“推演渐进式”编程与教学——以“汇编语言程序设计”为例

孔祥涛
成都锦城学院，中国·四川 成都 611731

摘　要：教科书将编程步骤化，在编程前先进行绘制流程图，这一过程易引入其他概念，分散学生精力，绘制流程图对于初学编程者在没有思路的情况下进行绘制，增加了学习难度。从具体问题出发，剥离出要做的核心事情，然后将其符号化，之后逐步推演，直至设计出符合程序语言规范的程序，这一编程过程和教学法称为“推演渐进式”教学法。将其与“教科书的编程步骤法”进行了教学效果对比，试验证明“推演渐进式”明显提高了初学者的汇编语言程序设计能力。最终提出，还原编程者的编程过程，而非按编程者的结果进行讲解授课。

关键词：编程步骤；程序流程图；推演；渐进；归纳总览式

“Progressive Programming and Teaching” — Taking “Assembly Language Programming” as an Example

Xiangtao Kong
Chengdu Jincheng College, Chengdu, Sichuan, 611731, China

Abstract: Textbooks simplify programming steps by drawing flowcharts before programming, which can introduce other concepts and distract students’ energy. Drawing flowcharts increases the difficulty of learning for novice programmers who may not have a clear idea. Starting from specific problems, extracting the core tasks to be done, then symbolizing them, and gradually deducing until a program that meets programming language standards is designed, this programming process and teaching method is called the “deductive progressive” teaching method. The teaching effect was compared with the “step-by-step method of programming in textbooks”, and the experiment proved that “deduction and progression” significantly improved the assembly language programming ability of beginners. Ultimately, it is proposed to restore the programming process of the programmer, rather than teaching based on the programmer’s results.

Keywords: programming steps; program flowchart; deduction; gradual; inductive overview formula

0 前言

汇编语言程序设计是《微机原理及接口技术》第 4 章的教学内容 [1-3]，汇编语言程序以难以理解和学习著称，学生们在学本章史时常感到异常沮丧，书看不懂，课又听不懂，如坐针毡。这种情况极其容易导致学生对编程失去信心。究其难点主要在于：①无从下手，不知从哪里开始编起；②如何运用寄存器和存储器，以及指令对寄存器和存储器的影响；③如何构造结构化程序。这里第 1 点是没有思路，第 2 点是指令基础不过关，第 3 点是不会运用程序控制类指令。

论文以汇编语言程序设计中的循环结构程序为例，分析了传统教学方法的原理及特点，并指出这类方法在初学者学习过程中是如何造成学生看不懂书，学不会汇编语言程序设计的原因，然后遵循初学者的认知规律，利用推演 [4-10]，逐步演变的方式完成汇编语言程序的设计。

1 传统教学方法及其缺点分析

传统的编程教学方式，以理论课的方式呈现，先讲编程步骤，然后再讲汇编语言的程序结构，最后按程序结构进行例题讲解。讲解完之后，就是布置习题进行练习。

这种教学内容编排以前人归纳总结的结论与规律出发，直接告诉编程的步骤，将程序设计步骤化。我们把这种编程与教学方式称之为“归纳总览式”。

“归纳总览式”教学法的特点是站在前人归纳总结的基础上，将事物归纳总结出一般性规律，并将其推广至处理类似事物中去。这一方法意在指引学习者按照此步骤或方法就可以解决问题。其特点在于规律普遍化，放之四海皆正确。

1.1 引入其他概念导致初学者抓不住主干

这种教学方式看似非常正确，采用的是权威性的编程经验，却是导致初学者无法掌握编程的根源，他切断了学生体会推演的进程，以权威的形式告诉学生应该怎么做，让学生按此步骤执行，但学生在按此步骤执行时，遇到了更多的问题困扰。前人总结出的编程经验是建立在多次编程之后，处理过多种情况之后，通过归纳法整理出来的。这种总结出的编程经验可以将复杂的事情步骤化、刻板化、条理化，便于执行，其实是便于共识者执行，但不适用于初学者学习。

究其原因，是编程经验总结是要全面，要能够面对各种复杂的情况，在这种情形下，不免会引入更多的知识面，引进更多的专业术语，而这种引入，导致了学生在学习编程过程中，精力被其他的知识点分散，引向了细枝末节，进而迷失方向，陷入一筹莫展的境地。如下，教科书上会明确指出汇编语言程序设计的基本步骤为：①分析问题，建立数学模型；②确定算法；③编制程序流程图；④编写汇编语言源程序；⑤上机调试 [1-3]。

但对于初学者来讲，这第一步就造成了非常大的困难，“建立数学模型”，是什么数学模型，什么是算法，“算法”也是一个非常复杂的事情，还没编过程序，哪里来的算法认知，脑子里没有存储过任何算法。此刻，很多同学无法进行编程会认为不会数学模型，心中没有算法。转而去攻克数学模型、算法，这样会导致忘了当时的主要任务是编程。因为不会编，认为是不会数学模型、算法导致，在学习数学模型和算法时，又引入了其他概念或者专业术语，这样从最开始的学习编程，到最终方向迷失，无法拔出泥潭。

还有一种同学，现在是要学习编程，而又提到先要掌握数学模型和算法，直接就望而却步。

这样两种情形，都会增加学生掌握编程的难度。前一种是迷失方向，后一种是畏难情绪。

在实际教学过程中，笔者也曾尝试用过教科书的步骤进行汇编语言程序讲解，很多同学在听完此步骤之后，直接陷入沉思，会提问“什么是数学模型”“什么是算法”，注意力直接被带入其他方向。

1.2 先绘制流程图再编程的顺序，导致初学者入手困难

而这第三步，要求初学者绘制流程图。绘制流程图，是要从开始思考至结尾，需要对全局进行把控好之后，才能正确绘制，而初学者在没有思路的情况下，被要求绘制程序流程图，是非常困难的，其很容易在这一步就思路卡壳，无法进入下一步，也就无法开始编程，切断了编程体验进程。

2 “推演渐进式”教学方式

根据初学者的认知习惯，在思路混沌的情况下，从题目出发，分析题目要做的事情，先用自己的符号表达想要做的事情，之后据此进行形式演变，将其演变为专业上的程式（指令），之后再进一步完善条件，最终程序成型，最后结合仿真软件进行结果验证。我们将这种教学方式称之为“推演渐进式”。这种学习方式最大的特点是具体问题具体分析，摸着石头过河，在行进中逐步完善。

3 教学实例分析

以下以教科书中典型的循环结构程序为例，进行汇编语言程序编写的教学演示。

例题 1：有两个 4 字节的无符号数相加，这两个数分别存放在 2000H 和 3000H 开始的存储单元中，得到的和也为 4 字节，存放在 2000H 开始的存储单元中，编一段程序完成这两个数的相加过程 [3-9]。

3.1 “归纳总览式”教学方法的缺点

教科书中把循环程序结构，总结为 4 部分：初始化部分、循环体、修改部分、控制部分，这就引入了对初始化、循环体等新专业术语，在介绍完这些专业术语或概念之后，反而增加了学生的学习负担，目前是解决这道题，还是要弄清这 4 部分的概念，而且学生很难从文字叙述中去理解刚才涉及概念。

教科书在编程的第三步，要求绘制程序流程图，在流程图中标注出初始化、循环体等等。但是学生不明白为什么要这些初始化、循环体呢？此题中的初始化、循环体如何安排？传统的程序流程图见图 1 所示。

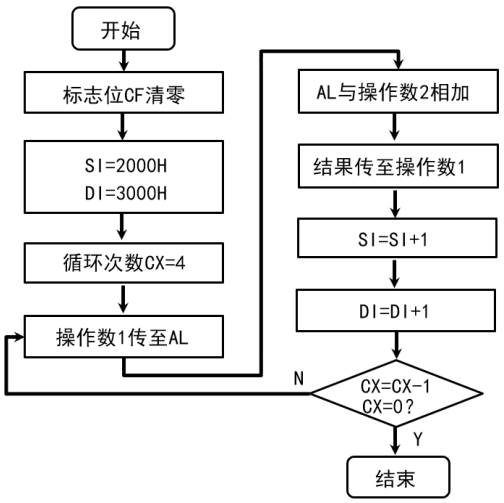

图 1 程序流程图 [1-3]

开始后，第一步是标志位清零，为什么要清零，清零和题目的内容有什么关系。

第二步让 SI=2000H，DI=3000H，为什么？是准备计算吗？

第 3 步让 CX=4，为什么？别的程序是不是也给 CX=4？CX 接下来是要参与什么运算？

第 4 步至第 9 步为什么要在这里循环？为什么不从第 3 步开始循环？

SI 加 1 后，有什么作用，寄存器数值加 1 和循环为什么能联系到一起？

程序流程图是思路归纳整理出来的结果，甚至是程序编制完成后，再完成的。而在编程之初就进行绘制，是很有难度的。

3.2 “推演渐近式”编程过程

笔者从这道题出发，按题意逐步推理、演变，直至最终完成编程。

先从题目中，分析并概括出要做的事：先绘制出题目中的数在存储器的形式，如图 2 所示。

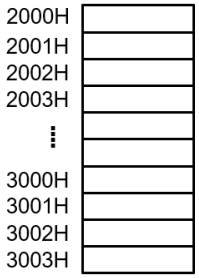

图 2 数在存储器中的分布

再把题目中要做的事情画一个简图，如图 3 所示，题目中要求 4 字节进行加法运算。

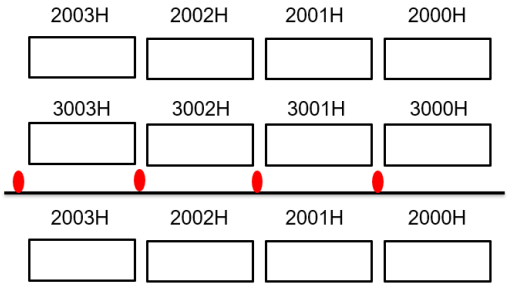

图 3 4 个字节的无符号数相加示意图

从这张图就可以得知要做的事是让对应的存储单元的内容进行加法运算。

然后用微机原理的指令写出要做的事：

ADD　[2000H]，[3000H]

的确，第一步是要完成（2000H）的数据和（3000H）的数据相加，但是微机原理的指令语法不允许两个操作数都为存储器，所以要对上面的程序进行变形，按指令规则变为如下：

MOV　AL，[2000H]

ADD　AL，[3000H]

MOV　[2000H]，AL

即先将（2000H）的数据传给寄存器 AL，再将 AL 中的数据与（3000H）的数据进行 ADD 运算，再将计算后的数据从 AL 传给存储单元（2000H），上面三行指令是按照题意用规范的指令表达出第一步想做的事情，接下来需要对图 3 中的（2001H）的数据和（3001H）的数据进行相加计算，可以写成：

MOV　AL，[2001H]

ADC　AL，[3001H]

MOV　[2001H]，AL

让学生仔细观察此程序与前段的程序相似性，是一种重复性的工作，都是做加法运算，这种重复性工作就可以用循环结构来表达，故需要对上述程序进行变形，以能够运用循环控制指令。变形结果如下：

MOV　AL，[SI]

ADC　AL，[DI]

MOV　[SI]，AL

用 [SI] 代替 [2000H]，用 [DI] 代替 [3000H]，形成寄存器间接寻址。然后让同学思考这几行程序执行完之后，如果要操作下一个存储单元如何做？需要对 SI 和 DI 进行加 1 操作。

在上述程序下面添加：

INC SI

INC DI

形成存储单元的地址自增操作，之后就需要回到“MOV AL，[SI]”这条指令处重新执行，那就需要用到循环指令 LOOP，写成如下：

BJ1：MOV　AL，[SI]

ADC　AL，[DI]

MOV　[SI]，AL

INC SI

INC DI

LOOP BJ1

这样就完成了此题最核心的事情，在此基础上再进行完善，从哪个存储单元开始，循环结构需要循环多少次，第一次相加的时候如果 CF 标志位为 1 的话怎么办？最终完成的程序如图 4 所示，在图 4 的右边列出了程序设计人员的实际编程步骤顺序。

```
        CLC              10
        MOV  CX, 4       9
        MOV  SI, 2000H   7
        MOV  DI, 3000H   8
BJ1:    MOV  AL, [SI]    1
        ADC  AL, [DI]    2
        MOV  [SI], AL    3
        INC  SI          4
        INC  DI          5
        LOOP BJ1         6
        HLT              11
```

图 4 最终的程序

从图 4 中可以看出编写程序不是从第一行开始编写，不是顺次完成的，是先分析出核心的事情，然后再进行条件完善。这种符合人类的认知和处理事情的思维进程，也是我们将其称之为“推演渐进式”编程法的原因。

而从课本这类平面教材中学习时，学生看到的第一句程序是 CLC，而在实际编程中这是编程人员写出的第 10 句程序，平面教材呈现的是编程结果，而非编程过程。平面教材会导致学生从第一句程序就陷入了思考漩涡，不仅如此，而且这种思考的方向会把学习者的思路引向细枝末节，找不到主干。

4 教学结果测试及分析

在笔者所授课的班级中，曾进行过以下试验，对于前

期基础相似和能力相似的两个班（班级 1 和班级 2），班级 1，先采用“归纳总览式”讲解例题 1，然后对学生进行编程测试 A，统计成功的人数，然后再按“推演渐进式”重新讲解一遍例题 1，再进行编程测试 B，统计成功的人数。结果如表 1 所示。班级 2，先采用“推演渐进式”讲解例题 1，然后对学生进行编程测试 A，统计成功的人数，然后再按“归纳总览式”重新讲解一遍例题 1，再进行编程测试 B，统计成功的人数。结果如表 2 所示。

其中编程测试 A 与测试 B 的内容是同一类型的循环结构程序习题。

表 1 班级 1 的测试结果

班级 1（总人数 30）	测试 A	测试 B
通过者人数	3 人	16 人
通过者人数占比	10%	53.3%

表 2 班级 2 的测试结果

班级 2（总人数 32）	测试 A	测试 B
通过者人数	17	21
通过者人数占比	53.1%	65.6%

通过以上试验得出，如果对两个班只采用其中一种教学方式，推演渐进式的教学效果比“归纳总览式”的效果高 431%。而两种教学方法依次采用后的，先“推演渐进式”后“归纳总览式”的教学效果比先“归纳总览式”后“推演渐进式”的效果高 12.6%。这说明即使两种方法都采用，先进行“推演渐进式”，后进行“归纳总览式”的教学安排还是要比先进行“归纳总览式”，后进行“推演渐进式”的教学效果好。

关键点在于先“归纳总览式”再“推演渐进式”，掌握人数占比从 10% 上升至 53.3%，掌握人数大幅上升，上升了 43.3%。而先“推演渐进式”再“归纳总览式”，掌握人数占比从 53.1% 上升至 65.6%，掌握人数上升了 12.5%。上升情况不如前一种教学安排。这表明了对编程成功起作用的学习方式是“推演渐进式”。

综上测试表明，对于初学编程者，“推演渐进式”教学法明显优于“归纳总览式”教学法。

在做一件比较困难的事情之前，应该把事情简单化，抓住主干，用词用语要尽量简洁，不要引入其他的更为复杂的概念，以防思路被引入其他方向。

故论文提出在教学过程中，要进行思维推演式教学，让学生进行推演式学习，这种教学方式的核心思想是“从实际事情或实例出发，分析题目要做的事情，先用自己的符号表达想要做的事情，之后据此进行形式演变，将其演变为专业上的程式（指令），之后再进一步完善条件，最终成型”。还原编程者的编程过程，而非按编程者的结果进行讲解授课。

当前，学生喜欢在 B 站等网络上观看同龄人制作的视频教程，最主要的原因是，学生能看到其他学生的制作过程，也就是进行“过程还原”，这种学习方法易于被大多数同学接受。而这种“过程还原”恰好是“推演渐进式”教学的其中一个环节。

5 结语

教科书上总结出的解题步骤适合已经熟练的编程人员，而对于刚刚接触某门课程的学生来讲，其总揽全局的能力有限，甚至常因为某些解题步骤引入了新的概念而陷入思维混乱，增加了学习过程中的困难。采用“推演渐进式”的教学方式，从题目逐步进行推演、修改、完善，将“思考过程”和“思维修正过程”呈现出来，进而引导学生掌握知识，并灵活运用。学生获取的能力是在摸索中前进的能力，而非照章办事。要求学生照章办事利于将事情规范化，但是容易导致初学者不知道其中的原因是什么。编程类课程学习，需要解决一个个具体问题，而且很短时间内就可以验证是否掌握和灵活运用，照章办事在理工科类中题目中极其容易出现，题目稍微一变，而学生不会了。当“推演渐进式”成为习惯之后，学生可以具体问题具体分析，获得的是分析解决问题的能力。

另外，“推演渐进式”教学方式不仅适用于程序类课程学习，也适合其他科技类课程，在历史类课程中也可采用“推演渐进式”教学。

参考文献：

[1] 马春燕.微机原理与接口技术(第3版)[M].北京:电子工业出版社,2018.

[2] 方红.微机原理及接口技术(第2版)[M].北京:电子工业出版社,2022.

[3] 马义德.微型计算机原理及应用(第三版)[M].北京:高等教育出版社,2004.

[4] 曾德妙,甘泉,李伟,等.初探病例复盘结合推演法在骨科临床思维建立中的应用[J].医学理论与实践,2023,36(17):3028-3050.

[5] 于彤,孙树峰,王素.多维案例推演教学模式在公安专业课程中的设计与实践——以“网络犯罪侦查”课程为例[J].广州市公安管理干部学院学报,2023,33(1):38-44.

[6] 魏晓熙.电子音乐的人机界面与创作思维研究——基于多学科推演[D].上海:上海音乐学院博士论文,2022.

[7] 曾德妙,刘炜,赵丽娟,等.病例推演法在骨科临床思维建立中应用的初步探讨[J].成都中医药大学学报(教育科学版),2022,24(2):20-23.

[8] 赵瑞莲.运用“复盘与推演”学活历史,让历史鲜活[J].中学历史教学,2019(12):47-49.

[9] 张艳.高中历史事件教学深度推演策略[J].广西教育,2021(34):132-133.

[10] 柳春艳,杨克虎.西方循证教育学推演:理论、方法及启示[J].电化教育研究,2022,43(3):25-31.

作者简介：孔祥涛（1982-），男，硕士，高级工程师，从事微机原理、单片机开发、工业自动化等方面的教学与科研。