

程式設計 (106-1)

作業十一

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/judge/>) 為第一題上傳一個 PDF 檔、為第二題做同儕互評，再為第三、四題各上傳一份 C++ 原始碼 (以複製貼上原始碼的方式上傳)。第四題是 bonus 加分題。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。請以英文或中文作答。

這份作業的截止時間是 **2017 年 12 月 12 日凌晨一點**。在你開始前，請閱讀課本的第 22.1-22.6 節¹。為這份作業設計測試資料並且提供解答的助教是楊佩蓉。

第一題

(40 分) 請考慮上課教的 `struct Point`：

- (a) (10 分) 我們想要把兩個二維平面上的點比出個大小，而比法是看誰的 x 值比較小，如果平手則看 y 的值誰比較小。舉例來說， $(1, 2)$ 比 $(2, -10000)$ 小，而 $(2, -10000)$ 又比 $(2, -9999)$ 小。

請寫一個成員函數

```
bool Point::isSmallerThan(Point p);
```

讓一個 `Point` 去呼叫後，可以跟傳進函數的 `Point p` 比較，如果呼叫函數的 `Point` 比 `p` 小則回傳 `true`，一樣大或比較大則回傳 `false`。

- (b) (10 分) 承上題，請寫一個 global 函數

```
int compare(Point p1, Point p2);
```

如果 `p1` 比 `p2` 小則回傳 `-1`、一樣大則回傳 `0`，比較大則回傳 `1`。

- (c) (20 分) 承上題，請寫一個 `struct Triangle` 裡面含有三個 `Point` 當成員變數，分別是三角形的三個端點。為 `Triangle` 寫一個成員函數

```
void Triangle::addPoint(Point p);
```

當一個 `Triangle` 呼叫此函數並且得到一個 `Point p` 時，如果 `p` 比任意一個原本的三角形端點「小」，就用 `p` 取代原本的三角形端點中最「大」的那個，來成為這個三角形的新頂點。若你要用前兩題寫過的程式，用哪一個比較合適？為什麼？

第二題

(0 分) 請在 PDOGS 上批改你被隨機分配到的作業十第三題的程式碼，根據它在正確性以外的部份給它 1 至 5 分的評分，並且說明你給分的依據。建議在評分時參考以下六個面向。在前五個面向上，一個

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

面向上做得好就得一分，還不錯則半分，不好則零分；在第六個面向上則在有必要時扣分。六個面向的分數合計後無條件進入即為你最後給的總分。

- 可讀性：變數與函數名稱是否具有合適的資訊量？程式碼排版是否良好且具有前後一致性？是否有合適的註解？關於註解，當然不需要每一行都有註解，但若你發現在某一大段落裡都沒有註解，或某個你感覺很不易看懂的部份沒有註解，你可以指出來；不要直接說「註解太少」但沒有說是哪邊缺乏註解。
- 模組化程度：是否有宣告合適的函數、`structure` 或 `class`²？是否有避免將非常類似的程式片段寫複數次而非寫成函數？是否有避免一個函數做非常多事情？函數間是否有合適的 `decoupling`？直接閱讀 `main function` 是否能很快地理解程式在大方向上的運算邏輯？
- 效率：程式運算是否有合理的運算效率？當然我們不要求每個同學都寫出超級有效率的精妙演算法，但至少一個程式不應該進行過多不必要的運算，也不應該耗用過多不必要的記憶體空間。如果你看不出這個程式的效率有明顯的問題，我們建議你直接給一分。
- 擴充性：當要解的問題變得更複雜的時候，我們能不能簡單地修改這個程式以解決新的問題，而不是寧可砍掉重練？這個議題當然也很主觀，所以如果你不能明確地指出在怎樣的新問題上，這個程式會有擴充性問題，我們建議你直接給一分；如果你不能指出很嚴重的問題，我們建議你至少給半分。但對批改者來說，這個關於擴充性的思考其實是很好的訓練。試試看吧！
- 其他：如果有任何其他令你想扣分的理由，請明確地寫出來並且在這個面向上扣分；沒有的話就給一分。
- 題目規範：你應該檢查那份程式碼有沒有違反題目的規範，如果有（例如題目說不可以用上課沒教過的東西，但他用了，或者題目說一定要用指標和動態記憶體配置，但他沒用），就扣他三分。當然，請明確地指出他哪邊違反了題目的規範。

本題其中 10 分取決於檢視你的程式碼的同學給你的分數總和（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性（原則上除非被申訴，且助教檢視後發現你確實評得很不公允，否則只要有評就會得到 10 分）。

第三題

(40 分) 有 n 個工作要分給 m 個人。工作 j 有其完成所需時間 c_j 小時和完成後可得之報酬 b_j ，而每個人有有限的時間 K 小時。一份工作不能切給兩個或更多人做，只能完整地由一個人做完。一個人被分到的工作所需的總時間，不能超過 K_i 。每個人都想得到盡量多的報酬，但我們分配工作時想要最大化公平性，亦即讓每個人都領得差不多。我們設定的目標是最大化賺得最少的人所賺的總金額。

以表 1 為例，我們有 $n = 10$ 個工作，各有其所需時間和完成報酬。假設我們有 $m = 3$ 個人，各有 $K = 15$ 小時可以工作。如果我們分配工作 1 到 3 給員工 1、工作 4 到 7 給員工 2、工作 8 和 9 給員工 3，那麼三個人都會恰好工作 14 小時，並且分別賺到 23、20、6 元。則賺最少的人是員工 3，他只賺了 6 元。這個分法顯然是不太好。如果工作 3 和工作 8 交換，則員工 1 會僅工作 13 小時並賺得 20 元，而員工 3 會工作恰好 15 小時並且賺得 9 元。我們說第二個分法比較好，因為此時賺最少的人賺了 9 元，而前一個分法中賺最少的人只能賺 6 元。

²我們還沒教 `class`，所以本次批改請看函數和 `structure` 就好。

j	1	2	3	4	5	6	7	8	9	10
c_j	4	3	7	2	5	4	3	6	8	11
b_j	9	8	6	6	5	5	4	3	3	10

表 1: 公平分配的例子

在本題中，請實做下述演算法。首先，我們將工作依照 b_j 由大到小排序。如果有數個工作的 b_j 相等，那他們之間就用 c_j 由小到大排序。再平手就用編號由小到大排序。根據這個排序規則，排序後的結果如表 2 所示。

j	10	1	2	4	3	6	5	7	8	9
c_j	11	4	3	2	7	4	5	3	6	8
b_j	10	9	8	6	6	5	5	4	3	3

表 2: 公平分配的例子：排序後

接著我們將執行數輪的分配，每一輪我們將分配一個工作給某人。我們將從排序後的清單每次拿出一個工作來試圖分配給目前累積總報酬最低的人。如果平手，我們從他們之中選目前累積總工時最高的，再平手就選編號小的。當我們決定這個工作應該分給誰之後，我們得要確認這個人確實還有夠多時間來完成這個工作，如果有我們就把這個工作分給他，不然我們就分給第二順位的，再不然就第三順位的，依此類推。如果沒有任何一個人有足夠時間做這份工作，我們就放棄安排這個工作，繼續考慮下一份工作。演算法會一直持續直到把整個清單上的工作都考慮過一遍為止。

我們繼續用上面的例子來說明這個演算法。由於我們有 $n = 10$ 個工作，因此演算法將進行 10 輪。令 $y_k \in \{1, \dots, n\}$ 為第 k 輪分配之工作， $x_k \in \{0, 1, \dots, m\}$ 為第 k 輪分配之工作被分配的對象，其中 $x_k = 0$ 表示沒有分配給任何人，則最終的分配結果是 $y = (10, 1, 2, 4, 3, 6, 5, 7, 8, 9)$ 和 $x = (1, 2, 3, 3, 2, 1, 3, 2, 0, 0)$ ，表示第一輪分配工作 10 給員工 1、第二輪分配工作 1 給員工 2，依此類推。表 3 和表 4 列出每輪結束後每個人的累積工時和累積報酬。最終賺最少的人賺了 15 元，比剛剛亂分配的好多了。

輪	1	2	3	4	5	6	7	8	9	10
員工 1	11	11	11	11	11	15	15	15	15	15
員工 2	0	4	4	4	11	11	11	14	14	14
員工 3	0	0	3	5	5	5	10	10	10	10

表 3: 公平分配的例子：每輪結束後的累積工時

在本題中，你將被給定上述資訊，請根據給定的演算法分配工作，決定每個人的工作，並且計算出賺最少的人賺了多少錢。

輸入輸出格式

系統會提供一共 20 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中會有若干行，第一行包含三個整數 n 、 m 和 K ，第二行包含 n 個整數 c_1 、 c_2 直到 c_n ，第三行包含 n 個整數 b_1 、 b_2 直到

輪	1	2	3	4	5	6	7	8	9	10
員工 1	10	10	10	10	10	15	15	15	15	15
員工 2	0	9	9	9	15	15	15	19	19	19
員工 3	0	0	8	14	14	14	19	19	19	19

表 4: 公平分配的例子：每輪結束後的累積報酬

b_n 。同一行的任意兩個值之間被一個空白隔開。已知 $1 \leq n \leq 1000$ 、 $1 \leq m \leq 50$ 、 $1 \leq K \leq 100000$ 、 $1 \leq c_i \leq 100$ 、 $0 \leq b_i \leq 100$ 。

讀入資料後，請根據給定的演算法，印出三行資訊，其中第一行印出 y_1 、 y_2 直到 y_n 、第二行印出 x_1 、 x_2 直到 x_n 、第三行賺最少的人共賺多少。任兩個數字之間用一個空白鍵隔開。舉例來說，如果輸入是

```
10 3 15
4 3 7 2 5 4 3 6 8 11
9 8 6 6 5 5 4 3 3 10
```

則輸出應該是

```
10 1 2 4 3 6 5 7 8 9
1 2 3 3 2 1 3 2 0 0
15
```

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，**而且**你**不可以**使用 *structure*。

評分原則

這一題的全部 40 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。

第四題 (bonus)

(40 分) 承上題，現在請重寫一次第三題，但是**一定要用** *structure*。你的程式裡面至少要有一個 *structure* 是關於工作的，像是

```
struct Job
{
    int id;
    int workload;
    int benefit;
```

```
// some member functions
};
```

你應該會想要寫一些 member function。舉例來說，我們有個任務是幫這些工作排序，而排序時三個屬性都會用到。你或許會想要寫一個 member function 像是

```
bool Job::isBefore(Job j);
```

當呼叫函數的工作應該被排在傳入的工作 j 之前時回傳 `true`，反之則回傳 `false`。你可能還想要更多 structure，像是人、工作清單等等；你可能也會想要更多 member function。這些就留給你去設計了！

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。資料的輸入輸出格式都和第三題一模一樣。

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，但你**一定要**用 structure。

評分原則

- 這一題的其中 20 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 分會在作業十二中被評定。屆時我們會讓同學們互相檢視彼此的本題程式碼，並且就可讀性、易維護性、模組化程度、排版等面向寫評語和給評分（當然一切都是匿名的）。該任務在本題中會佔 20 分，其中 10 分取決於檢視你的程式碼的同學給你的分數（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性。若你在本次作業中完全沒有寫這一題，那屆時自然沒有人能檢視你的程式碼，你也就得要損失這 10 分了。