【深基1.习1】清扫教室

解题思路

这是一道典型的“工作效率”问题，可以通过**归一法**或**比例法**解决。
题目给出：

- **初始条件**：3 名同学 3 小时可以打扫 3 间教室。
- **目标条件**：9 名同学 9 小时可以打扫多少间教室？

方法 1：归一法（计算单人单小时的效率）

1. **计算单人单小时的打扫能力**：

- 3 名同学 3 小时打扫 3 间教室。
- 1 名同学 3 小时可以打扫 1 间教室（因为人数减少 3 倍，工作量也减少 3 倍）。
- 1 名同学 1 小时可以打扫 $\frac{1}{3}$ 间教室（因为时间减少 3 倍，工作量也减少 3 倍）。

2. **计算 9 名同学 9 小时的打扫能力**：

- 1 名同学 9 小时可以打扫 $9 \times \frac{1}{3} = 3$ 间教室。
- 9 名同学 9 小时可以打扫 $9 \times 3 = 27$ 间教室。

方法 2：比例法（直接比较人数和时间的变化）

1. **人数变化**：从 3 名增加到 9 名，人数变为原来的 3 倍。
2. **时间变化**：从 3 小时增加到 9 小时，时间变为原来的 3 倍。
3. **工作量变化**：人数和时间都变为原来的 3 倍，因此工作量变为 $3 \times 3 = 9$ 倍。
4. **最终答案**：初始可以打扫 3 间教室，现在可以打扫 $3 \times 9 = 27$ 间教室。

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

int main() {
    // 初始条件：3 名同学 3 小时打扫 3 间教室
    // 计算单人单小时的打扫能力
    double single = 3.0 / (3 * 3); // 1/3 间教室

    // 计算 9 名同学 9 小时的打扫能力
    int total_people = 9;
    int total_hours = 9;
    int classrooms = total_people * total_hours * single;

    // 输出结果
    cout << classrooms << endl;

    return 0;
}
```

【深基1.例10】简单的猴子吃桃

题目分析

这是一个典型的逆向思维数学问题，一只猴子每天吃桃子数量的规律为：

1. 每天吃掉当天桃子数量的一半加一个
2. 经过三天这样的吃法后，第四天只剩一个桃子
3. 需要求出最初有多少个桃子

解题思路

从最后一天倒推回去：

- 第四天剩余：1个
- 第三天剩余：(第四天剩余 + 1) × 2
- 第二天剩余：(第三天剩余 + 1) × 2
- 第一天剩余：(第二天剩余 + 1) × 2

解题步骤

1. 设第四天剩余桃子为1
2. 计算第三天吃之前的数量：(1 + 1) × 2 = 4
3. 计算第二天吃之前的数量：(4 + 1) × 2 = 10
4. 计算第一天吃之前的数量：(10 + 1) × 2 = 22
5. 初始桃子数量就是第一天吃之前的数量：22

代码实现

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

int main() {
    int day4 = 1; // 第四天剩余的桃子数量

    // 逆向计算三天前的桃子数量
    int day3 = (day4 + 1) * 2;
    int day2 = (day3 + 1) * 2;
    int day1 = (day2 + 1) * 2;

    cout << day1 << endl; // 输出初始桃子数量

    return 0;
}
```