JT-63 1 到 6 的排列

題目

將 1 到 6 的所有排列印出，數字越小的在越前面，每個數佔一列。 例如最前面幾列應為

```
123456
123465
123546
123564
123645
123654
124356
124365
…
```

解題思惟

1. 1 到 6 的所有排列中，最小的為 123456，最大的為 654321。
2. 如果跑一個迴圈，把 123456 到 654321 其中的數全部檢查一次，如果檢查出來該數是 1~6 的一個排列，就把它印出來。因為是從小到大檢查，所以也會從小到大依序印出來，這樣這一題就解開來了。
3. 但是怎麼知道一個 6 位數 n，是否為 1~6 的一個排列呢? 我們可以分解每一個位數，如果分出來的 6 個數，分別為 1~6 的話，那就是 1~6 的一個排列。怎麼知道 6 個數分別為 1~6 呢?
4. 不妨假設 a[0]… a[9]都等於 0，如果分解出來的一個位數是 k，就把 a[k]加 1。6 個數字都計算過之後，最後 a[0]…a[9]分別會代表 0~9 出現的數字次數。如果 a[1]…a[6]都等於 1，就代表這個數是 1~6 的一個排列。
5. 其實每次計算時，只要檢查 k，如果 k 落在 0 或者 7~9，則該數不是 1~6 的排列（為什麼? ）。如果 k 落在 1~6，先檢查 a[k]是否比 0 大，如果是的話，則不是 1~6 的排列（為什麼? ），那如果 a[k]是 0 的話，則把它加 1。如果都檢查完，沒有發現違反的狀況，則該數是 1~6 的一個排列（為什麼? ）。
6. 這樣只要寫一個函數 int isValid(int n)，檢查 n 是否為 1~6 的一個排列，這一題就很容易解答了。

程式碼

```c
#include <stdio.h>

int isValid(int n);

int main()
{
    int i;
    for (i=123456; i<=654321; i++) {
        if (isValid(i)) printf("%d\n", i);
    }
    return 0;
}
```

```c
int isValid(int n)
{
     int d, a[7]={0};
     while (n) {
          d = n%10;
          n /= 10;
          if (d==0 || d>6 || a[d]) return 0;
          a[d]++;
     }
     return 1;
}
```