

# 1 L1585 画正方形 ☆

## 题目描述

输入一个正整数  $n$ ，要求输出一个  $n$  行  $n$  列的正方形图案（参考样例输入输出）。图案由大写字母组成。其中，第 1 行以大写字母 A 开头，第 2 行以大写字母 B 开头，以此类推；在每行中，第 2 列为第 1 列的下一个字母，第 3 列为第 2 列的下一个字母，以此类推；特别的，规定大写字母 Z 的下一个字母为大写字母 A。

## 输入输出格式

输入：输入一行，包含一个正整数  $n$ 。约定  $2 \leq n \leq 40$ 。

输出：输出符合要求的正方形图案。

输入示例	输出示例
3	ABC BCD CDE

## 样例解释

输入  $n = 3$ ，需要输出  $3 \times 3$  的矩阵。

行号	起始字母	该行内容
第 1 行	A (第0个字母)	A → B → C
第 2 行	B (第1个字母)	B → C → D
第 3 行	C (第2个字母)	C → D → E

## 算法分析

### 1. 问题分析

我们需要打印一个二维矩阵。对于矩阵中的每一个位置  $(i, j)$ ，最核心的难点在于如何处理“Z 的下一个字母是 A”这一规则。

- 我们可以将 A ~ Z 映射为数字 0 ~ 25。
- 这是一个典型的循环队列模型：当数字到达 25 (代表 Z) 时，再 +1 应该回到 0 (代表 A)。
- 取模运算 (%) 完美解决了这个问题。 $x = (x + 1) \% 26$ 。

### 2. 算法选择

- 解法 - 模拟：使用双重循环遍历行  $i$  和列  $j$ 。对于第  $i$  行，起始数值为  $i$ 。后续每往后一列，数值加 1。在输出时，通过模运算计算出对应的字母偏移量。

### 3. 实现思路

- 核心逻辑

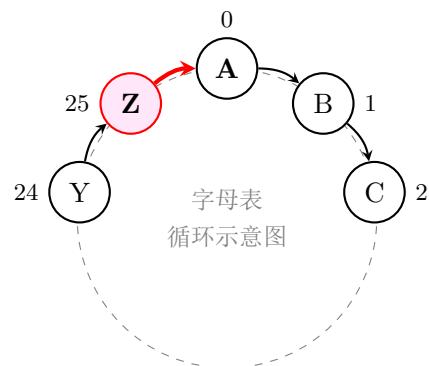
- (1) 循环结构：外层循环控制行（0 到  $n - 1$ ），内层循环控制列（0 到  $n - 1$ ）。

```
for (int i = 0; i < n; i++) { ... }
```

- (2) 字符计算与取模原理：假设当前我们应该输出第  $cur$  个字母（0 对应 A, 1 对应 B...）：

- 核心公式： $(cur \% 26)$

- 如下图所示，当  $cur$  增加到 26 时，取模结果自动变回 0 (A)。



最后加上字符 A 即可还原为大写字母。

```
int cur = i; // 当前行从第 i 个字母开始
for (int j = 0; j < n; j++) {
    // 先对26取余，确保数值限制在0-25之间，再转换为字母
    std::cout << (char)((cur++ % 26) + 'A');
}
std::cout << '\n'; // 每行结束后换行
```

## 拓展思考

- 如何只输出正方形的外框（即第一行、最后一行、第一列、最后一列）？
- 如果规则改为：Z → Y → ⋯ → B → A → Z（逆序）。该做哪些修改？