

在此次作業附上兩個檔案，分別是：classA.txt 及 classB.txt。分別表示：

classA.txt：為 A 班同學的生日資料表。其中，每一列記著一位同學的資料。No 代表學號，Name 代表姓名，Birthday 包含有出生年、月、日〔例：2002/1/2 表示 2002 年 1 月 2 日〕。

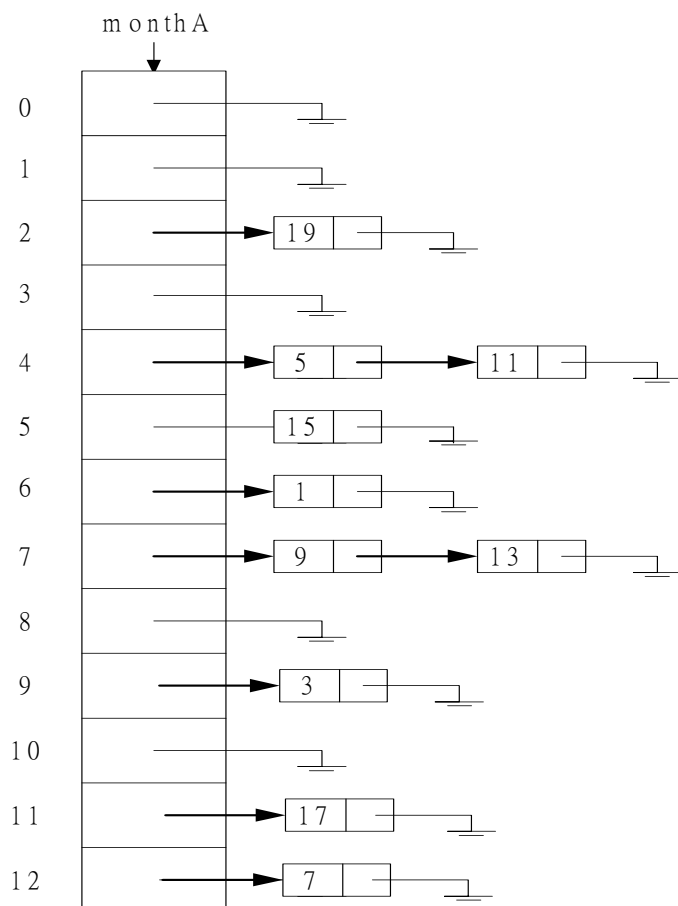
classB.txt：為 B 班同學的生日資料表。解讀方式如同 classA.txt

首先，Structure Definition 定義一個新的型別〔名為student〕。其中 id 表示某一個學生的學號，birthMonth表示該生在何月份出生。如下段程式碼所示：

```
struct personalData{
    int id;
    int birthMonth;
    struct personalData *link;
};

typedef struct personalData student;
```

1. 使用上述的所定義的型別，配合著 linked list 的結構，將 classA.txt 中的資料以下圖結構存入程式中：



上圖中 monthA 是一個長度為 13、型別為 (student*)的一維陣列。其中：

monthA[0] = 0 〔不使用此元素〕

monthA[1]指向生日為 1 月的 A 班同學所構成之 linked list

monthA[2]指向生日為 2 月的 A 班同學所構成之 linked list

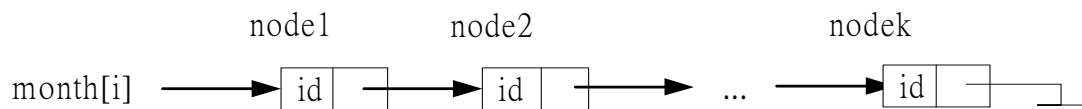
monthA[3]指向生日為 3 月的 A 班同學所構成之 linked list

·
·
·

monthA[12]指向生日為 12 月的 A 班同學所構成之 linked list

而 linked list 中的每一個節點的型別為 student。若 A 班同學沒人在某一月份〔例 i 月〕出生，則設 monthA[i] = 0。

而最後所有的 linked list 中的元素均需以”學號”順序串接。舉例來說，若一個 list 中有 k 個 node 如下圖〔node1 ~ nodek〕：

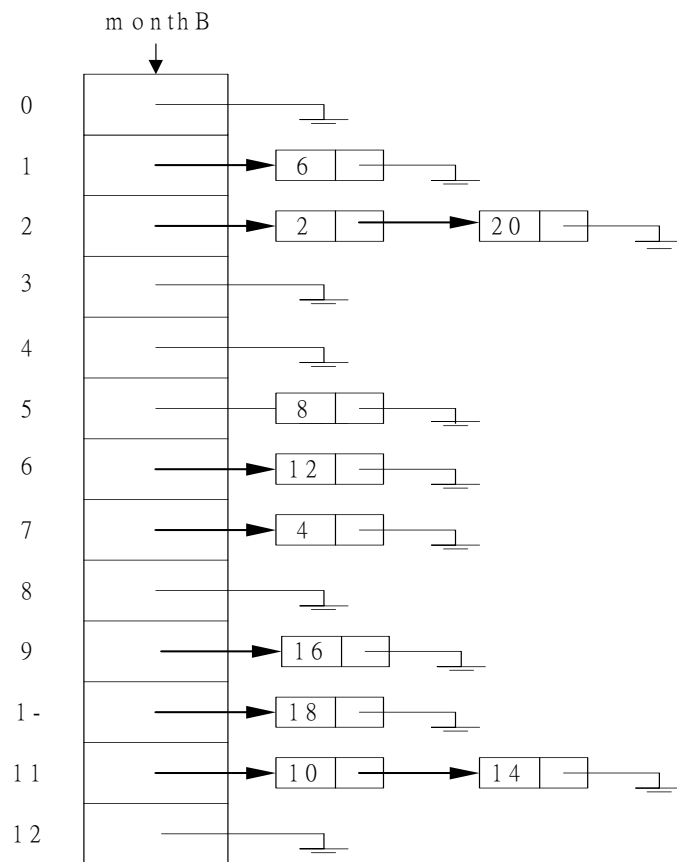


若以”學號”順序串接，則 $node1.id \leq node2.id \leq \dots \leq nodek.id$ 。

最後，將結果以下列方式輸出：

month	students' id
1	
2	19
3	
4	5,11
5	15
6	1
7	9,13
8	
9	3
10	
11	17
12	7

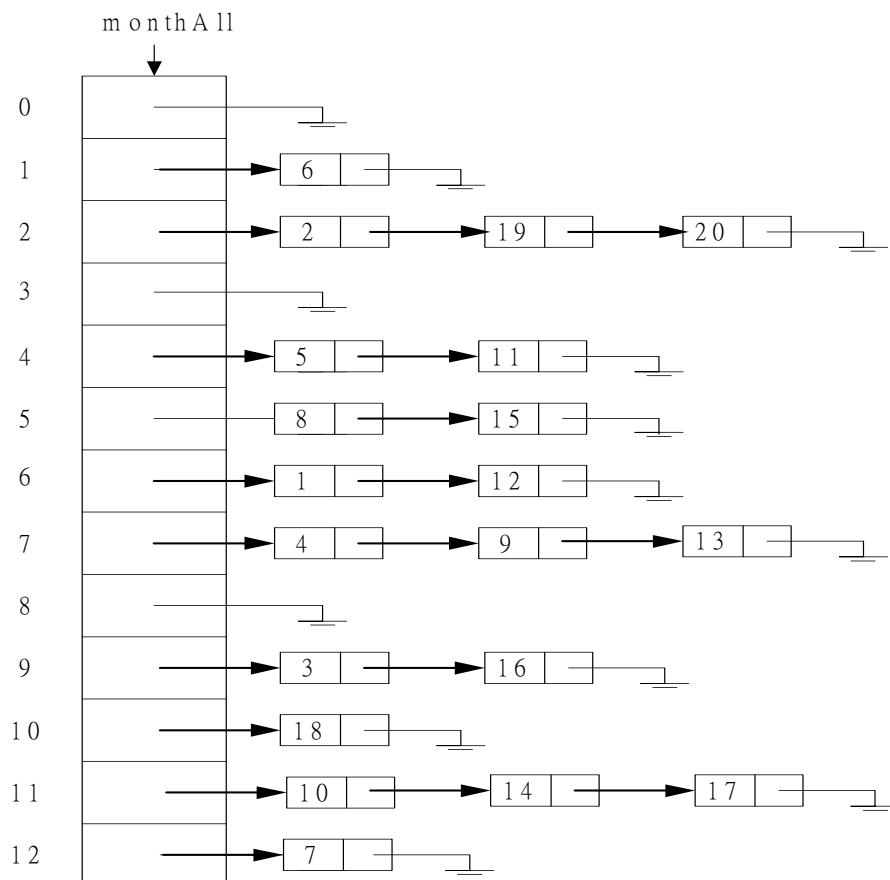
2. 以同樣的規則，將 classB.txt 的資料存入程式中，如下圖：



最後，將結果以同樣的方式輸出：

month	students' id
1	6
2	2,20
3	
4	
5	8
6	12
7	4
8	
9	16
10	18
11	10,14
12	

3. 在 1,2 小題所得 monthA 及 monthB 內所含的資料，以同樣的規則將 monthA 及 monthB 合併成下圖的結構：



最後，將上述的結果再經由程式的處理，使此程式能以下述的兩種方式〔如 3.1 及 3.2 所述〕輸出：

- 3.1 使用者輸入一個代表月份的整數，程式會由所存的資訊來輸出在該月份出生同學的學號。如下範例所示：

```
Input a month: 2
=====
2,19,20
```

```
Input a month: 3
=====
No student was born in this month!
```

3.2 程式會由所存的資訊，以下列方式輸出至一個名為 result.txt 的文字檔。

No	birth month
1	6
2	2
3	9
4	7
5	4
6	1
7	12
8	5
9	7
10	11
11	4
12	6
13	7
14	11
15	5
16	9
17	11
18	10
19	2
20	2

其中，每一列是一位學生的資料。id 欄位表示學號，birth month 欄位表示出生月份。