

## 第04课

# 矿工计算器

把“够不够”和“值不值”变成可验证的计算



带七块煤和十根木棍，能合成多少火把？150个圆石能装几组？一趟采矿的收益怎样扣掉成本？这些问题不需要电脑猜答案，只需要把规则写成表达式。这一课，我们会区分算术、比较和逻辑，并给结果安排边界测试。

**阅读方式：先猜结果，再运行核对；每次只改一个条件。**

使用 Java 11 或更新版本。故事与插画借用方块世界，本系列练习运行在普通 Java 控制台。

## 01 先弄清问题，再挑运算符



进矿洞前算清资源，比在黑暗里猜自己还能撑多久更有把握。

算数量，通常需要算术运算；问“是否足够”，需要比较；多个要求同时出现，就要用逻辑把它们连接起来。三种运算得到的结果不同，不要把“有多少”和“是否足够”混成一件事。

- 多少根火把？得到一个整数。
- 利润是否至少50？得到 true 或 false。
- 利润达标并且挖到钻石？仍然得到 true 或 false。

本课计价单位与单价都是练习规则。数据先写在变量里，第6课再学键盘输入；别把“作品的初始数据”误当成已经会互动。

## 02 五把运算工具：+ - \* / %



拆开每一步计算，复杂结果就有了可追踪的来源。

**片段：放在 main 中，单独运行**

```
int coal = 7, sticks = 10, blocks = 150;  
System.out.println(coal + 3);  
System.out.println(sticks - coal);  
System.out.println(coal * 4);  
System.out.println(blocks / 64);  
System.out.println(blocks % 64);
```

**运行核对**

```
10  
3  
28  
2  
22
```

加法记录获得，减法记录消耗，乘法处理“每组几个”。对非负物品数量，用 / 数完整组，用 % 数剩余数量。这里暂按一种物品每组最多64个的练习规则计算。

"煤炭" + 7 得到 "煤炭7"，不会凭空多一个空格。7 + 3 才是整数加法。需要计算再输出时，把算术部分放进括号。

# 03 优先级：谁先被计算？

本课常用运算符，从高到低

| 层次   | 运算符       | 阅读提示       |
|------|-----------|------------|
| 括号   | ( )       | 先明确分组      |
| 取反   | !         | 翻转真假       |
| 乘除余  | * / %     | 同层通常从左到右   |
| 加减   | + -       | 也留意字符串拼接   |
| 大小比较 | > >= < <= | 得到 boolean |
| 相等比较 | == !=     | 得到 boolean |
| 逻辑与  | &&        | 两个都满足      |
| 逻辑或  |           | 至少一个满足     |

片段：放在 main 中，单独运行

```
System.out.println(2 + 3 * 4);
System.out.println((2 + 3) * 4);
int total = 3 * 12 + 5;
System.out.println(total);
```

运行核对

14  
20  
41

括号首先是给读者看的。当一个表达式让你读第二遍，就把意图用括号写清。表格只列本课常用运算，不是整部 Java 语言的完整优先级表。

## 04 完整组与余数，运输格与箱子

$$150 = 2 \times 64 + 22$$

第1组

64 个

第2组

64 个

剩余

22 个

片段：放在 main 中，单独运行

```
int blocks = 150;
int fullStacks = blocks / 64;
int leftover = blocks % 64;
System.out.println(fullStacks + " 组, 余 " + leftover);
int slotsNeeded = (int) Math.ceil(65 / 64.0);
System.out.println("65个物品需要的格数=" + slotsNeeded);
```

运行核对

2 组, 余 22  
65个物品需要的格数=2

第二个结果是物品栏格数，不是箱子数量。这里设定一种物品每格放64个；箱子会有多个格，算箱子数还要知道每个箱子的格数。先说清单位，计算才有意义。

至少要容纳全部物品，就要向上取整。先写 / 64.0 得到小数，再用 ceil；若先做整数除法，遗失的零头不会被 ceil 找回来。

# 05 比较：用一个问题得到真假

六种比较，逐项核对

| 运算符 | diamonds = 4 时 | 结果    |
|-----|----------------|-------|
| >   | 4 > 2          | true  |
| >=  | 4 >= 4         | true  |
| <   | 4 < 10         | true  |
| <=  | 4 <= 3         | false |
| ==  | 4 == 4         | true  |
| !=  | 4 != 0         | true  |

片段：放在 main 中，单独运行

```
int diamonds = 4;
System.out.println(diamonds > 2);
System.out.println(diamonds >= 4);
System.out.println(diamonds < 10);
System.out.println(diamonds <= 3);
System.out.println(diamonds == 4);
System.out.println(diamonds != 0);
```

运行核对

```
true
true
true
false
true
true
```

= 是赋值，== 是相等比较。Java 不能把 0 < y < 64 连写成范围判断，要写 y > 0 && y < 64。字符串内容比较会在第5课专门学习。

# 06 逻辑：把两个答案连起来



与、或、非是判断工具；插画中的红石门帮助我们记住它们的分工。

## 先看两个输入，再看结果

| A     | B     | A && B | A    B |
|-------|-------|--------|--------|
| true  | true  | true   | true   |
| true  | false | false  | true   |
| false | true  | false  | true   |
| false | false | false  | false  |

!true 是 false; !false 是 true。

## 片段：放在 main 中，单独运行

```
int torches = 24;
boolean hasPickaxe = true;
boolean ready = torches >= 20 && hasPickaxe;
boolean canStay = false || true;
System.out.println(ready);
System.out.println(canStay);
System.out.println(!ready);
```

## 运行核对

```
true
true
false
```

&& 要求左右都满足；|| 只需至少一边满足。它们还会短路：&& 左边为 false、|| 左边为 true 时，右边就不必求值。第5课会用它避免对 null 调用方法。

## 07 Math：给常见计算一个好名字

### 四件常用工具

| 方法         | 示例                    | 结果           |
|------------|-----------------------|--------------|
| Math.min   | Math.min(7, 10)       | 7            |
| Math.max   | Math.max(18, 23)      | 23           |
| Math.round | Math.round(12.6)      | 13 (long)    |
| Math.ceil  | Math.ceil(129 / 64.0) | 3.0 (double) |

### 片段：放在 main 中，单独运行

```
int batches = Math.min(7, 10);
int torches = batches * 4;
long distance = Math.round(12.6);
int slots = (int) Math.ceil(129 / 64.0);
System.out.println(torches);
System.out.println(distance);
System.out.println(slots);
```

### 运行核对

```
28
13
3
```

火把配方会被数量较少的材料限制，因此先取煤与木棍的较小值，再乘4。round 用于本例的最近整数；ceil 用于不能漏下零头的容量计算。

返回类型也要检查：本例 Math.round(double) 得到 long，Math.ceil 得到 double。最后的强转只用于已知范围内的格数，不能替代所有数值验证。

## 08 错误不是只靠改符号就能解决



给计算器做检查，需要分清语法错误、运行错误和规则错误。

### 三种情况，三条不同线索

- 把 `diamonds == 3` 写成 `diamonds = 3`：询问变成赋值，意图改变。
- 整数除以0：运行时会抛出 `ArithmeticException`。
- 把负数当成矿石数量：语法可能合法，却不符合本作品规则。

**片段：放在 main 中，单独运行**

```
int players = 0;
boolean canDivide = players != 0;
System.out.println(canDivide);
```

**运行核对**

```
false
```

`canDivide` 只保存检查结果，不会自动阻止后面的除法。必须让程序根据结果选择行动。下一页的完整作品借用一个 `if` 守住入口，第7课会详细讲它。

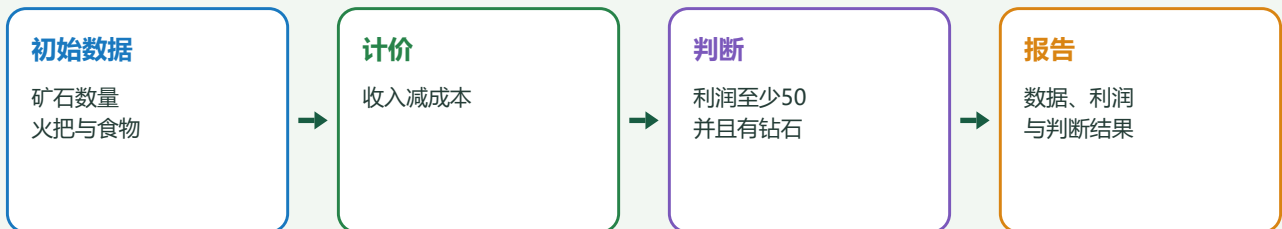
容量计算至少测0、63、64、65：需要的格数应为0、1、1、2。不要只测试自己已经熟悉的150。

## 09 先设计采矿收益的规则



同一份记录要同时记收入和消耗，不能只盯着挖到的钻石。

### 收益计算器的四个步骤



把煤、铁、金、红石、钻石的练习单价分别设为1、3、5、2、12。火把成本为每根1，食物为每份2。单价都用同一种“练习积分”，避免混用单位。

- 收入：每种矿石数量乘单价，再相加。
- 成本：火把数量加上食物数量的两倍。
- 利润：收入减成本；允许出现亏损。
- 无效数据：任何数量为负数，都先给出提示。

## 10 完整作品：先检查，再计算

### 完整程序 • MinerCalculator.java

```
public class MinerCalculator {
    public static void main(String[] args) {
        int coal = 20, iron = 8, gold = 3;
        int redstone = 12, diamonds = 2;
        int torchesUsed = 16, foodUsed = 4;
        boolean valid = coal >= 0 && iron >= 0 && gold >= 0
            && redstone >= 0 && diamonds >= 0
            && torchesUsed >= 0 && foodUsed >= 0;
        // 先借一个条件守门员；第7课会展开它。
        if (!valid) {
            System.out.println("数量必须是非负整数");
            return;
        }
        int income = coal + iron * 3 + gold * 5
            + redstone * 2 + diamonds * 12;
        int cost = torchesUsed + foodUsed * 2;
        int profit = income - cost;
        boolean richTrip = profit >= 50 && diamonds > 0;
        System.out.println("收入=" + income + ", 成本=" + cost);
        System.out.println("利润=" + profit);
        System.out.println("丰收=" + richTrip);
    }
}
```

### 运行核对

收入=107, 成本=24  
利润=83  
丰收=true

if 里的内容只在数据无效时执行；return 在这里结束 main。核心任务仍是读懂 valid、income、cost 与 richTrip 的表达式，并能手算同一份结果。

# 11 让边界值检查你的想法

## 第四课的能力地图



- 基础：389个圆石，完整组为6，余5；每格64个时至少需要7格。
- 进阶：火把至少20且有矿镐，并且食物或床至少有一种。写出带括号的判断。
- 创造：设计折扣公式，用 `Math.max(1, 计算出的价格)` 设置自己的最低价规则。

## 测试完整作品

把所有矿石设为0，保留消耗：利润应为-24，丰收为false。把 `coal` 设为-1：只输出数量提示。把成本调到使利润正好为50，再观察 `diamonds` 为0与大于0的区别。

自测：/ 与 % 能互换吗？`ceil` 算的是箱子数吗？答案都是否；单位和规则要先说明。下一课《聊天栏里的字符串魔法》学习组织、比较和清理文字。

技术参考：Java SE 11 的 Math API 与语言规范第15章表达式，官方文档见 [docs.oracle.com](https://docs.oracle.com)。