B3870 [GESP202309 四级] 变长编码

> **核心结论：积木拼装法**
>
> 想象你在拼积木。每块积木只有 7 个格子可以存数。
> 如果后面还有没拼完的数，就在当前积木头顶插一面“小红旗”（最高位设为 1）。

题目大意

把一个大数 `N` 按每 7 位二进制一组进行拆分。

- 每组占一个字节。
- **除了最后一组**，其他组的最高位（第 8 位）都要补 `1`。
- 输出要把每个字节转成两位大写的十六进制。

思路分析

1. 拆分逻辑

我们可以用 `n % 128`（或者 `n & 127`）取出最低的 7 位。
取完后，把 `n` 缩小 128 倍（`n /= 128` 或 `n >>= 7`）。

2. 打标记（插红旗）

如果缩小后的 `n` 仍然大于 0，说明后面还有数，当前这组就要加上 `128`（即最高位变 1）。

3. 十六进制输出

小学生背 `cout << hex << uppercase...` 比较痛苦。这里推荐使用 `printf("%02X", byte)`：

- `%02`：不足两位补 0。
- `X`：大写十六进制。

C++ 参考代码

```cpp
#include <iostream>
#include <cstdio>
using namespace std;
```

```cpp
int main() {
    long long n;
    cin >> n;

    int bytes[20], cnt = 0;
    // 使用 do-while 保证 n=0 时也能输出一个 00
    do {
        int b = n % 128;      // 取出最后 7 位
        n /= 128;             // 砍掉最后 7 位
        if (n > 0) b += 128;  // 如果后面还有，最高位变 1
        bytes[cnt++] = b;
    } while (n > 0);

    for (int i = 0; i < cnt; i++) {
        if (i > 0) printf(" ");
        printf("%02X", bytes[i]); // 格式化输出：两位、大写、补0
    }
    printf("\n");

    return 0;
}
```

为什么这种写法更适合信奥选手

结论：**适合。**

因为它直接抓住了题目的本质：**每 7 位一组，也就是每次按 `128` 拆。**

- `n % 128`：取出当前这一组
- `n /= 128`：去掉当前这一组
- 如果后面还有组，就给这一组加上 `128`，表示最高位是 `1`

这样写有两个优点：

- 比“先转二进制字符串再分组”更短、更不容易错
- 更符合信奥里常见的“按进制拆位”思路

这段代码在教学时要提醒的细节

讲原代码时，提醒学生这三点就够了：

- `n = 0` 时也要输出一组，所以 `do...while` 更自然
- 不要混用 `cout` 和 `printf`
- 输出时别在最后多留一个空格

`printf("%02X", v)` 到底是什么意思

这是这题最重要的格式：

```
printf("%02X", v);
```

意思是：

- `%`：按格式输出
- `02`：不够 2 位就在前面补 `0`
- `X`：按大写十六进制输出

例如：

- `0` 输出 `00`
- `10` 输出 `0A`
- `255` 输出 `FF`

`printf` 常用格式小抄

信奥里最常用的记这几个就够了：

```
printf("%d", x);      // int
printf("%lld", n);    // long long
printf("%c", ch);     // 字符
printf("%s", s);      // 字符串
printf("%.2f", x);    // 保留两位小数
printf("%x", x);      // 小写十六进制
printf("%X", x);      // 大写十六进制
printf("%05d", x);    // 不够 5 位补 0
```

补充一点：

- `%s` 用在字符数组上
- 如果是 `string`，写 `str.c_str()`

这题里最该记住的几个格式

对这道题来说，最该记住的是：

```
%d
%lld
%02X
```

其中：

- `%d` 输出普通整数
- `%lld` 输出 `long long`
- `%02X` 输出两位大写十六进制，不够补 `0`

这三个在比赛里都很常见。