程式設計（112-1）

作業九

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、三題上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳），並且到 NTU COOL 以線上輸入的方式輸入第四題的答案。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。這份作業的程式碼和書面報告的截止時間都是 ***11 月 21 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顏子鈞。

在你開始前，請閱讀課本的第 8 和 18 章（分別關於 File I/O 以及 C++ string）[1]。

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）接續作業八的第一題，以下將再從頭敘述一次該題的相關題目要求，跟作業八一樣，有一些細節與之前的不盡相同，建議大家在此詳細閱讀一次題目。

有家公司請你寫程式維護他們家關於產品的資訊，並且提供某些搜尋、彙總功能，任務如下。該公司有 n 種產品，每種產品都有其產品名稱（不重複且只包含英數和空白字元的字串）、單位零售價、單位人力成本（與作業第一題不同）和截至去年底爲止的總銷售量資訊。我們稱呼產品 i 的產品名稱爲 x_i、單位零售價爲 p_i、單位人力成本爲 c_i^L、截至去年底爲止的總銷售量爲 q_i。爲了簡單起見，我們假設這些產品的成本和價格都從未改變過。

每一項產品由數種零件組成，每種零件都會有其零件名稱（不重複且只包含英數和空白字元的字串），以及單位原料成本。我們稱呼零件 j 的零件名稱爲 z_j、單位原料成本爲 c_j^M，而產品 i 使用到的零件被放在 M_i 這個多重集合（multiset）裡[2]。一樣爲了簡單起見，我們假設這些零件的成本都從未改變過。

有了以上零件的資訊，產品 i 的單位生產總成本（簡稱單位總成本）即爲

$$c_i = c_i^L + \sum_{j \in M_i} c_j^M \text{。}$$

舉例來說，該公司可能有 $n = 4$ 種產品，其基本資訊如表 1 所示，同時可能有 $m = 5$ 種零件，其基本資訊如表 2 所示，最後，4 種產品與 5 種零件之間的關係如表 3 所示，記錄每一種產品分別由哪些零件構成。請注意組成關係表內沒有排序，且一個產品可能用到一個零件複數次（像是 jPhone 41 Pro 產品與零件 Lens 分別在關係 7 與 13 都有出現，這代表著一支 jPhone 41 Pro 是由兩個 Lens 零件組成的）。給定上述資訊，表 4 中即爲各產品的單位毛利（單位零售價減單位總成本），以及歷史總營收（單位零售價乘以歷史總銷售量）、歷史總利潤（單位毛利乘以歷史總銷售量）。

該公司會對你的程式發動查詢，在作業八的第一題中，我們請你回傳根據某指標由大到小排序後前 k 名之產品的有史以來的利潤總和。可能用來排序的指標包含各產品的單位零售價、單位總成本、單位

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

[2] 所謂多重集合，就是一個元素可以出現複數次的集合。詳情可以參閱 `https://en.wikipedia.org/wiki/Multiset`。

產品編號	產品名稱	單位零售價	單位人力成本	截至去年底的總銷售量
1	jPhone 41	200	100	1000
2	jPhone 41 Pro	250	90	800
3	WaterPods	40	5	1200
4	WaterPods Max	50	15	900

表 1: 範例產品資訊

零件編號	零件名稱	單位原料成本
1	Screen	5
2	Speaker	10
3	Battery	5
4	Motherboard	10
5	Lens	10

表 2: 範例零件資訊

毛利（單位零售價減單位總成本）、歷史總銷售量、歷史總營收（單位零售價乘以歷史總銷售量），以及歷史總利潤（單位毛利乘以歷史總銷售量）共六種。排序時如果遇到平手，一律依 `<cstring>` 函式庫裡面的 `strcmp` 對產品名稱的排序排序（字典順序較前的排名在前）。舉例來說，如果排序指標是總銷售量且 $k = 3$，則前三名分別是「WaterPods」、「jPhone 41」和「WaterPods Max」，歷史利潤總和是 $24000 + 80000 + 18000 = 122000$。如果排序指標是總營收且 $k = 1$，則第一名是「jPhone 41」，歷史利潤總和是 80000；請注意「jPhone 41」和「jPhone 41 Pro」的總營收相同，則因爲 `strcmp` 把「jPhone 41」排在「jPhone 41 Pro」前面，所以「jPhone 41」是第一名。

在本題中，讓我們依據助教們在作業八提供的參考解答，來完成以下三項進階任務：

- 實作 `Product` 的 copy constructor。
- 對 `Product` 多載「`<`」這個運算子，讓兩個產品之間可以直接透過此運算子，對於歷史總利潤（單位毛利乘以歷史總銷售量）進行比較。換言之，如果之前有個叫 `isInFrontOf` 的函數，那就是把它改一改。
- 對 `Product` 多載「`[ ]`」這個運算子，讓一個產品物件 `p` 可以透過 `p[i]` 來取得索引值爲 i 的零件的指標。如果不存在索引值爲 i 的零件（例如當 $i < 0$ 時），請直接回傳一個名稱爲空字串、原料成本爲零的 `Item` 物件指標。

我們會在 main function 中使用一些方式，來測試以上任務是否被完整地實作。根據作業八提供的參考解答（該 CPP 檔在 NTU COOL 上可以下載）的架構，程式碼大概會長得像下面這樣子，

```
class Item {
  // ...
};

class Product {
```

關係編號	產品名稱	零件名稱
1	jPhone 41	Screen
2	jPhone 41	Battery
3	jPhone 41 Pro	Screen
4	jPhone 41 Pro	Speaker
5	jPhone 41 Pro	Battery
6	jPhone 41 Pro	Motherboard
7	jPhone 41 Pro	Lens
8	jPhone 41	Motherboard
9	WaterPods	Screen
10	WaterPods	Speaker
11	WaterPods Max	Screen
12	WaterPods Max	Motherboard
13	jPhone 41 Pro	Lens

表 3: 範例產品與零件關係資訊

產品編號	產品名稱	單位毛利	總營收	總利潤
1	jPhone 41	80	200000	80000
2	jPhone 41 Pro	110	200000	88000
3	WaterPods	20	48000	24000
4	WaterPods Max	20	45000	18000

表 4: 範例歷史總資訊

```
private:
    // ...
public:
    Product(char* name, int price, int laborCost,
            int salesQty, int itemCnt);
    ~Product();
    // Implement the following
    Product(const Product& other);
    bool operator<(const Product& other) const;
    Item* operator[](int index) const;
    // ...
};
```

此外，在本題中我們會以如下的 main function 來測試任務的完成度：

```
int main() {
    // see the attached file for details here
```

```
    // testing the copy constructor
    Product newProduct(*products[1]);
    newProduct.addItem(items[1]);
    cout << products[1]->getName() << "\n";
    cout << products[1]->calculateProfit() << "\n";
    cout << newProduct.getName() << "\n";
    cout << newProduct.calculateProfit() << "\n";

    // testing the "<" operator overloading
    if(*products[0] < *products[1]) {
        cout << products[0]->getName() << " is less profitable than "
             << products[1]->getName() << "\n";
    }
    else {
        cout << products[0]->getName() << " is no less profitable than "
             << products[1]->getName() << "\n";
    }

    // testing the "[]" operator overloading
    Item* secondItemOfFirstProduct = (*products[0])[1];
    cout << secondItemOfFirstProduct->getName() << "\n";
    cout << secondItemOfFirstProduct->getMaterialCost() << "\n";

    // see the attached file for details here
}
```

在這題之中，助教已經在 PDOGS 上設定好絕大部分程式碼了。在繳交到 PDOGS 時請只上傳你對 copy constructor 和兩個運算子的實作，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。在每個檔案中會有 $n+m+u+2$ 行：

- 第一行包含三個整數，依序爲 n、m、u，依序代表產品數量、零件數量，以及產品零件關係數量；
- 第二行起的 n 行中，第 $i+1$ 行包含產品 i 的產品名稱 x_i（只含有英數和空白字元的字串，且頭尾一定不會是空白）、單位零售價 p_i（整數）、單位人力成本 c_i^L（整數）、截至去年底的總銷售量 q_i（整數），以及組成該產品的零件數量（若某產品要用到 1 個零件 A 和 2 個零件 B，則其零件數量爲 3 而非 2）；每一行的 x_i 和 p_i 之間被一個逗點隔開，剩下的相鄰整數之間被一個空白字元隔開。

- 第 $n+2$ 行起的 m 行中，第 $n+1+j$ 行包含零件 j 的零件名稱 z_j（只含有英數和空白字元的字串，且頭尾一定不會是空白）、單位原料成本 c_j^M（整數）；每一行的 z_j 和 c_j^M 之間被一個逗點隔開。
- 第 $n+m+2$ 行起的 u 行中，第 $n+m+1+k$ 行包含關係 k 的產品名稱 x_k、零件名稱 z_k；兩兩之間被一個逗點隔開。
- 最後一行包含兩個整數 y 和 k，中間以一個空白字元隔開；當 y 的值是 1、2、3、4、5、6 時，排序條件分別是單位零售價、單位成本、單位毛利、總銷售量、總營收、總利潤。

已知 $1 \le k \le n \le 100$、$1 \le m \le 1000$、$n \le u \le 10000$、$y \in \{1, 2, ..., 6\}$、x_i 的長度介於 1 和 100 之間（包含 1 和 100）、z_j 的長度介於 1 和 100 之間（包含 1 和 100）、$1 \le |M_i| \le 2m$（$|M_i|$ 是產品 i 使用的零件數，這裡是表示一個產品用到的零件數最多是 $2m$ 個）、$0 \le c_j^M \le 50$、$1 \le \sum_{j \in M_i} c_j^M + c_i^L \le p_i \le 1000$、$0 \le q_i \le 10000$、任兩個 x_i 或 z_j 均不完全相同。

讀入這些資訊後，程式應該執行前述 main function 並且輸出執行結果。舉例來說，如果輸入是

```
4 5 13
jPhone 41,200 100 1000 3
jPhone 41 Pro,250 90 800 6
WaterPods,40 5 1200 2
WaterPods Max,50 15 900 2
Screen,5
Speaker,10
Great Battery,5
Motherboard,10
Lens,10
jPhone 41,Screen
jPhone 41,Great Battery
jPhone 41 Pro,Screen
jPhone 41 Pro,Speaker
jPhone 41 Pro,Great Battery
jPhone 41 Pro,Motherboard
jPhone 41 Pro,Lens
jPhone 41,Motherboard
WaterPods,Screen
WaterPods,Speaker
WaterPods Max,Screen
WaterPods Max,Motherboard
jPhone 41 Pro,Lens
4 3
```

根據以上的寫好程式碼爲例，輸出應該是

```
122000
jPhone 41
```

```
80000
jPhone 41
70000
WaterPods is less profitable than jPhone 41
Speaker
10
```

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只包含指定的函數 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）本題我們要來執行貨幣字串的處理任務，請實作一個類別叫做 `CurrencyString`，裡面會存著一個整數，代表貨幣的金額，此外，我們希望可以實作一個函數 `getFormattedStr`，專門針對不同的表示方式來回傳貨幣格式的字串。

首先，我們希望在所有數額前面都加上一個金錢符號「`$`」。接著，這些帶有 `$` 的貨幣格式字串還有兩種表達方式。第一種是「是否使用逗號作爲千分位符號」，也就是當一個數字超過三位數時，每三個數字應該要以一個分隔符號間隔，而分隔符號可能是逗號「`,`」，也可能是小數點「`.`」[3]。第二種是「是否標示負號」，當一個數額爲負數時，我們可以選擇在錢字號的前面加上負號「`-`」，也可以選擇不加負號，改用半形括號「`()`」將錢字號和數額包起來表示負數，而非負數則完全不受影響。我們希望可以藉由實作這個函數，將 `CurrencyString` 的數額，以規定的表示方式，輸出對應的字串。

舉例來說，如果我們有一個正數 582930 以及一個負數 -7859306：

- 若選擇「使用逗號作爲分隔符號，且要標示負號」，則輸出 $\$582,930$ 以及 $-\$7,859,306$。
- 若選擇「使用逗號作爲分隔符號，且不要標示負號」，則輸出 $\$582,930$ 以及 $(\$7,859,306)$。
- 若選擇「不使用逗號作爲分隔符號，且要標示負號」，則輸出 $\$582.930$ 以及 $-\$7.859.306$。
- 若選擇「不使用逗號作爲分隔符號，且不要標示負號」，則輸出 $\$582.930$ 以及 $(\$7.859.306)$。

根據以上敘述，你的程式碼應該會長得像下面這個樣子：

[3]世界上有些國家是用一個英文句點當小數點、用一個逗點當千分位符號，例如臺灣，但也有很多國家是剛好相反的，例如德國和許多歐洲國家。

```
#include<iostream>
using namespace std;

class CurrencyString
{
private:
    int num;
public:
    CurrencyString() : num(0) {}
    CurrencyString(int n) : num(n) {}
    string getFormattedStr(bool commaSep, bool minusSign);
};

string CurrencyString::getFormattedStr(bool commaSep, bool minusSign)
{
   // implement this function
}
```

請根據題目輸入，輸出指定格式的貨幣字串。

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。在每個檔案中會有一行，第一行含有三個整數，依序是 n、c 以及 m，依序代表貨幣金額、是否以逗點當千分位符號，以及是否以負號表示負數。當 $c = 1$ 代表以逗號當千分位符號，當 $c = 0$ 時則以句號當千分位符號。當 $m = 1$，在數額爲負時要在錢字號之前加上負號，當 $m = 0$ 時則在數額爲負時在前後加上半形括號。已知 $-1,000,000,000 \leq n \leq 1,000,000,000$、$c \in \{0,1\}$、$m \in \{0,1\}$。

讀入這些資訊後，程式應該執行前述 main function 並且輸出執行結果。舉例來說，如果輸入是

```
-2568 1 0
```

則輸出應該是

```
($2,568)
```

如果輸入是

```
582930 0 1
```

則輸出應該是

```
$582.930
```

如果輸入是

```
-7859306 0 1
```

則輸出應該是

```
-$7.859.306
```

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）本題要實作的是文章的情緒分析。題目會給定一篇文章，以及一份正向情緒字典。請寫一支程式，計算出該篇文章中，總共有多少個正向情緒詞彙。

一篇指定的文章會長得像下面這樣（請見 PD112-1_hw9-3_sample_article.txt）

> In a bright, cozy village, there lived a cheerful bunny named Toby. Every morning, Toby hopped out of bed with a smile as wide as the river. With a heart full of joy and a skip in his step, he greeted his friends with a sunny “Good morning!” They would often spend the day exploring the lush meadows, playing in the soft grass, and sharing delightful stories. As the sun set, painting the sky in shades of pink and orange, Toby would whisper, “What a wonderful day!” and dream of tomorrow’s adventures.

一個正向情緒詞彙的字典會長得像下面這樣，其中第一行的數字表示該字典中有那麼多個正向情緒詞彙（請見 PD112-1_hw9-3_sample_dictionary.txt）

```
30
Joyful
Cheerful
Bright
Cozy
Friendly
Smile
Heartwarming
Skip
Greet
```

```
Sunny
Delightful
Explore
Lush
Playful
Sharing
Soft
Stories
Sunset
Painting
Pink
Orange
Whisper
Wonderful
Dream
Adventure
Peaceful
Charming
Blissful
Thriving
Serene
```

根據指定的文章與字典，我們要在文章中搜索出字典裡有的詞彙，並累計總出現次數。搜索的規則是該詞彙必須要「完整且獨立」的出現在文章中才算出現，而獨立與否取決於在文章中該詞彙前後是否有英文字元，如果有則不獨立，反之則獨立。舉例來說，如果我們要搜索「happy」這個詞彙，若文章內容是「I am unhappy.」或「I am happyhappy」，則視爲「happy」這個詞彙沒有出現；若文章內容是「Am I happy?」，則忽略標點符號的「?」而將「happy」視爲出現過一次；若文章內容是「Am I happ?y」，則視爲「happy」這個詞彙沒有出現（因爲出現過的是「happ」和「y」）；若文章內容是「This is John's pen.」，則視爲「John」這個詞彙有出現而「Johns」沒出現。請執行 case-insensitive search，意即同一個字母的大小寫，一律視爲相同，例如在「Am I HApPY?」中「happy」被視爲出現過一次。

最後提醒大家，文章不一定總是符合英文寫作規範，例如也有可能出現「I is a student.」、「I ama student.」或「I am a student,not a teacher.」等等文法、拼字、標點符號運用、空格運用的問題，所以在處理文章時，請不要讓你的程式需要文章「正確」才能運作。

請依上述題目指示，計算指定文章的正向情緒詞彙總出現次數。

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。在每個檔案中會有兩行，第一行爲文章的檔案路徑；第二行爲正向情緒字典的檔案路徑。文章內容只有一行，只包含大小寫英文字母、空白字元、逗點、句點、單引號、雙引號、驚嘆號、問號、分號和冒號。總共包含 x 個字元，且 $1 \leq x \leq 10000$。字典內容的第一行是一個整數 n，代表字典的詞彙數量，且 $1 \leq n \leq 100$。後面 n 行中各含有一個詞彙，且每個詞彙都只包含大小寫英文字母，不包含空白字元、標點符號、換行符號等任何

其他符號。所有詞彙在 case-insensitive search 下是獨特的，不存在重複的詞彙。也就是，字典裡不會同時存在「Apple」和「apple」兩個詞彙。

讀入文章與字典的檔案路徑字串後，請把這兩個字串不做任何修改、直接傳進你用來讀取檔案的 constructor 或函數中，就可以讀取已經被放在 PDOGS 上的檔案。接著請讀取文章和字典內容，並輸出正向情緒詞彙在指定的文章中出現的總次數。舉例來說，如果輸入是

```
/assisting_data/PD112-1_hw9-3_sample_article.txt
/assisting_data/PD112-1_hw9-3_sample_dictionary.txt
```

且這兩個檔案的內容如附件（請假設這兩個檔案在 PDOGS 上放置的路徑是在 assisting_data 底下），則輸出應該是

```
17
```

舉例來說，如果輸入是

```
/assisting_data/PD112-1_hw9-3_sample_article.txt
/assisting_data/PD112-1_hw9-3_sample_dictionary2.txt
```

且字典檔的內容是（請見 PD112-1_hw9-3_sample_dictionary2.txt）

```
1
Joyful
```

則輸出應該是

```
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 C++ 字串、自訂標頭檔、`<fstream>` 裡面的所有功能，以及在之前的作業中正面表列過的語法。
- 確定不可以使用的語法包含 `printf`、`scanf`、繼承、多型等等。

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！

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結構、運算邏輯、可讀性（包含排版、變數命名、註解等等）、可擴充性、模組化程度，以及是否使用了還沒教過的語法。請寫一個「好」的程式吧！

第四題

特別說明：

- 本題爲作文題而非程式題，請不要上傳東西到 PDOGS。由於班上同學很多，但助教人力有限，所以原則上我們會從所有繳交中隨機批改約 50%。沒被抽到的同學在本題就會得到滿分，但不會得到直接的助教回饋；有被抽到的同學們可能會被扣分，但相對應地也會得到助教的回饋。
- 本題請大家在 NTU COOL 上直接輸入答案，而非繳交 PDF 檔。由於直接輸入的是純文字，大家可以不用費心排版、調整格式等等。但如果你的敘述會有數個段落，請用空行區隔段落，而各段開頭不用（不要）手動縮排。

（20 分）在第二題中，我們把 `commaSep` 和 `minusSign` 設成 instance function 的參數。你的朋友小明說，應該要把它們設成 `CurrencyString` 這個類別的 static variable 才對。請問你認同小明的說法嗎？爲什麼？寫成 static variable 有什麼好處，維持是 instance function 的參數又有什麼好處？最後，不論你認同與否，都不用重寫一次程式，只要說明優點缺點和你是否認同即可。