**在练习中，同学们们填写自己估计的时间， 以及记录实际花费的时间， 填写下面的表格：**

|  |  |  |
| --- | --- | --- |
| **PSP 各个阶段** | 自己预估的时间（分钟） | 实际的记录（分钟） |
| **计划:**明确需求和其他因素，估计以下的各个任务需要多少时间 | （填在计划阶段花费的时间） | （此项应该和预估值一致） |
| **开发**（包括下面 8 项子任务） | （以下都填预估值） | （以下都填实际花费值） |
| ·   需求分析 (包括学习新技术、新工具的时间) | 1h40mins | 5+0+30+30=1h5mins |
| ·  生成设计文档 （整体框架的设计，各模块的接口，用时序图，快速原型等方法） | 0 | 0 |
| ·  设计复审 (和同伴审核设计文档，或者自己复审) | 20mins | 5+5+5+5=20mins |
| ·   代码规范 (为目前的开发制定或选择合适的规范) | 0 | 0+1+2+0=3mins |
| ·  具体设计（用伪代码，流程图等方法来设计具体模块） | 0 | 0 |
| ·  具体编码 | 40mins | 3+5+10+2=20mins |
| ·  代码复审 | 10mins | 0+0=0 |
| ·  测试（自我测试，修改代码，提交修改） | **10mins** | **2+20+5+10=37mins** |
| **报告** |  |  |
| ·  测试报告（发现了多少bug，修复了多少） | 平均每个代码出现一个bug | 平均每个功能出现一个bug |
| ·  计算工作量 (多少行代码，多少次签入，多少测试用例，其他工作量) | 每个项目3个测试用例 | 每个项目3个测试用例 |
| ·  事后总结, 并提出改进计划 （包括写文档的时间） |  | 因为本次工程较为简单，设计文档可缩减为一句话就没写，实际完成中发现服用以前自己写的算法容易出点难找的小bug,debug过程没用测试用例不容易说明问题，在大型工程中需要设置专门各环节的测试用例。 |
| **总共花费的时间**（分钟） | 3H | 2H25MINS |

PSP  的特点:

•       不局限于某一种软件技术 (如编程语言),  而是着眼于软件开发的流程,  这样不同应用的工程师可以互相比较。

•       不依赖于考试, 而主要靠工程师自己收集数据, 然后统计提高。

•       在小型，初创的团队中,  高质量的项目需求很难找到，这意味着给程序员的输入质量不高，在这种情况下,  程序员的输出 (程序/软件) 往往质量不高, 然而这并不能全部由程序员负责。

•       PSP 依赖于数据

•       需要工程师输入数据, 记录工程师的各项活动, 这本身就需要不小的代价。

•       如果数据不准确或有遗失, 怎么办?  让工程师编造一些?

•       如果一些数据不利于工程师本人 (例如: 花很多时间修改缺陷), 怎么能保证工程师能如实地记录这些数据呢?

•       PSP的目的是记录工程师如何实现需求的效率,  而不是记录顾客对产品的满意度。工程师可能很高效地开发出一个顾客不喜欢的软件, 那这个工程师还是一个优秀的工程师么?