【滿分最高 125 分】【一、三、七，三題最多計 50 分】【四、五、六，三題最多計 50 分】

(隨身碟、手機、書包放教室前後，編譯成功可執行，並瞭解自己程式意義，助教詢問程式意義，依據題目定義與通過測資類別給分，每 1 類測試資料(明/暗測)都通過，得該題分數。每完成一題，務必舉手評分一次，程式不得使用全域變數

一、LinkList 建構唯一二元樹(26%)(4%, 4%, 4%, 4%, 5%, 5%)

給定前序或後序以及中序，建構唯一的二元樹(非二元搜尋樹)，節點資料為大寫英文字母。輸出剩下的尋訪資料（舉例來說，如果輸入二元樹的前序和中序，則輸出的內容應該是二元樹的後序）。
前序代號：P
中序代號：I
後序代號：O

本題須使用以下 struct 與 Link List 實作才計分。

```
typedef struct node_s {
char data;
struct node_s *right, *left;
} tree_t;
typedef tree_t *btree;
```

輸入說明	輸出說明
Line 1:輸入二元樹節點數量 n, 1<=n<=20 Line 2:輸入前序、中序或後序代號。 Line 3:輸入上筆的尋訪資料。 Line 4:輸入前序、中序或後序代號。 Line 5:輸入上筆的尋訪資料	1. 輸出剩下的尋訪資料。

Sample Input 1: 全滿二元樹找後序	Sample Output:
15 P ACEIJFKLDGMNHOP I IEJCKFLAMGNDOHP	IJEKLFCMNGOPHDA
Sample Input 2: 全滿二元樹找前序	Sample Output:
15 I TEPCHFIAMLGDOKF O TPEHIFCMGLOFKDA	ACETPFHIDLMGKOF
Sample Input 3: 全滿二元樹缺右葉節點找後序	Sample Output:
14 P KDFGSHTAEIBCLY	GSFTAHDBCIYLEK
I GFSDTHAKBICELY	
Sample Input 4: 全滿二元樹缺右葉節點找前序	Sample Output:
14 I KBTICMGPAOEFDY O KBIMCTPOAFYDEG	GTBKCIMEAPODFY
Sample Input 5: 全滿二元樹缺左葉節點找後序	Sample Output:
14 P ZMFGHCTEBPKAON I GFHMCTZPBKEOAN	GHFTCMPKBONAEZ
Sample Input 6: 全滿二元樹缺左葉節點找前序	Sample Output:
14 I PAKEOBGDCHFXIY O PKAOGBEHCXYIFD	DEAPKBOGFCHIXY
隱藏測資	
Sample input 1:	Sample output:
7 P ACEFDGH I ECFAGDH	EFCGHDA
Sample input 2:	Sample output:
7 I BAHDCEF O BHACFED	DABHECF
Sample input 3:	Sample output:
6 P CADFBK I DAFCKB	DFAKBC
Sample input 4:	Sample output:
6 I BDACHE O BDCEHA	ADBHCE
Sample input 5:	Sample output:

6 P ADKFBY I KDFABY	KFDYBA
Sample input 6:	Sample output:
6 I CEAGDH O ECGHDA	ACEDGH

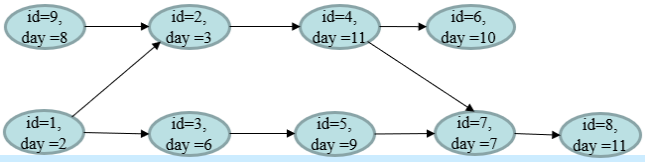
二、LinkList 開發專案(29%)(4%, 4%, 5%, 5%, 5%, 6%)

開發專案時，專案會被分割為許多項目，分配給多組程式設計師開發。

這些項目有順序關係，且只有當順序在前的項目完成，才能開始開發順序在後的項目。

本題使用一個有向無環圖，表示這些項目的開發順序，每一個節點代表一個項目。

以下圖為例，節點 2 完成後，才能開始節點 4 的開發。節點 4 與節點 5 都完成後，才能開始節點 7 的開發。節點 1 需 2 天，節點 2 需 3 天，節點 3 需 6 天，節點 4 需 11 天，節點 5 需 9 天。完成專案最少需 40 天，其中花費時間最長的路徑為 9,2,4,7,8。完成項目最多的路徑為 1,3,5,7,8 和 9,2,4,7,8。



本題須使用以下 struct 與 Link List 實作 task 才計分

```
typedef struct node_s {
int id;
int day;
struct node_s *nexts[30];
int nextCount;
} task_t;
typedef task_t *pTask;
```

輸入說明
Line 1: 正整數 M，代表輸出的類型。 M=1 代表輸出完成專案所需花費的最少時間。 M=2 代表輸出專案開發時花費時間最長的路徑。 M=3 代表輸出專案開發時完成項目最多的路徑。 Line 2: 正整數 N，代表專案共有 N 個 task。 Line 3~N+1: 從第 1 個 task 開始依序往後： 每一行 依序輸入正整數 T, K, t1~tk； T 是完成此 task 時間。 K 表示此 task 完成，後面 K 個 task 才能開始做。 t1~tk 表示等待此 task 完成的 task 編號，其中 tn+1 必定大於 tn。
輸出說明
依照正整數 M，輸出對應的結果。 *如果結果為多條路徑,選擇前面經過節點編號較小的輸出。(舉例來說，當有兩條項目最多的路徑 1,3,5,7,8 和 9,2,4,7,8，因為起始節點的編號 1 比 9 小，輸出 1,3,5,7,8)

Sample input 1:	Sample output:
1 7 2 2 2 3 3 1 4 2 1 5 4 1 6 4 1 7 3 1 7 2 0	14
Sample input 2:	Sample output:
1 5 2 2 2 3 2 1 4 6 1 5 3 1 5 2 0	10
Sample input 3:	Sample output:
2 8 3 3 2 3 4 4 1 5 2 1 6 2 1 8 2 1 8 3 1 7 4 1 8 2 0	1 3 6 7 8
Sample input 4:	Sample output:
2 7 3 2 2 3 2 2 4 5 8 1 7 2 1 6 4 1 6 3 0 3 0	1 3 7
Sample input 5:	Sample output:
3 8 1 2 2 3 2 1 4 2 1 7 4 2 5 6 3 0 2 1 8 4 0 3 0	1 2 4 6 8
Sample input 6:	Sample output:
3 9 4 2 2 3 2 1 4 1 1 8 1 2 5 6 2 1 9 2 1 7	1 2 4 5 9

3 0 9 0 1 0	
隱藏測資	
Sample input 1:	Sample output:
1 6 2 2 2 3 3 1 4 2 1 6 1 1 5 4 1 6 1 0	11
Sample input 2:	Sample output:
1 6 2 2 2 3 9 1 6 2 1 4 4 1 5 2 1 6 1 0	12
Sample input 3:	Sample output:
2 9 3 3 2 3 4 4 1 5 3 1 6 2 1 7 2 1 8 2 1 9 4 1 9 3 1 9 6 0	1 2 5 8 9
Sample input 4:	Sample output:
2 10 2 2 2 3 6 1 4 2 2 6 7 3 1 5 8 0 2 1 8 2 1 9 2 1 10 3 1 10 1 0	1 2 4 5
Sample input 5:	Sample output:
3 9 3 2 2 3 2 1 4 2 1 5 4 0 3 2 6 7 2 0 3 1 8 2 1 9 2 0	1 3 5 7 8 9

Sample input 6:	Sample output:
3 11 2 2 2 3 3 1 4 2 2 6 7 4 1 5 6 0 2 1 8 2 1 9 3 1 11 3 1 10 1 0 2 0	1 3 6 8 11

三、Double Link List(25%)(6%,6%,6%,7%)

針對空的 List，根據下方指令進行加入、刪除、插入、替換等操作。輸出操作後 Link List 資料。測試資料不會出現相同整數資料。

本題須使用以下 Link List struct 實作才計分。

```
typedef struct node_s{
int data;
struct node_s *prev,*next;
} node_t;
```

輸入說明
Line 1, 輸入操作指令的數量整數 C。1<=C<=10。 Line 2~C+1, 輸入操作指令編號整數 M 和所須整數資料 x 或 x, y。 操作指令： M=1. 從最前面加入資料 x。 M=2, 刪除最前面節點,若 List 無節點，則不必刪除。 M=3,刪除最後面節點,若 List 無節點，則不必刪除。 M=4, 刪除 List 內資料為 x 的節點；若 x 不存在則不刪除任何節點。 M=5,搜尋 List 內資料為 x 的節點,並在其前面加入數值 y 的新節點。若 x 不在 List 中,則不必加入 y 。 M=6,替換數值為 x 節點的資料為 y,若 x 不在 List 中，則不必替換。
輸出說明
從最前面節點開始依序輸出各節點的值，每個數字中間以空白間隔；若 List 為空，則輸出 "None"

Sample Input 1:	Sample Output 1:
5 1 4 1 3 1 2 1 6 1 7	7 6 2 3 4
Sample Input 2:	Sample Output 2:
10	7 6 3

1 2 1 3 1 5 2 1 6 1 8 1 7 3 4 8 4 2	
Sample Input 3:	Sample Output 3:
7 1 3 1 4 2 2 1 5 3 3	None
Sample Input 4:	Sample Output 4:
9 1 2 1 3 1 6 1 7 5 3 4 1 9 5 1 8 6 3 12 6 3 14	9 7 6 4 12 2
隱藏測資	
Sample Input 1:	Sample Output 1:
7 1 6 1 3 1 2 1 9 1 4 1 8 1 10	10 8 4 9 2 3 6
Sample Input 2:	Sample Output 1:
9 1 6 1 7 1 9 4 6 2 1 8 3 4 1 1 5	5 8

Sample Input 3:	Sample Output 1:
11 1 2 1 4 1 6 1 5 1 7 2 2 2 3 3 3	None
Sample Input 4:	Sample Output 1:
10 1 2 1 3 1 5 5 7 9 1 7 5 5 12 6 2 8 6 1 17 1 11 1 9	9 11 7 12 5 3 8

四、文章處理(25%)(6%, 6%, 6%, 7%)

輸入一個英文文章 A（空白、大小寫英文字母組合）、 2 個英文字(word-以大小寫英文字母組合) P, Q、文章分析處理指令 C，根據處理指令 C 去對文章 A 進行操作

輸入說明	輸出說明
Line 1, 輸入文章 A Line 2, 輸入字串 P Line 3, 輸入字串 Q Line 4, 輸入整數 C	C=1,在文章 A 中 P 字串前插入轉成大寫後的 Q 字串並輸出。 C=2,將文章 A 中 P 字串以轉成小寫後的 Q 字串取代並輸出。 C=3,將文章 A 中 P 字串刪除並輸出。 C=4,分析文章 A 所有英文字的頻率，依頻率由大到小排序，頻率相同則以英文字的字典順序由大到小排序(e.g. That>This....)輸出。 輸出單字與頻率中間以空白隔開

Sample Input 1:
Shell and bug danced in the bug a choreography of errors Bug bug Dog 1
Sample output 1:
Shell and DOG danced in the DOG a choreography of errors Bug

Sample Input 2
echo the echo silence a hole appear hole all sound and light hole hole AIR 2
Sample output 2:
echo the echo silence a air hole appear air hole all sound and light air hole
Sample Input 3
Yes rules yes complexity is not yes needed Simplify communicate clearly achieve more yes yes yes 3
Sample output 3:
Yes rules complexity is not needed Simplify communicate clearly achieve more
Sample Input 4
air and pod a air of modernity pod to create seamless air in AIR pod AIr POD 4
Sample output 4:
pod 3 air 3 to 1 seamless 1 of 1 modernity 1 in 1 create 1 and 1 a 1 AIR 1
隱藏測資
Sample Input 1:
An apple can provide essential nutrients while a can preserves food freshness can daN 1
Sample output 1:
An apple DAN provide essential nutrients while a DAN preserves food freshness
Sample Input 2
Bird bird freedom bird embracing the sun bird of its wings a bird of light and freedom bird DOg 2
Sample output 2:
Bird dog bird freedom dog bird embracing the sun dog bird of its wings a dog bird of light and freedom
Sample Input 3

Cryptographic key ensure secure communication across key Key key key 3
Sample output3:
Cryptographic ensure secure communication across Key
Sample Input 4
code programmer diligently worked on the code ensuring its code before base it with the project base TEST NOTHING 4
Sample output 4:
code 3 the 2 base 2 worked 1 with 1 project 1 programmer 1 on 1 its 1 it 1 ensuring 1 diligently 1 before 1

五、圖形(25%)(6%, 6%, 6%, 7%)

使用 struct 定義 shape(圖形), circle(圓), rectangle (矩形), square(正方形), triangle(三角形)。

圓有半徑，矩形有長寬，正方形有邊長，三角形有三個邊。計算各圖形周長、面積，及所有圖形周長、面積加總(三角形無需考慮是否為合法三角形)。

注意: PI 設 4。所有長度、計算均以整數處理。

本題須使用以下 struct 定義，以及定義與實做相對的 function pointer 的 function 才計分。

假設有一個三角形，邊長分別為a, b, c，三角形的面積A可由以下公式求得：

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$，其中$s = \frac{a+b+c}{2}$

```
#define shapeText(TYPE) char name[10]; \
int (*perimeter)(struct TYPE*); \
int (*area)(struct TYPE*);
typedef struct shape_s
{  shapeText(shape_s); } shape_t;
```

```
typedef struct
circle_s {
shapeText(circle_s);
int radius;
} circle_t;
```

```
typedef struct
rectangle_s {
shapeText(rectangle_s
);
int width, height;
} rectangle _t;
```

```
typedef struct
square_s{
shapeText(square_s);
int side;
} square_t;
```

```
typedef struct
triangle_s {
shapeText(triangle_s)
;
int s1, s2, s3;
```

	} triangle_t;

輸入說明	輸出說明
Line 1, 輸入圖形個數 N。 1<=N<=10 Line 2, 輸入圖形種類。 Line 3, 輸入圖形所需資料。 Line 4, 輸入圖形種類。 Line 5, 輸入圖形所需資料。 …. … .. circle，下一行輸入半徑。 rectangle，下一行輸入長寬。 square，下一行輸入邊長。 triangle，下一行輸入三個邊。	Line 1~N: 輸出 N 個圖形的種類、周長與面積，以空白間隔。接著以周長大到小排序後輸出，若周長相同，則以面積大到小排序輸出。 Line N+1: 輸出 N 個圖形的周長總和、面積總和，以空白間隔。

Sample Input 1:
3 circle 10 rectangle 5 6 square 9
Sample output 1:
circle 80 400 square 36 81 rectangle 22 30 138 511
Sample Input 2
2 circle 2 triangle 3 4 5
Sample output 2:
circle 16 16 triangle 12 6 28 22
Sample Input 3
4 square 10 triangle 13 5 12 circle 1 rectangle 8 20

Sample output 3:
rectangle 56 160 square 40 100 triangle 30 30 circle 8 4 134 294
Sample Input 4
2 triangle 3 4 5 rectangle 2 4
Sample output 4:
rectangle 12 8 triangle 12 6 24 14
隱藏測資
Sample Input 1:
5 circle 6 circle 1 circle 9 triangle 6 8 10 rectangle 10 3
Sample output 1:
circle 72 324 circle 48 144 rectangle 26 30 triangle 24 24 circle 8 4 178 526
Sample Input 2
1 circle 5
Sample output 2:
circle 40 100 40 100
Sample Input 3
3 square 2 square 3 circle 9
Sample output3:
circle 72 324 square 12 9 square 8 4 92 337
Sample Input 4
3 square

1 triangle 3 4 5 rectangle 2 4
Sample output 4:
rectangle 12 8 triangle 12 6 square 4 1 28 15

六、工作排程(26%)(5%, 5%, 5%, 5%, 6%)

有 M 個工作要在 N 台機器上加工，每個工作 i 包含若干個工序 Oij，這些工序須依序加工，也就是前一道工序 Oi(j-1)完成後才可開始下一道工序 Oij。

每道工序 Oij 可用一個有序對(Kij,Tij)表示它需在機器 Kij 上面花費 Tij 小時完成，而每台機器一次只能處理一道工序。

所謂一道工序 Oij 的「最早完成時間的 Cij 」是指考慮目前排程中機器 Kij 之可用性以及前一道工序 Oi(j-1)(若該工序存在)之完成時間後可得的最早完成時間。

所有工序的排程規則如下：

針對每一個工作的第一個尚未排程的工序，計算出此工序的「最早完成時間」，然後挑選出最早完成時間最小的工序納入排程中，如果有多個完成時間都是最小，則挑選其中最小工作編號之工序，一個工序一旦納入排程就不會再更改，重複以上步驟直到所有工序皆納入排程。

輸入說明
Line 1: 正整數 N 與 M，N 代表有多少台機器，M 代表有多少個工作。 接下來會有 M 個工作資訊 ， 輸入順序即為工作編號順序，每個工作資訊有兩行，第一行為整數 P，代表到工序數量。 Line 2: 有 2 * P 個整數，依序每兩個一組代表一道工序的機器編號與需求時間，每個整數間以空白符號相隔開。
輸出說明
輸出每個工作完成時間的總和

Sample Input 1:	Sample Output 1:
4 4 3 0 3 0 5 0 8 2 1 2 1 6 4 2 3 2 4 2 1 2 1 3 3 1 3 2 3 3	39
Sample Input 2:	Sample Output 2:
3 3 1 2 8 1 2 4 1 2 9	37
Sample Input 3:	Sample Output 3:
4 4 3 0 3 1 8 1 1 4 0 6 0 1 1 7 1 8 1 2 6 3 2 1 2 2 2 2	55
Sample Input 4:	Sample Output 4:
4 3 3 0 3 2 6 1 5 2 0 4 2 5 1 0 6	41
Sample Input 5:	Sample Output 5:
1 5 3 0 1 0 2 0 3 10 0 1 0 5 0 9 0 1 0 3 0 8 0 1 0 3 0 10 0 9 1 0 40 2 0 9 0 46 5 0 11 0 22 0 7 0 1 0 3	525
隱藏測資	
Sample Input 1:	Sample Output 1:
3 3 2 0 2 0 4 3 1 2 1 3 1 5 2 2 2 2 5	23
Sample Input 2:	Sample Output 2:
4 2 1 0 3 1 0 5	11
Sample Input 3:	Sample Output 3:
4 4 3 0 2 1 3 1 4 4 0 3 0 2 1 5 1 6 1 2 4 2 2 2 2 1	39
Sample Input 4:	Sample Output 4:
3 3 3 0 1 2 4 1 3	23

3 0 2 2 2 1 1 1 0 3	
Sample Input 5:	Sample Output 5:
1 5 3 0 2 0 4 0 6 10 0 2 0 5 0 7 0 9 0 10 0 9 0 7 0 5 0 12 0 7 1 0 30 2 0 14 0 36 5 0 17 0 23 0 7 0 3 0 5	662

七、LinkList 多項式相加、相減、相乘(28%)(5%,5%, 6%, 6%, 6%)

(1) void add(pol_t* X, pol_t* Y, pol_t* Z);
//兩個多項式 X, Y 相加，Z 是結果
(2) void sub(pol_t* X, pol_t* Y, pol_t* Z);
//兩個多項式 X, Y 相減，Z 是結果
(3) void mul(pol_t* X, pol_t* Y, pol_t* Z);
//兩個多項式 X, Y 相乘，Z 是結果
本題須使用以下 struct 與 Link List 實作才計分。

```
typedef struct node_s {
int coef;
int exp;
struct node_s * next;
} node_t;
typedef node_t * nodep_t;
typedef pol_s {
nodep_t root;
} pol_t
```

輸入說明
Line 1: 多項式 X 的項數個數 N1。 Line 2: 多項式 X 的 N1 個係數(整數),第 1 個是 n-1 次方的係數,...,第 n 個代表 0 次方係數。 Line 3: 多項式 Y 的項數個數 N2。 Line 4: 多項式 Y 的 N2 個係數(整數),第 1 個是 n-1 次方的係數,...,第 n 個代表 0 次方係數。 例 2 0 3 4 1 -1 3,為 2x^6 + 3x^4 + 4x^3 + x^2 - x + 3
輸出說明
Line 1, X, Y 多項式相加結果 Line 2, X, Y 多項式相減結果 Line 3, X, Y 多項式相乘結果 (從最高次方到 0 次方的係數與次方)

Sample Input 1:	Sample Output 1:
5 2 3 0 1 -1 6 1 0 -1 4 -3 2	1 5 2 4 2 3 4 2 -2 1 1 0 -1 5 2 4 4 3 -4 2 4 1 -3 0 2 9 3 8 -2 7 6 6 5 5 -6 4 11 3 -7 2 5 1 -2 0
Sample Input 2:	Sample Output 2:
6 1 0 -1 4 -3 2 2 1 1	1 5 0 4 -1 3 4 2 -2 1 3 0 1 5 0 4 -1 3 4 2 -4 1 1 0 1 6 1 5 -1 4 3 3 1 2 -1 1 2 0
Sample Input 3:	Sample Output 3:
4 9 -8 3 -2 4 11 3 -4 2	20 3 -5 2 -1 1 0 0 -2 3 -11 2 7 1 -4 0 99 6 -61 5 -27 4 37 3 -34 2 14 1 -4 0
Sample Input 4:	Sample Output 4:
5 1 2 3 4 5 5 -1 -2 -3 -4 -5	0 4 0 3 0 2 0 1 0 0 2 4 4 3 6 2 8 1 10 0 -1 8 -4 7 -10 6 -20 5 -35 4 -44 3 -46 2 -40 1 -25 0
Sample Input 5:	Sample Output 5:
5 5 4 3 2 1 5 5 4 3 2 1	10 4 8 3 6 2 4 1 2 0 0 4 0 3 0 2 0 1 0 0 25 8 40 7 46 6 44 5 35 4 20 3 10 2 4 1 1 0
隱藏測資	
Sample Input 1:	Sample Output 1:
3 1 3 5 6 6 5 0 3 0 1	6 5 5 4 0 3 4 2 3 1 6 0 -6 5 -5 4 0 3 -2 2 3 1 4 0 6 7 23 6 45 5 28 4 9 3 16 2 3 1 5 0
Sample Input 2:	Sample Output 2:
5 -2 -3 6 0 1 2 2 2	-2 4 -3 3 6 2 2 1 3 0 -2 4 -3 3 6 2 -2 1 -1 0 -4 5 -10 4 6 3 12 2 2 1 2 0
Sample Input 3:	Sample Output 3:
3 7 9 10 3	4 2 14 1 17 0 10 2 4 1 3 0

-3 5 7	-21 4 8 3 64 2 113 1 70 0
Sample Input 4:	Sample Output 4:
4 -4 -3 -2 -1 4 4 3 2 1	0 3 0 2 0 1 0 0 -8 3 -6 2 -4 1 -2 0 -16 6 -24 5 -25 4 -20 3 -10 2 -4 1 -1 0
Sample Input 5:	Sample Output 5:
4 1 1 1 1 4 1 1 1 1	2 3 2 2 2 1 2 0 0 3 0 2 0 1 0 0 1 6 2 5 3 4 4 3 3 2 2 1 1 0