文件与 FRQ

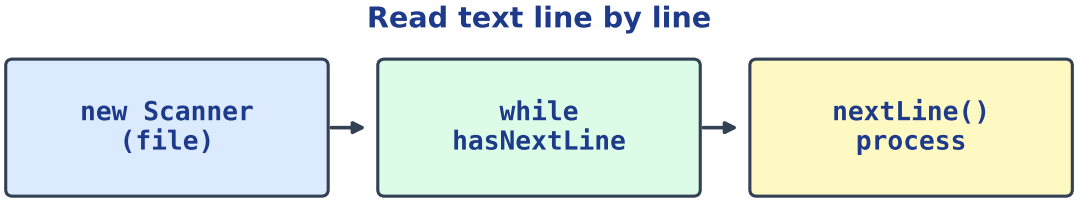
Java Reference

用 Scanner 读写文本文件

`Scanner` 一次读取一行文本。读取真实文件时写 `new Scanner(new File("scores.txt"))`; 这里我们包装一个 `String`, 这样示例在任何地方都能运行。用 `.split(" ")` 把一行拆分 (split) 成几部分, 用 `Integer.parseInt(...)` 从文本中解析 (parse) 出数字。

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        // 真实文件:  Scanner in = new Scanner(new File("scores.txt"));
        String data = "Alice 80\nBob 95\nCara 72";
        Scanner in = new Scanner(data);
        int total = 0, count = 0;
        while (in.hasNextLine()) {
            String line = in.nextLine();
            String[] parts = line.split(" ");      // 按空格拆分
            total += Integer.parseInt(parts[1]);
            count++;
        }
        System.out.println("average = " + (total / count));   // average
↪   = 82
    }
}
```



Exam tip: hasNextLine guards the loop; split + parseInt turn a line into data

Scanner reads lines while hasNextLine is true

AP FRQ 题型

AP CS A 考试有四道自由作答 (free-response) 题, 每道都有固定的形式:

- **第 1 题 — 方法与控制:** 按给定规范写方法; 循环、`if`、`String`/`Math`。
- **第 2 题 — 类的设计:** 根据描述写一个完整的类 (字段、构造方法、方法)。
- **第 3 题 — 数组 / ArrayList:** 处理一维数组或 `ArrayList`(查找、计数、构建新列表)。
- **第 4 题 — 二维数组:** 按行和列遍历一个网格。

技巧始终相同: 读懂规范, 严格按描述写出方法, 返回正确的类型。

```
public class Main {
    public static void main(String[] args) {
        // 第 1 题风格：按规范实现一个方法，然后接受测试。
        System.out.println(countEven(new int[]{4, 7, 10, 3, 6}));   // 3
    }

    /** 返回 arr 中有多少个偶数。 */
    public static int countEven(int[] arr) {
        int count = 0;
        for (int x : arr)
            if (x % 2 == 0) count++;
        return count;
    }
}
```

常见错误

- `nextInt()` 会把换行符留下, 导致紧接的 `nextLine()` 读到空行 — 先把它读掉。
- 读之前先检查 `hasNext()`, 避免读过文件末尾。
- 在 FRQ 里仔细看方法头: 返回类型和参数要完全对上。