

C-Programming Mid-Term 2

Department of Computer Science and Engineering, National Sun Yat-sen University.

2018/11/13

Note:

- (1) 程式碼全部撰寫在同一個 .c 檔中。
- (2) 請存成 .c 檔，檔名用自己的學號命名。
- (3) 請勿更改給定的 main function，若更動 main function 導致程式無法執行，後果自負。
- (4) 輸出格式須與範例“一模一樣”，勿多寫跟少寫，否則會斟酌扣分。
- (5) 助教批改時會使用新安裝的 Devcpp 進行編譯，若在考試時對編譯器進行更動或使用非預設之標頭檔，後果自負。
- (6) 對 1 題 20 分，對 2 題 40 分，對 3 題 60 分，對 4 題 75 分，對 5 題 85 分，全對 100 分

1. 請設計一程式印出你的學號及姓名

Please design a program to print your student ID and name.

Name: 王小明

Student ID: M073040000

請按任意鍵繼續 ...

2. 救救廖助教 Rescue Assistant Liao

可憐的助教在改作業，請設計一個程式幫助教統計分數

Poor assistant is grading homeworks, please design a program to help assistant statistical scores.

- 輸入資料: 請參考 input3.txt 及 Fig 3 (示意圖)
- 輸出資料: 輸出檔名為 input4.txt，格式請參考 Fig 4.(示意圖)
- 每位學生有八個作業，每個學生的作業的題數不一定(但是只會是 4 題或 5 題)
- 藍色框代表座號
- 紅色框代表題數
- 綠色框代表成績(-1 代表沒交 -2 代表抄襲 其他代表成績)
- 若有成績者，請將成績加總後取平均(無條件捨去)

➤ 注意!!助教批改時，會使用其他測資(格式一樣)

- Input data: please refer to input3.txt and Fig 3(example)
- Output data: Output file name is "input4.txt", please refer to Fig 4
- Every student has 8 homework, and has different sub questions in each homework(4 or 5 problem)
- Blue Box means student's ID
- Red Box means the number of sub question in this homework
- Green Box means grades(-1 means didn't hand in, -2 means plagiarism, others means grades)
- If the student has scores in this homework, please average the scores

➤ Warning!! TA will use the other test data(format will be same)

input3 - 記事本

檔案(F) 編輯(E) 格式

```

1
4 -2
4 22 38 9 37
4 -1
4 47 38 32 39
5 79 92 19 72 35
5 80 75 99 65 5
4 32 98 48 79
4 -2
2
4 -2
4 74 39 89 45
4 82 92 39 56
4 43 70 27 74
4 37 62 100 51
4 -2
4 59 6 6 37
5 62 29 89 24 79
3
4 -2
5 -1
5 -1
4 38 62 55 43
5 -2
5 4 48 54 24 2
5 7 42 21 22 83
5 92 27 3 40 86
4
5 26 91 79 34 98
5 7 69 42 87 56
5 68 23 48 25 41
5 88 84 59 28 6
5 34 61 78 6 39
5 89 47 4 66 42
5 64 70 37 6 74
4 67 26 55 66

```

Fig 3

input4 - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V)

```

1 -2:26:-1:39:59:64:64:-2
2 -2:61:67:53:62:-2:27:56
3 -2:-1:-1:49:-2:26:35:49
4 65:52:41:53:43:49:50:53
5 45:-1:44:41:62:43:66:54
6 -2:53:35:-1:-2:39:47:-2
7 82:49:47:-1:94:42:51:55
8 -1:62:-2:-2:22:47:48:62
9 40:28:77:56:46:46:48:60
10 49:-1:49:-2:45:47:-2:68
11 69:49:17:-1:-1:34:49:34
12 62:48:36:57:49:59:-2:-2
13 43:28:-1:-1:58:54:-2:64
14 38:63:51:60:42:49:65:60
15 63:70:33:50:40:-2:56:61
16 16:-1:-1:57:-1:52:43:28
17 18:38:-1:-2:43:-1:22:40
18 66:50:33:-2:53:51:70:-2
19 24:35:47:63:34:31:43:39
20 33:-2:75:55:36:33:49:-2
21 68:-2:67:-2:43:36:-1:67
22 47:49:37:66:47:24:37:64
23 33:52:41:58:85:-2:57:49
24 63:76:55:-2:59:-2:48:85
25 26:61:-2:64:40:48:62:70
26 -1:46:57:42:-1:60:-2:72
27 47:-1:60:30:38:55:50:-2
28 -1:-2:-2:64:59:-1:40:53
29 28:72:48:59:64:-2:-2:-2
30 36:18:54:41:39:24:57:-2
31 52:34:43:-1:60:28:60:55
32 50:57:63:58:-2:77:30:-1
33 60:68:42:-2:42:58:58:47
34 -2:53:43:62:34:60:34:52
35 40:47:56:33:36:77:-1:57
36 -2:-2:-2:33:46:54:33:69
37 29:58:43:-2:58:61:-2:61
38 66:74:69:-2:-1:-2:58:-2
39 31:-1:-1:44:35:52:41:66
40 34:59:-2:49:65:38:70:-1

```

Fig 4

3. 學生作業成績管理系統(Students' homework score management system)

請設計一成績管理系統，依照輸入之資料及規則，計算總成績，規則如下：

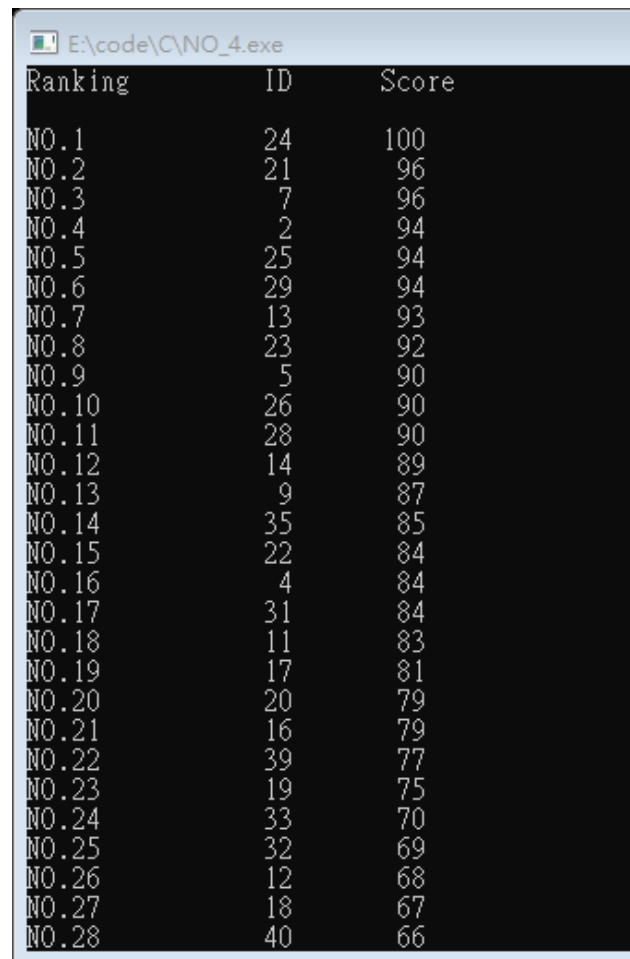
- A. 總成績為全學期 8 次作業中最高的 5 次成績之平均
- B. 缺繳者(成績為-1)當次作業以 0 分計算
- C. 抄襲者(成績為-2)當次作業以 0 分計算，並且總成績改為全學期 8 次作業中最低的 5 次成績之平均
- D. 總成績一律取整數
- E. 最後調整分數：計算完總成績後，找到總成績最高者，計算該成績與 100 分之差即為每個人加上的分數。

Please design a score management system to calculate the total score according to the input information and rules. The rules are as follows:

- A. The total score is the average of the top 5 scores of the 8 assignments throughout the semester.
- B. Missing (with a score of -1), the current assignment is calculated with 0 points
- C. Plagiarism (with a score of -2), the current assignment is calculated with 0 points, and the total score is changed to the average of the lowest 5 of the 8 assignments in the whole semester.
- D. The total score is always an integer
- E. Final adjustment score: After calculating the total score, find the highest total score, and calculate the difference between the score and 100 points, which is the score added for each person.

輸入資料：為上一題之 input4.txt

輸出資料：請參考 Fig 5



Ranking	ID	Score
NO.1	24	100
NO.2	21	96
NO.3	7	96
NO.4	2	94
NO.5	25	94
NO.6	29	94
NO.7	13	93
NO.8	23	92
NO.9	5	90
NO.10	26	90
NO.11	28	90
NO.12	14	89
NO.13	9	87
NO.14	35	85
NO.15	22	84
NO.16	4	84
NO.17	31	84
NO.18	11	83
NO.19	17	81
NO.20	20	79
NO.21	16	79
NO.22	39	77
NO.23	19	75
NO.24	33	70
NO.25	32	69
NO.26	12	68
NO.27	18	67
NO.28	40	66

Fig 5

4. 讓我們定義正整數 S_0 中每個數字的平方和為 S_1 。以相同的方法我們定義 S_1 中每個數字的平方和為 S_2 ，並依此類推。假如有某個 $S_i = 1$ ($i \geq 1$) 則我們說 S_0 是一個 Happy number。如果某一個數不是 Happy number，那他也就是一個 Unhappy number。

例如：7 是一個 Happy number，因為 $7 \rightarrow 49 \rightarrow 97 \rightarrow 130 \rightarrow 10 \rightarrow 1$ 。

但是 4 是一個 Unhappy number，因為 $4 \rightarrow 16 \rightarrow 37 \rightarrow 58 \rightarrow 89 \rightarrow 145 \rightarrow 42 \rightarrow 20 \rightarrow 4$ ，永遠也無法產生 1。

輸入說明

輸入的第一列有一個整數代表以下有多少組測試資料

每組測試資料一列含有 1 個正整數 N ($N < 10^9$)

輸出說明

對每組測試資料輸出一列

輸出 N 為 Happy number 或 Unhappy number

請參考 Sample Output

範例輸入

3

7

4

13

範例輸出

Case #1: 7 is a Happy number.

Case #2: 4 is an Unhappy number.

Case #3: 13 is a Happy number.

Let the sum of the square of the digits of a positive integer S_0 be represented by S_1 . In a similar way, let the sum of the squares of the digits of S_1 be represented by S_2 and so on. If $S_i = 1$ for some $i \geq 1$, then the original integer S_0 is said to be Happy number. A number, which is not happy, is called Unhappy number. For example, 7 is a Happy number since $7 \rightarrow 49 \rightarrow 97 \rightarrow 130 \rightarrow 10 \rightarrow 1$ and 4 is an Unhappy number since $4 \rightarrow 16 \rightarrow 37 \rightarrow 58 \rightarrow 89 \rightarrow 145 \rightarrow 42 \rightarrow 20 \rightarrow 4$.

Input

The input consists of several test cases, the number of which you are given in the first line of the input.

Each test case consists of one line containing a single positive integer N smaller than 10^9 .

Output

For each test case, you must print one of the following messages:

Case #p: N is a Happy number.

Case #p: N is an Unhappy number.

Here p stands for the case number (starting from 1). You should print the first message if the number N is a happy number. Otherwise, print the second line.

Sample Input

3
7
4
13

Sample Output

Case #1: 7 is a Happy number.

Case #2: 4 is an Unhappy number.

Case #3: 13 is a Happy number.

5. 電腦無法產生真正的亂數 (Random numbers)，但是經由某些程序電腦可以產生虛擬亂數 (pseudo-random numbers)。亂數被使用在很多應用上，像是模擬等。

有一種常用的虛擬亂數產生方法：如果上一個亂數是 L ，那下一個亂數產生的方法是

$(Z \times L + I) \bmod M$ ，在這裡 Z 、 I 、 M 都是常數。例如：假設 $Z=7$ $I=5$ $M=12$ 。如果第一個亂數 (通常叫做 seed) 是 4，那我們可以產生以下幾個虛擬亂數：

Last Random Number, L	$(Z \times L + I)$	Next Random Number, $(Z \times L + I) \bmod M$
4	33	9
9	68	8
8	61	1
1	12	0
0	5	5
5	40	4

我們可以發現，經過 6 個數字後，虛擬亂數的序列重複了，也就是說 cycle length=6。在這個問題中，你將會被給 Z 、 I 、 M 還有 L (就是 seed) 的值 (全部不大於 9999)，對每一組 Z 、 I 、 M 、 L ，要請你輸出所產生的虛擬亂數的 cycle length。請注意：cycle 並不一定從 seed 開始。

輸入說明

輸入的每一行有 4 個整數，依序為 Z , I , M , L 。(L 一定比 M 小)
輸入的最後一行為 4 個 0，代表輸入結束。

輸出說明

對每一行輸入，輸出這是第幾組測試資料 (連續數字，從 1 開始) 和所產生的虛擬亂數的 cycle length。

範例輸入

```
7 5 12 4
5173 3849 3279 1511
9111 5309 6000 1234
1079 2136 9999 1237
0 0 0 0
```

範例輸出

```
Case 1: 6
Case 2: 546
Case 3: 500
Case 4: 220
```


Computers normally cannot generate really random numbers, but frequently are used to generate sequences of pseudo-random numbers. These are generated by some algorithm, but appear for all practical purposes to be really random. Random numbers are used in many applications, including simulation.

A common pseudo-random number generation technique is called the linear congruential method. If the last pseudo-random number generated was L , then the next number is generated by evaluating $(Z \times L + I) \bmod M$, where Z is a constant multiplier, I is a constant increment, and M is a constant modulus. For example, suppose Z is 7, I is 5, and M is 12. If the first random number (usually called the seed) is 4, then we can determine the next few pseudo-random numbers are follows:

Last Random Number, L	$(Z \times L + I)$	Next Random Number, $(Z \times L + I) \bmod M$
4	33	9
9	68	8
8	61	1
1	12	0
0	5	5
5	40	4

As you can see, the sequence of pseudo-random numbers generated by this technique repeats after six numbers. It should be clear that the longest sequence that can be generated using this technique is limited by the modulus, M . In this problem you will be given sets of values for Z , I , M , and the seed, L . Each of these will have no more than four digits. For each such set of values you are to determine the length of the cycle of pseudo-random numbers that will be generated. But be careful: the cycle might not begin with the seed!

Input

Each input line will contain four integer values, in order, for Z , I , M , and L . The last line will contain four zeroes, and marks the end of the input data. L will be less than M .

Output

For each input line, display the case number (they are sequentially numbered, starting with 1) and the length of the sequence of pseudo-random numbers before the sequence is repeated.

Sample Input

7 5 12 4
5173 3849 3279 1511
9111 5309 6000 1234
1079 2136 9999 1237
0 0 0 0

Sample Output

Case 1: 6
Case 2: 546
Case 3: 500
Case 4: 220

6. 把一個數字反轉並相加的方法很簡單：就是把數字反轉並加上原來的數字。假如這個和不是一個迴文（指這個數字從左到右和從右到左都相同），就一直重複這個程序。舉例說明：

195 開始的數字
591

786
687

1473
3741

5214
4125

9339 迴文出現了

在這個例子中，經過了 4 次相加後得到了迴文 9339。幾乎對所有的整數這個方法都會得到迴文，但是也有有趣的例外。196 是第 1 個用這個方法找不到迴文的數字，然而並沒有證明該迴文不存在。

現在給你一個開始的數字，你的任務就是求出經過多少次相加後，會產生哪一個迴文。對所有的測試資料，你可以假設：1. 都會有 1 個答案。2. 在 1000 個相加內都會得到答案。3. 產生的迴文不會大於 4294967295.

輸入說明

第 1 列有一個整數 N ($0 < N \leq 100$)，代表以下有幾組測試資料。每筆測試資料一行，各有 1 個整數 P ，就是開始的數字。

輸出說明

對每一測試資料，請輸出 2 個數字：得到迴文所需的最少次數的相加，以及該迴文。

範例輸入

```
5
195
265
750
2
99
```

範例輸出

```
4 9339
5 45254
3 6666
1 4
6 79497
```

The “reverse and add” method is simple: choose a number, reverse its digits and add it to the original.

If the sum is not a palindrome (which means, it is not the same number from left to right and right to left), repeat this procedure.

For example:

195 Initial number

591

—

786

687

—

1473

3741

—

5214

4125

—

9339 Resulting palindrome

In this particular case the palindrome ‘9339’ appeared after the 4th addition. This method leads to palindromes in a few step for almost all of the integers. But there are interesting exceptions. 196 is the first number for which no palindrome has been found. It is not proven though, that there is no such a palindrome.

You must write a program that give the resulting palindrome and the number of iterations (additions) to compute the palindrome.

You might assume that all tests data on this problem:

- will have an answer ,
- will be computable with less than 1000 iterations (additions),
- will yield a palindrome that is not greater than 4,294,967,295.

Input

The first line will have a number N ($0 < N \leq 100$) with the number of test cases, the next N lines will

have a number P to compute its palindrome.

Output

For each of the N tests you will have to write a line with the following data :
minimum number of iterations(additions) to get to the palindrome
and the resulting palindrome itself separated by one space.

Sample Input

3

195

265

750

Sample Output

4 9339

5 45254

3 6666

Main Function

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <time.h>
5
6  void Question1();
7  void Question2();
8  void Question3();
9  void Question4();
10 void Question5();
11 void Question6();
12
13
14 int main(){
15     srand(time(NULL));
16     int i;
17     while(1){
18         printf("Please enter the number of question(0 for exit): ");
19         scanf("%d",&i);
20
21         if (i==0) break;
22
23         switch(i){
24             case 1:
25                 Question1();
26                 break;
27             case 2:
28                 Question2();
29                 break;
30             case 3:
31                 Question3();
32                 break;
33             case 4:
34                 Question4();
35                 break;
36             case 5:
37                 Question5();
38                 break;
39             case 6:
40                 Question6();
41                 break;
42             default:
43                 printf("Undefined number!!");
44         }
45     }
46 }
```