7.1.1

程序设计语言是人和计算机通信的最基本的工具，会影响人的思维和解题方式，影响人和计算机通信的方式和质量，影响其他人阅读和理解程序的难易程度。

选择适宜的程序设计语言的原因：（可以做成问题）

根据设计去完成编码时，困难最少；

可以减少需要的程序测试量；

可以得到更容易阅读和更容易维护的程序

相比于汇编语言高级语言更加的实用高效，除了在特殊领域如对程序执行时间和使用时间和使用空间都有很严格限制的情况，需要产生任意的甚至非法的指令序列等：

区别：

汇编语言编码需要把软件设计翻译成机器操作的序列，既困难又容易出差错；

高级语言写程序比用汇编语言写程序生产率可以提高好几倍；

用高级语言写的程序容易阅读、容易测试、容易调试、容易维护。

选择程序语言有理想标准和实用标准

理想标准：

应该有理想的模块化机制，以及可读性好的控制结构和数据结构；使程序容易测试和维护以减少软件的总成本

使编译程序能够尽可能多地发现程序中的错误；便于调试和提高软件的可靠性

应该有良好的独立编译机制。降低软件开发和维护的成本

实用标准：

系统用户的要求；

可以使用的编译程序；

可以得到的软件工具；

工程规模；

程序员的知识；

软件可移植性要求；

软件的应用领域。

7.1.2编码风格

源程序代码的逻辑简明清晰、易读易懂是好程序的一个重要标准，为了做到这一点，应该遵循下述规则。

第一点是程序内部的文档

所谓程序内部的文档包括恰当的标识符、适当的注解和程序的视觉组织等

第二点是数据说明

数据说明的次序应该标准化；

当多个变量名在一个语句中说明时，应该按字母顺序排列这些变量；

如果设计时使用了一个复杂的数据结构，则应该用注解说明用程序设计语言实现这个数据结构的方法和特点。

第三点是语句构造

下述语句构造的原则有助于使语句简单明了：

不要为了节省空间而把多个语句写在同一行；

尽量避免复杂的条件测试；

尽量减少对“非”条件的测试；

避免大量使用循环嵌套和条件嵌套；

利用括号使逻辑表达式或算术表达式的运算次序清晰直观。

第四点是输入输出

在设计和编写程序时需考虑有关输入输出风格的规则：

对所有输入数据都进行检验；

检查输入项重要组合的合法性；

保持输入格式简单；

使用数据结束标记，不要要求用户指定数据的数目；

明确提示交互式输入的请求，详细说明可用的选择或边界数值；

程序设计语言对格式有严格要求时，应保持输入格式一致；

设计良好的输出报表；

给所有输出数据加标志。

第五点是效率

效率主要指处理机时间和存储器容量两个方面。

效率是性能要求，因此应该在需求分析阶段确定效率方面的要求；

效率是靠好设计来提高的；

程序的效率和程序的简单程度是一致的，不要牺牲程序的清晰性和可读性来不必要地提高效率。

效率又从三个方面进行分析，

1）程序运行时间

写程序之前先简化算术的和逻辑的表达式；

仔细研究嵌套的循环，以确定是否有语句可以从内层往外移；

尽量避免使用多维数组；

尽量避免使用指针和复杂的表；

使用执行时间短的算术运算；

不要混合使用不同的数据类型；

尽量使用整数运算和布尔表达式。

1. 存储器效率

在大型计算机中必须考虑操作系统页式调度的特点，一般说来，使用能保持功能域的结构化控制结构，是提高效率的好方法。

在微处理机中如果要求使用最少的存储单元，则应选用有紧缩存储器特性的编译程序，在非常必要时可以使用汇编语言

提高执行效率的技术通常也能提高存储器效率。

1. 输入输出的效率

在写程序的角度有以下原则

所有输入输出都应该有缓冲，以减少用于通信的额外开销；

对二级存储器(如磁盘)应选用最简单的访问方法；

二级存储器的输入输出应该以信息组为单位进行；

如果“超高效的”输入输出很难被人理解，则不应采用这种方法。