**Ch3 12. 很多项目经理根据过去项目中程序员的生产率来计划项目的进度，生产率通常根据单位时间的单位规模来测量。例如，一个组织机构可能每天生产300行代码或每月生产1200个应用点。用这种方法测量生产率合适吗？根据下列事项讨论生产率的测度：**

**● 用不同的语言实现同样的设计，可能产生的代码行数不同。**

**● 在实现开始之前不能用基于代码行的生产率进行测量。**

**● 程序员可能为了达到生产率的目标而堆积代码。**

经过我们组的激烈讨论与分析，得出结论：用代码行数或应用点测量程序员的生产率的合理性存在明显缺陷，需结合以下因素综合讨论：

1. 编程语言差异导致代码行数不可比

不同编程语言实现相同功能所需代码行数差异显著。例如，Python的简洁性可能导致代码行数远少于Java或C++，但这并不意味着Python程序员效率更高。仅以代码行数衡量生产率会忽视语言特性带来的影响，导致跨项目或跨团队比较失真。

2. 代码行数无法在前期准确预估

在项目启动阶段，代码行数难以预测，尤其是采用新技术或需求频繁变更时。若依赖历史代码行数据制定计划，可能导致进度安排脱离实际，增加项目延期风险。

3. 代码堆积的负面激励

以代码行数为目标可能鼓励程序员编写冗余代码（如重复逻辑、无意义注释），甚至牺牲代码质量（如过度复杂的实现）。这会增加维护成本、降低可读性，最终损害项目长期效率。

改进评价意见如下：

① 结合多维指标：除代码行数外，可纳入功能完成度、缺陷率、测试覆盖率、用户价值交付速度等指标。

② 重视代码质量：通过代码评审、技术债务管理、自动化测试等确保产出有效性。

③ 使用更客观的度量标准：如“故事点”（基于功能复杂度）或“周期时间”（任务从开始到完成的耗时），减少对代码规模的依赖。

④ 动态调整计划：根据团队历史表现和项目实际进展灵活调整目标，避免僵化的指标约束。