**1.工作量估算：**

**ch3 习题12（小组讨论）**

**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。**

**单位时间内的单位规模作为生产力指标的不足之处**

以单位时间内的单位规模(例如代码行数或功能点数)来衡量软件开发的生产力存在着许多问题，看似直观，但实际上忽略了软件开发的复杂性和多样性，并可能导致误导性的结论。尽管如此，由于缺乏更有效的替代方案，它仍然被广泛使用。

以下是单位时间内的单位规模作为生产力指标的一些不足之处：

**1.受多种因素影响:**

（1）编程语言和应用领域:不同的编程语言和应用领域，其代码复杂性和开发难度差异很大，实现相同的功能，用简洁的语言可能只需要几行代码，而用繁琐的语言则可能需要几十行代码，导致代码行数或功能点数无法进行公平的比较。

（2）问题难度和新颖性:对于新颖或复杂的问题，程序员需要花费更多时间进行研究和探索，这会降低单位时间内的产出，但并不意味着生产力低下。

（3）代码复用程度:高效的代码复用可以减少代码量，但并不意味着生产力下降，反而体现了程序员的经验和能力。

**2.难以准确测量:**

（1）规模估算的滞后性:在项目初期，往往难以准确估算最终的代码规模或功能点数，导致无法及时制定合理的生产力目标。

（2）规模估算的主观性:功能点数的估算依赖于对软件功能的理解和分解，而这往往是一个主观的过程，容易受到个人经验和偏见的影响。 因此，基于功能点数的生产率测量也难以做到客观和准确。

**3.可能导致不良行为:**

（1）程序员追求数量而非质量:一味追求代码行数，可能会导致程序员编写冗长、低效、难以维护的代码，从而降低代码质量。优秀的代码应该简洁、高效、易于理解和维护，而代码行数无法反映这些特性。

（2）管理层制定不合理的期望:基于不准确的生产力指标制定目标，可能导致管理层对项目进度和成本产生误判，并给程序员带来不必要的压力。

**4.需要更深入的研究**

为了更好地理解和评估软件开发的生产力，我们需要进行更深入的研究，包括以下方面：

（1）开发更有效的生产力指标:探索能够更客观、更全面地反映软件开发效率和质量的指标，例如目标达成率、代码质量（通过代码审查评估代码的质量，包括可读性、可维护性、效率和安全性等方面）、修复bug的数量和速度、用户满意度（收集用户反馈，了解软件的易用性、功能性和稳定性，并以此评估开发团队的绩效）、团队合作（评估团队成员之间的协作效率，例如沟通能力、冲突解决能力和知识共享能力）等。

（2）分析影响生产力的因素:研究管理风格、团队结构、程序员技能水平、工作环境等因素对生产力的影响，并制定相应的改进策略。

（3）评估生产力目标的影响:调查生产力目标对程序员行为和项目结果的影响，并探索更有效的目标设定方法。

总而言之，代码行数和功能点数并不能准确地衡量软件开发的生产力。 我们需要采用更全面、更客观的方法，并结合项目实际情况进行综合评估，才能更有效地衡量和提升软件开发的效率和质量。