# 2_涼子電腦 (Coool Computing)

***(10分)***

## 題目敘述

由$100^{200} < 99^{202} < \ldots n^{\lfloor 20000/n \rfloor} \ldots < 4^{5000} < 3^{6666} > 2^{10000}$, 可以知道三位元是最少狀態來表達最多的數字, 同時利用類比訊號的正電流 (+1)、負電流 (-1) 和近0電流(0), 也可以更有效率的 (更少的電子元件) 來儲存、運算或是傳輸資料, 再加上與二進位的差異會更加安全, 或是在二進位上提供錯誤檢查或是metadata (增加附屬屬性, 如時間), 以及適用於表達三軸 $(x, y, z)$ 的空間象限, 或是 RGB 的色彩的情況上。

然而在人機介面上仍需要將 3 進位轉為 10 進位, 所以我們每 **11 個 3 進位** $00000000000$ - $12002011200$ 來表示 **5 個 10 進位** ($00000$ - $99999$), 因為在**合理值範圍內最高位 (最左邊) 不會是 2, 所以我們用它做安全性的檢查碼**, 將它設為 2 的規則, 依11 個 3 進位的數字和取 3 的餘數(n值), 再看是否出現位置是對的筆數, 這樣我們除了可以檢查數字是否有錯外, 還可以避免中間資料被增加或篡改。所以設為 2 的情況有 3 種規則：

- 規則1：第 1, 4, 7, ... 筆時, 將 11 個 3 進位數字各別加起來取3 的餘數, 如果是 0是, 則將最高位設為 2。
- 規則2：第 2, 5, 8, ... 筆時, 將 11 個 3 進位數字各別加起來取3 的餘數, 如果是 1是, 則將最高位設為 2。
- 規則3：第 3, 6, 9, ... 筆時, 將 11 個 3 進位數字各別加起來取3 的餘數, 如果是 2是, 則將最高位設為 2。

以下是含檢查碼的11位3進位所代表的符合規則的10進位值: ($00000 - 99999$ 任何值依出現位置都可能設為 2)

| **規則1:餘是0時,高位元設為2** | **規則2: 餘是1時,高位元設為2** | **規則3:餘是2時,高位元設為2** |
|---|---|---|
| 20000000000 -> 0 | 00000000000 -> 0 | 00000000000 -> 0 |
| 00000000001 -> 1 | 20000000001 -> 1 | 00000000001 -> 1 |
| 00000000002 -> 2 | 00000000002 -> 2 | 20000000002 -> 2 |
| 00000000010 -> 3 | 20000000010 -> 3 | 00000000010 -> 3 |
| 00000000011 -> 4 | 00000000011 -> 4 | 20000000011 -> 4 |
| 20000000012 -> 5 | 00000000012 -> 5 | 00000000012 -> 5 |
| ... | ... | ... |
| 10000000000 -> 59049 | 20000000000 -> 59049 | 10000000000 -> 59049 |
| 10000000001 -> 59050 | 10000000001 -> 59050 | 20000000001 -> 59050 |
| 20000000002 -> 59051 | 10000000002 -> 59051 | 10000000002 -> 59051 |
| 10000000010 -> 59052 | 10000000010 -> 59052 | 20000000010 -> 59052 |
| 20000000011 -> 59053 | 10000000011 -> 59053 | 10000000011 -> 59053 |
| 10000000012 -> 59054 | 20000000012 -> 59054 | 10000000012 -> 59054 |
| ... | ... | ... |
| 12002011111 -> 99994 | 22002011111 -> 99994 | 12002011111 -> 99994 |
| 12002011112 -> 99995 | 12002011112 -> 99995 | 22002011112 -> 99995 |
| 12002011120 -> 99996 | 22002011120 -> 99996 | 12002011120 -> 99996 |
| 12002011121 -> 99997 | 12002011121 -> 99997 | 22002011121 -> 99997 |
| 22002011122 -> 99998 | 12002011122 -> 99998 | 12002011122 -> 99998 |
| 22002011200 -> 99999 | 12002011200 -> 99999 | 12002011200 -> 99999 |

試寫一支程式排除錯誤資料後輸出總和。

## 輸入格式

第一行輸入為資料的筆數，剩下每一行都是一組 3 進位值

# 輸出格式

不含錯誤資料的10 進位總和值, 總和值小於等於 $99999000$

# 資料範圍

- 輸入資料範圍第一行(筆數) 值小於 1000
- 後面的每一行值介於 $00000000000 - 2222222222$

# 測試範例

## 輸入範例 1

```
5
00000000000
00000000001
00000000002
22222222222
20000000001
```

## 輸出範例 1

```
1
```

## 輸入範例 2

```
9
20000000000
20000000001
20000000002
00000000010
00000000011
00000000012
00000000020
00000000021
00000000022
```

## 輸出範例 2

```
36
```

## 輸入範例 3

```
11
20000000120
20000000121
20000000122
22002011200
22002011200
22002011200
22002011201
22002011201
22002011201
12222222222
12222222222
```

## 輸出範例 3

```
181948
```

# 範例說明

## 範例 1

00000000000 => 因是第 1筆, n=0, 所以 00000000000 應該是20000000000 才正確

00000000001 => 因是第 2筆, n=1, 所以 00000000001 應該是20000000001 才正確

00000000002 => 因是第 3 筆, n=2, 所以 00000000002 應該是20000000002 才正確

22222222222 => 因是第 4筆, n=0, 所以 22222222222 可轉為 12222222222 但其值轉為 10 進位時超過 99999

20000000001 => 因是第 5筆, n=1正確, 所以 20000000001 可轉為 00000000001, 十進位值為1

所以 4 筆資料是錯的, 所有合理值的十進位總和為 1

## 範例 2

20000000000, 因是第1筆, n=0, 合理轉為 0,

20000000001, 因是第2筆, n=1, 合理轉為 1,

20000000002, 因是第3筆, n=2, 合理轉為 2,

00000000010, 因是第4筆, n=0, 合理轉為 3, 3進位10=十進位3, $(1 \times 3 + 0 = 3)$

00000000011, 因是第5筆, n=1, 合理轉為 4, 3進位11=十進位4, $(1 \times 3 + 1 = 4)$

00000000012, 因是第6筆, n=2, 合理轉為 5, 3進位12=十進位5, $(1 \times 3 + 2 = 5)$

00000000020, 因是第7筆, n=0, 合理轉為 6, 3進位12=十進位6, $(2 \times 3 + 0 = 6)$

00000000021, 因是第8筆, n=1, 合理轉為 7, 3進位12=十進位7, $(2 \times 3 + 1 = 7)$

00000000022, 因是第9筆, n=2, 合理轉為 8, 3進位12=十進位8, $(2 \times 3 + 2 = 8)$

所以和是 $0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36$