討論使用漢明碼的 FP 問題

這篇文章原本用中文撰寫並發布在 CSDN 上。

問題連結

這個問題要求找到最小字典序的 `n` 個數字，使得其中任意兩個數字之間的漢明距離至少為 `d`。

漢明距離可以使用異或（XOR）來計算。`1^0=1, 0^1=1, 0^0=0, 1^1=0`。所以，對兩個數字進行異或操作會得到一個數字，其中的設置位對應於不同的位。接下來，我們可以計算結果中設置位的數量。

我曾經犯過一個錯誤，因為輸出要求每行 10 個數字，最後一行可能少於 10 個。我的初始輸出在最後一行最後一個數字後面有一個尾隨空格，然後是換行。

我認為這是一段很好的函數式編程風格代碼。好處是它更有結構，使 `main` 看起來像 Lisp 或其他函數式語言中的頂層。

這樣，我就不需要創建一個新的 cpp 文件來測試不熟悉的函數或調試單個函數了。我可以直接註釋掉 `deal()` 並將 `main` 作為頂層 REPL（讀取-打印-求值循環）來使用。

Lisp 也教會我盡可能地用函數式方式編程。這樣，每個函數都可以單獨提取和調試。語義也更加清楚。例如：

`hamming(0, 7, 2)` 表示檢查 0 和 7 的二進制表示是否至少在 2 位上不同。7 是 `111`，所以它們在 3 位上不同，函數返回 true。

所以，我可以註釋掉 `deal()` 並添加 `hamming(0, 7, 2)` 來獨立測試這個函數。

AC 代碼：

```
/*
{
ID: lzwjava1
PROG: hamming
LANG: C++
}
*/
#include<cstdio>
#include<cstring>
#include<math.h>
#include<stdlib.h>
#include<algorithm>
#include<ctime>
using namespace std;
const int maxn=1000;
```

```
bool hamming(int a,int b,int d)
{
    int c=a^b;
    int cnt=0;
    for(int i=0;i<=30;i++)
    {
        if((1<<i) & c)
        {
            cnt++;
            if(cnt>=d) return true;
        }
    }
    return false;
}

void printArr(int *A,int n)
{
    for(int i=0;i<n;i++)
    {
        printf("%d",A[i]);
        if((i+1)%10==0 || (i==n-1)) printf("\n");
        else printf(" ");
    }
}

bool atLesat(int *A,int cur,int i,int d)
{
    for(int j=0;j<cur;j++)
        if(!hamming(A[j],i,d))
            return false;
    return true;
}

void dfs(int *A,int cur,int n,int d)
{
    if(cur==n)
    {
        printArr(A,n);
        return;
    }
```

```
    int st=(cur==0? 0: A[cur-1]+1);
    for(int i=st;;i++)
    {
        if(atLesat(A,cur,i,d))
        {
            A[cur]=i;
            dfs(A,cur+1,n,d);
            return;
        }
    }
}

void deal()
{
    int n,b,d;
    scanf("%d%d%d",&n,&b,&d);
    int A[n];
    dfs(A,0,n,d);
}

int main()
{
  freopen("hamming.in","r",stdin);
  freopen("hamming.out","w",stdout);
    deal();
    //printf("%.2lf\n",(double)clock()/CLOCKS_PER_SEC);
  return 0;
}

/*
*/
```