le K$。

「開心觀光巴士」營運一段時間後，由於民眾上下車時有時候會有漏刷卡、多刷卡等各種狀況，而「開心觀光巴士」的資訊系統也不是太理想，因此可能會留下不合理的紀錄，讓公司員工在檢查過往乘車紀錄時，可能會看到兩種的不合理現象，第一種爲「下車人數大於車上人數」，也就是找到一個車站 k 滿足

$$y_k > w + \sum_{j=1}^{k-1} x_j - \sum_{j=1}^{k-1} y_j ;$$

第二種則是「車上人數大於載客人數上限」，也就是找到一個車站 k 滿足

$$K < w + \sum_{j=1}^{k} x_j - \sum_{j=1}^{k} y_j 。$$

本題會給定一組巴士的乘車紀錄，請幫助公司同仁檢查是否有發生以上任一種不合理現象，有的話，則找出是在哪一站發生並輸出該站的編號。如果這班車發生了複數次不合理現象，也只需要輸出第一次發生在哪一站（亦即有發生的車站中編號最小的）就好。接著請依據該站發生之不合理現象的種類輸出一個字元，若是第一種不合理現象「下車人數大於車上人數」請輸出 N、第二種不合理現象「車上人數大於載客人數上限」請輸出 C、同時在這個車站出現兩種不合理現象則請輸出 B；若全程都沒有發生不合理現象，請直接輸出 0 。

在判定上車後人數是否超過 K 時，請一律以原問題的下車人數來檢查，即使在該站下車後的人數不合理（是負的），亦即如果下車紀錄已經出現問題（例如因爲下車人數超過車上實際人數導致的人數

[1] 課本是 Deitel and Deitel 著的 *C++ How to Program: Late Objects Version* 第七版。

爲負值），那麼在計算上車人數時，應該基於這個不合理的人數來進行。舉例來說，假設車上最多只能載 40 人（即 $K = 40$）。假如到某一站時，車上原有 20 人，但下車紀錄爲 39 人，導致車上實際人數爲 -19 人。若接著紀錄顯示有 50 人上車，儘管這似乎是一個明顯不合理的紀錄，但如果我們基於前面的 -19 人來計算，車上實際人數將是 31 人，這在 $K = 40$ 的限制下是合理的。

輸入輸出格式

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中會有三列，第一列裝著三個正整數，依序是 n、w、K，代表除了起點站的總站數、起點站上車人數，以及巴士載客限制；第二列存了 $n-1$ 個非負整數，依序是 x_1、x_2 直到 x_{n-1}，代表第一站的上車人數、第二站的上車人數，直到終點站前一站的上車人數；第三列存了 $n-1$ 個非負整數，依序是 y_1、y_2 直到 y_{n-1}，代表第一站的下車人數、第二站的下車人數，直到終點站前一站的下車人數。已知 $2 \le n \le 20$、$1 \le w \le 500$、$1 \le K \le 500$、$0 \le y_i \le 50$、$0 \le x_i \le 50$、$w \le K$。請依題目指示，找到第一次出現不合理現象是在哪一站，並輸出該車站編號以及不合理現象種類，兩個字之間以一個逗號隔開。若全程都沒有出現上述不合理現象，就輸出 0。

舉例來說，如果輸入是

```
5 20 40
6 9 3 2
8 5 7 1
```

則輸出應該是

```
0
```

如果輸入是

```
5 2 40
6 9 3 2
8 5 7 1
```

則輸出應該是

```
1,N
```

如果輸入是

```
5 20 40
6 9 3 50
8 5 7 1
```

則輸出應該是

```
4,C
```

你上傳的原始碼裡應該包含什麼

你的 .cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你可以使用任何方法。

評分原則

這一題的所有分數都根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。

第二題

（20 分）在本題中，你跟你的夥伴要一起用「函數」的做法重新改寫第一題（題幹敘述、輸入與輸出格式皆維持不變）。你的夥伴已經把 main function 寫好了，要求你按照他的設計寫一個函數；他會把讀入的資料存入對應的變數或陣列，接著把這些變數和陣列傳進你寫的函數，讓你做計算並且回傳是否有車站的紀錄出錯，有的話是哪個車站，以及是哪種錯誤。

具體來說，首先我們定義 `MAX_STATION_NUM` 這個 constant 來代表車站數的最大可能的值，接著定義這個函數的 prototype 爲

```
int checkCorrectness(int stationNum, int initPass, int maxPass, const int
    getInPass[], const int getOutPass[]);
```

其中 `stationNum` 代表除了起點站的總站數，`initPass` 是起點站上車人數，`maxPass` 是巴士人數上限，`getInPass` 是一個長度爲 `MAX_STATION_NUM` 但應該要裝著 `stationNum - 1` 個元素的一維陣列，裡面每個元素就是第一站、第二站直到倒數第二站的各站上車人數，`getOutPass` 的資料型態和 `getInPass` 相同，裡面每個元素就是第一站、第二站直到倒數第二站的各站下車人數。

若你仔細地思考，應該會發現這個函數可能會需要回傳兩個值，這在 C++ 用基本作法是做不到的。因此，你的夥伴設計了你的函數的回傳機制如下。首先，如果整份資料都沒有不合理的情況，就回傳 0；若有不合理的現象，假設第一次發現不合理現象的車站編號爲 i，若該站是發生「下車人數大於車上人數」則回傳 $n+i$（n 是起點站之外的車站個數）、第二種不合理現象「車上人數大於載客人數上限」請回傳 $2n+i$、同時在這個車站出現兩種不合理現象則請回傳 $3n+i$。由於 i 最大也只會是 $n-1$，這樣的機制可以正確地透過回傳一個整數來傳遞兩份資訊[2]。

他寫的 main function 如下（如附件 PD112-1_hw02_main02.cpp）：

```
#include <iostream>
using namespace std;
const int MAX_STATION_NUM = 20;

// 這是你需要完成的函數的 prototype
int checkCorrectness(int stationNum, int initPass, int maxPass, const int
    getInPass[], const int getOutPass[]);
```

[2]說不定你會覺得這個機制蠻不自然的。不過因爲你的夥伴已經把 main function 都寫好了，爲了不要破壞和氣，你決定還是按照他的規劃去寫。在第四題我們會有另一種設計。

```
int main(){
    int stationNum = 0, initPass = 0, maxPass = 0;
    int getInPass[MAX_STATION_NUM] = {0};
    int getOutPass[MAX_STATION_NUM] = {0};

    cin >> stationNum >> initPass >> maxPass;
    // 請注意在下面的兩個迴圈中，
    // 第一站的上下車人數放在陣列的第一個而非第零個元素。
    // 其他依此類推
    for(int i = 1; i < stationNum; i++){
        cin >> getInPass[i];
    }
    for(int i = 1; i < stationNum; i++){
        cin >> getOutPass[i];
    }

    int ret = checkCorrectness(stationNum, initPass, maxPass, getInPass,
       getOutPass);

    if(ret == 0){
        cout << ret;
    }
    else
    {
        if(ret - 3 * stationNum > 0){
            cout << ret - 3 * stationNum << ",B";
        }
        else if(ret - 2 * stationNum > 0){
            cout << ret - 2 * stationNum << ",C";
        }
        else{
            cout << ret - stationNum << ",N";
        }
    }

    return 0;
}

// 你上傳你寫的函數後，PDOGS 會把你上傳的程式碼貼在這裡，然後編譯
```

特別注意：在這題之中，助教已經在 PDOGS 上設定好上面的「你的夥伴」寫的程式了。你需要完成一個完整的 `checkCorrectness` 函數，自己測試的時候當然需要結合上面的 main function，但在

繳交到 PDOGS 時請只上傳這個 `checkCorrectness` 函數，PDOGS 會自動把你上傳的函數跟已經在 PDOGS 上的程式拼起來去編譯。換言之，在本題你被迫必須要實作本題指定的函數；如果你上傳了任何帶有你寫的 main function 的程式，你會無法得到分數的！

輸入輸出格式

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。輸入輸出格式與第一題都完全相同。

你上傳的原始碼裡應該包含什麼

你的 .cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你可以使用任何方法。

評分原則

這一題的所有分數都根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。

第三題

（50 分）承上題，「開心觀光巴士」在歷史上已經開出很多班了，我們不想要一班一班檢查，而是希望有個函數可以一次幫我們檢查許多班的紀錄。本題會一次給定 m 輛巴士的乘車紀錄，請逐一檢查每一輛巴士有沒有出現上一題的三種不合理狀況。每一輛巴士都有可能發生複數次不合理狀況，此時只要關注其第一次發生並判斷其狀況是三種的哪一種即可。最後，請輸出 m 輛巴士的第一次不合理共有幾次是類型一（下車後人數變成負的，但該站上車後人數合理）、類型二（上車後人數超過巴士容量，但該站下車人數合理），以及類型三（下車後人數變成負的，在同一站緊接著上車後人數超過巴士容量）。由於每輛巴士只會被歸類到最多一類不合理，最後輸出的三個數字加總一定不會超過 m。

輸入輸出格式

系統會提供一共 15 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中會有 $m+1$ 列，第一列裝著三個正整數，依序是 n、m、K，代表除了起點站的總站數、巴士數量，以及巴士載客限制；第二列到第 $m+1$ 列分別存著 $2n-1$ 個整數，其中第 i 列依序存放 w_i、$x_{i,1}$、$x_{i,2}$、...、$x_{i,n-1}$、$y_{i,1}$、$y_{i,2}$、...、$y_{i,n-1}$，依序代表第 i 台巴士的起點上車人數、每一站的上車人數以及每一站的下車人數。已知 $2 \le n \le 20$、$2 \le m \le 50$、$1 \le K \le 500$、$1 \le w_i \le 500$、$0 \le y_{ij} \le 50$、$0 \le x_{ij} \le 50$、$w_i \le K$。

請依題目指示寫一個完整的程式（不是只完成一個函數），讀入上述資訊後，輸出 m 輛巴士中有幾輛的第一次不合理被歸類在第一種、第二種和第三種，並將三個數字以逗點隔開。舉例來說，如果輸入是

```
5 3 40
20 6 9 3 2 8 5 7 1
2 6 9 3 2 8 5 7 1
```

```
20 6 9 3 50 8 5 7 1
```

則輸出應該是

```
1,1,0
```

如果輸入是

```
5 4 40
10 6 3 8 2 50 2 4 1
20 5 3 1 9 6 3 5 20
2 40 6 2 5 1 1 2 30
5 8 7 9 5 3 10 2 2
```

則輸出應該是

```
2,1,0
```

你上傳的原始碼裡應該包含什麼

你的 .cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你***不可以***使用上課沒有教過的方法：

- 確定可以使用的語法包含 `if-else`、`for`、`while`、陣列、函數、`<climits>` 裡面所有的東西、`<iomanip>` 裡面所有的東西、`<cmath>` 裡面的 `abs()` 和 `sqrt()`、`sizeof()`、`static_cast()`、constants 等。
- 確定不可以使用的語法包含 `printf`、`scanf`、`max`、`min`、`<cmath>` 裡面除了 `abs()` 和 `sqrt()` 以外的函數、動態配置記憶體等等。

請注意正面表列的固然是都確定可以用，但沒有被負面表列的不表示可以用喔！

此外，請幫你的程式寫適當的函數。你可以用第二題已經寫好的函數、你幫第四題寫的函數，或者你自己設計的函數，但總之你的程式應該要藉由使用函數來實現理想的程式結構。

評分原則

- 這一題的其中 30 分會根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。
- 這一題的其中 20 分會根據你所寫的程式的品質來給分。助教會打開你的程式碼並檢閱你的程式的運算邏輯、可讀性（包含排版、變數命名、註解等等）、可擴充性，以及是否使用了還沒教過的語法。請寫一個「好」的程式吧！

  除此之外，如果這一題你沒有自行定義函數，也會被扣分喔！

第四題

（20 分）本題承第二題，但是與第三題無關，而題目的要求和第二題一樣，只是你的夥伴修改了他的 main function，要求你按照他的設計更改你的函數[3]。他會把讀入的資料存入對應的變數或陣列，接著把這些變數和陣列傳進你寫的函數，讓你做計算並且回傳是否有車站的紀錄出錯，有的話是哪個車站，以及是哪種錯誤，但這次回傳兩個值的方式是透過一個裝回傳值的陣列。

具體來說，首先我們定義 `MAX_STATION_NUM` 這個 constant 來代表車站數的最大可能的值，接著定義這個函數的 prototype 爲

```
void checkCorrectness(int stationNum, int initPass, int maxPass, const int
    passenger[][MAX_STATION_NUM], int ret[] );
```

其中 `stationNum`、`initPass`、`maxPass` 的型態和意義同第二題、`passenger` 是一個大小爲 `2 * MAX_STATION_NUM` 但理論上應該裝著 `2 * (stationNum - 1)` 個變數的二維陣列，裡面每個元素就是從起點站、第一站、第二站，直到第 $n-1$ 站（倒數第二站），每一站上車與下車的人數（請參考下方 main function 瞭解具體格式）。最後 `ret` 是一個長度爲 2 的一維陣列，在傳入時裡面存什麼都無所謂，但函數執行完畢時裡面需要存著函數想要回傳的兩個值（因應 C++ 的函數只能有一個回傳值的設計，這是一種回傳多個值的方法），其中 `ret[0]` 存 0 或車站編號，如果函數參數代表的班次沒有不合理之處就存 0，反之就存第一次發生不合理現象的車站編號；若沒有不合理之處則 `ret[1]` 存什麼都無所謂，反之則存 1、2、3 分別代表第一種（太多人下車）、第二種（太多人上車）或第三種（前兩者同時發生）錯誤。

助教們會提供好 main function（如附件 PD112-1_hw02_main04.cpp），程式碼如下：

```
#include <iostream>
using namespace std;
const int MAX_STATION_NUM = 20;

// 這是你需要完成的函數的 prototype
void checkCorrectness(int stationNum, int initPass, int maxPass, const int
    passenger[][MAX_STATION_NUM], int ret[]);

int main(){
    int stationNum = 0, initPass = 0, maxPass = 0;
    int passenger[2][MAX_STATION_NUM] = {0};
    int ret[2] = {0};
    cin >> stationNum >> initPass >> maxPass;
    // 請注意在下面的兩個迴圈中，
    // 第一站的上下車人數放在陣列的第一個而非第零個元素。
    // 其他依此類推
    for(int i = 1; i < stationNum; i++){
        cin >> passenger[0][i];
```

[3]實務上這是最不應該發生的事，負責寫 main function（整體程式架構）的人和負責寫函數（完成特定任務）的人應該先討論、約定好函數的規格（輸入格式、該做的運算，以及輸出格式），然後除非必要否則不要變動，以便任何一方修改自己的程式碼時，另一方不需要跟著修改，這樣才有發揮模組化的效果。當然理想歸理想，如果是兩個都沒什麼經驗的程式設計師一起合作，那開發到後期去修改前期設計好的函數規格也是常有的事，只能靠經驗和學習慢慢累積、成長了！

```
    }
    for(int i = 1; i < stationNum; i++){
        cin >> passenger[1][i];
    }

    checkCorrectness(stationNum, initPass, maxPass, passenger, ret);

    if(ret[0] == 0){
        cout << ret[0];
    }
    else{
        if(ret[1] == 1){
            cout << ret[0] << ",N";
        }
        else if(ret[1] == 2){
            cout << ret[0] << ",C";
        }
        else if(ret[1] == 3){
            cout << ret[0] << ",B";
        }
    }

    return 0;
}

// 你上傳你寫的函數後，PDOGS 會把你上傳的程式碼貼在這裡，然後編譯
```

請注意 `int passenger[][MAX_STATION_NUM]` 前面有加 `const` 修飾詞，這是爲了防止傳入函數的參數值（argument）被函數內的任何操作更改到，但 `int ret[]` 卻沒有，因爲這個陣列在函數內就是要被寫入值的。若你對這一切感到有點陌生，建議回頭複習課程影片。

特別注意：與第二題相同，在這題之中，助教已經在 PDOGS 上設定好上面的「你的夥伴」寫的程式了。你需要完成一個完整的 `checkCorrectness` 函數，自己測試的時候當然需要結合上面的 main function，但在繳交到 PDOGS 時請只上傳這個 `checkCorrectness` 函數，PDOGS 會自動把你上傳的函數跟已經在 PDOGS 上的程式拼起來去編譯。換言之，在本題你被迫必須要實作本題指定的函數；如果你上傳了任何帶有你寫的 main function 的程式，你會無法得到分數的！

輸入輸出格式

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。輸入輸出格式與第二題都完全相同。

你上傳的原始碼裡應該包含什麼

你的 .cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你可以使用任何方法。

評分原則

這一題的所有分數都根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。