プログラミング教育におけるモデルプログラム

梶浦　文夫
倉敷芸術科学大学産業科学技術学部

（2005年９月30日　受理）

１．はじめに

著者は情報系学科の学生を対象にしたプログラミング入門教育に長年携わり，さまざまな教育実践を行ってきた[1) 2) 3) 4) 5)]。プログラミング入門教育において，学生はまず基本的な知識や約束事を学んだ後，簡単な例題プログラムを実際に作成する作業を通じて，プログラミングへの理解を深めていく。この場合，例題プログラムの良し悪しはプログラミングの理解に対して決定的な影響を与える。そこで，プログラミング教育に携わる者は良質の例題プログラムをできるだけ多く学生に提示することが望ましい。

以上のような観点から，著者もさまざまな種類の多数の例題プログラムを学生に示し，説明し，作成させ，ヒントを与えながら改良させ，理解を深められるよう工夫してきた。しかし，多数の例題プログラムを学習することは，初心者の学生にとって消化不良を起こさせる危険性も伴っている。そこで，著者はこれまでどおり多数の例題プログラムを学習させると同時に，それらの中で特に重要なプログラムを「モデルプログラム」として選び出し，集中的に学習・理解させるという試みを行った。2004年度には１本，2005年度には３本のモデルプログラムを決めて集中的に学習させた。その結果，期末試験の成績が予想以上に上がるとともに，学生たち自身が「特定のプログラムを繰り返して学ぶ」ことを重要だと考えていることがわかった。

以下では，2004年度，2005年度の授業実践およびその学習効果について報告する。

２．実践授業の概要

２．１　対象科目

一昨年度（2003年度）から今年度（2005年度）まで授業実践を行っている対象の科目は，情報系学科１年次生を対象としたCプログラミング入門のための「プログラミングの基礎Ⅰ」および「同演習」である。これらの科目の開講時期は１年次前期である。受講者数は再履修者を含めて80名から100名と多人数であるため，３～４クラスに分けて授業を行っている。クラス分けは能力別ではなく，学生番号で機械的に行っている。著者はそれらのうちの１クラスを担当している。

この２つの科目で学習する内容を表１に示す。表１に含まれない「関数」「ファイル」「ポインタ」「構造体」などは，後期の「プログラミングの基礎Ⅱ」および「同演習」で学習する。

2．2 授業の進め方

以下では2003年度，2004年度および2005年度に共通した授業の進め方を述べる。授業には黒板，スクリーン，プロジェクタ，電源+LANコネクタ付普通教室を使用した。対象の学生は入学時に学科指定のNotebookパソコン（WindowsXP, Linux導入済み）を購入しているので講義と実習を行うことができる。また，Notebookパソコンを机のLANコネクタに接続すればインターネットにアクセスできる。講義に関する連絡事項，さまざまな情報，参考資料はWeb上の講義のページに掲載しておくようにしている。

表1 対象科目の学習内容

- プログラムの入力からコンパイル／実行まで
- 初めてのCプログラム・プログラムの書き方
- 簡単な出力・関数を使う・書式付出力 ・演算
- データ型 ・変数と定数 ・書式付入力
- 選択文(if, switch など)
- 繰り返し(while, for など)

Cプログラミングの実習環境は，OSがLinux，コンパイラがgcc，エディタがxemacsである。情報系の学科の学生であるため，UNIXに慣れさせる目的で，WindowsでなくLinux上でプログラミングをさせている。1年次前期開講の授業なので，Linuxに慣れている学生はほとんどおらず，Linuxの操作にも若干の時間がとられている。

表2 前期の授業の概要

教室	電源，LAN コネクタ，スクリーン，プロジェクタ，黒板
プログラミング環境	Notebook パソコン，Linux，gcc，xemacs，講義 Web ページ
例題プログラム	2003,2004 年度 52 本 2005 年度 48 本 (少しずつ異なるプログラムが多数ある)
小テスト (最後の 5 分)	2003 年度 12 回 2004 年度 16 回 2005 年度 19 回

授業には，黒板とプロジェクタ（+スクリーン）の両方を使っている。主に操作や実行結果の確認にはプロジェクタを，新しいことや考えさせる内容の説明には黒板を用いている。プログラミング学習の初期には与えられたソースプログラムを入力して実行させることも多いが，できるだけ考えさせて，例題プログラムを自分なりに改良させるように促している。2003年度および2004年度前期中に作成した例題プログラムはどちらも52本，2005年度は48本だった（実際には類似のプログラムが多数含まれている）。

最初のうちは全員が同じようなプログラムを作成する場合が多いので，レポートを少なくし，その代わりに小テストを頻繁に行った（2004年度は全講義回数27回のうち16回，2005年度は19回）。小テストの答案は，実施した日に採点し，間違い箇所を訂正し，説明を加え，次回の講義時には返却するようにしている。小テストの内容は，その日に学んだことの復習のための問題やその時点で集中的に学んでいるモデルプログラムの書き取りなどである。

表3 繰り返し処理のモデルプログラム

```
for 文で○～○までの○の倍数の和を求める
/* Sample program   Date   Student's name */
#include <stdio.h>
main()
{
    int   i, sum = 0;
    for(i = ○;I < ○;i += ○)
          sum += i;
           printf("○～○までの○の倍数の和=%d", sum);
}
```

３．2004，2005年度の取り組み

３．１　2004年度の取り組み

2004年度は以下のような取り組みを実践した。すなわち，「プログラムを考えるときのベースとなるモデルプログラムをしっかりと身につけさせる」という取り組みである。2004年度は，繰り返し処理について，このようなプログラムのモデルを１つ決定し，授業でも強調した上，小テストでも５回取り上げることによって集中的に理解度を高めるようにした。

表４　if文による条件判定のプログラム

```
入力した整数が○の倍数かif文で判定する
/* Sample program   Date   Student's name */
#include <stdio.h>
main()
{
    int   n;
    printf("n が○の倍数かどうか調べます¥n");
    printf("n を入力してください¥n");
    printf("n = ");
    scanf("%d", &n);
    if(n % ○ == 0)
        printf("○の倍数です¥n");
    else
        printf("○の倍数ではありません¥n");
}
```

表３に繰り返し処理のモデルプログラムを示す。このプログラムの内容は，「for文を用いて○から○までの○の倍数の和を求める」ものである。内容として含まれているのは，（１）制御変数を使った繰り返し，（２）繰り返し処理の中での制御変数の利用，（３）合計処理の基本などである。また，コメントやプリプロセッサディレクティブまで含んだ完全なプログラム（このままコンパイルできる）であることが重要である。

まず，その重要性を指摘した上で，表３の形のプログラムを授業中に説明し，例題プログラムとして作成させた。その授業のときから５回連続で表３の内容の小テストを行った。この５回の小テストでは，何も見ずにソースプログラムを解答用紙に全て書かせた。提出された小テストの答案は採点し，間違いがあれば正解や注意点を書いて全員に次回の講義の折に返却している。さらに，間違いの多い箇所について授業中に全員に注意して次の小テストを受けさせるようにした。

図１に，繰り返しのモデルプログラムについての小テストの平均得点の推移を示す。５回の小テストは全て５点満点で採点している。２回目の小テストで多くの学生が５点満点をとるようになったが，そこから５回目までは徐々に理解度が増加していっていることが分かる。

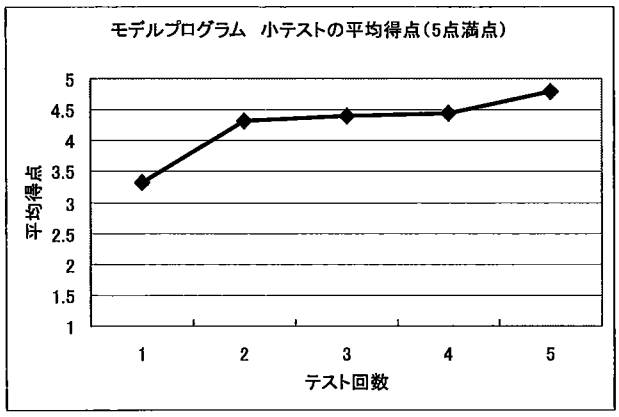

図１　モデルプログラムの平均点の推移

３．２ 2005年度の取り組み

2005年度は，モデルプログラムを３つに増やした。それらは，

（1）プログラムの基本形理解のための「Hello,world」を出力するモデルプログラム

（2）if～else文を使い，入力した整数が○の倍数であるかどうかを判定するモデルプログラム

（3）for文を使って○から○までの○の倍数の和を求めるモデルプログラム（04年度と全く同じ）

の３本である。（2）のプログラムを表４に示す。また，（3）のプログラムは表３に示した2004年度と全く同じものである。

2004年度と違って，講義の初期からその時その時の学習内容に合わせて３つのモデルプログラムを集中的に学習し，理解を深めるようにした。

図２から図４に（1）から（3）のモデルプログラムを書かせる小テストの平均点の推移のグラフを示す。2005年度の小テストは全て3点満点である。これら３つのグラフを見ると，最初の

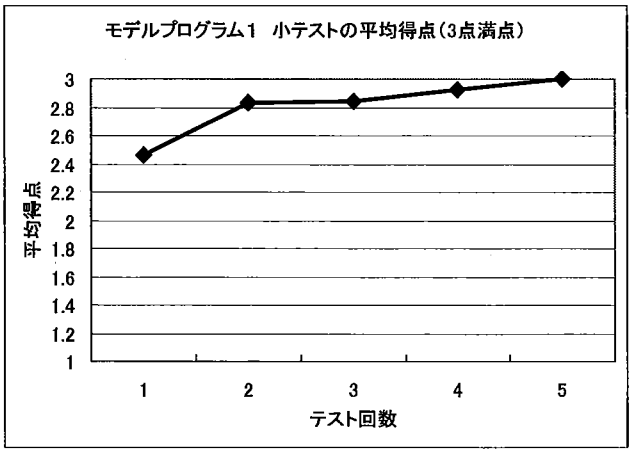

図２ モデルプログラム１の平均点の推移

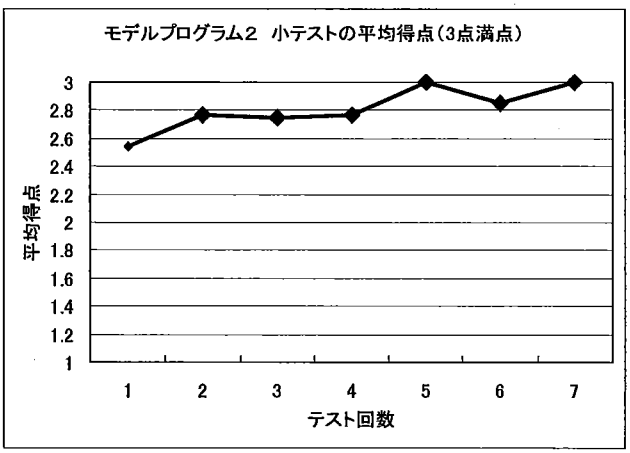

図３ モデルプログラム２の平均点の推移

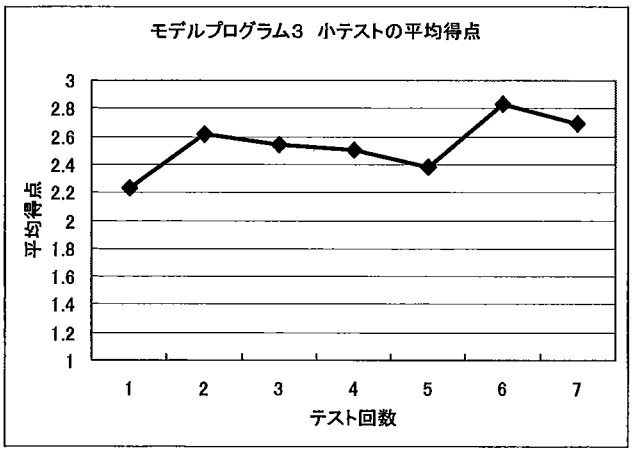

図４ モデルプログラム３の平均点の推移

モデルプログラム１の小テストの平均得点のグラフが一般的な学習曲線に近い形であるのに対して，モデルプログラム２および３のグラフは順調に増加していっていないことが分かる。特に，モデルプログラム３ではその傾向が顕著である。これは，内容が段々と難しくなっているのに加えて，小テストによる書き取りテストが多少ともマンネリ化してきていたのかもしれない。

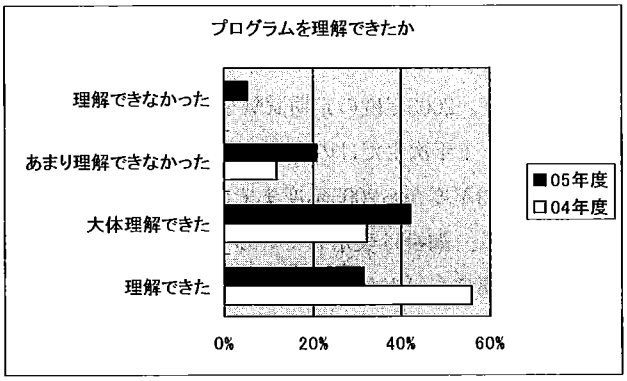

図５　アンケート結果１

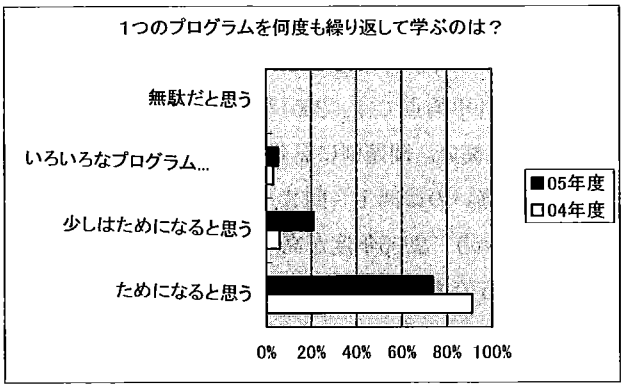

図６　アンケート結果２

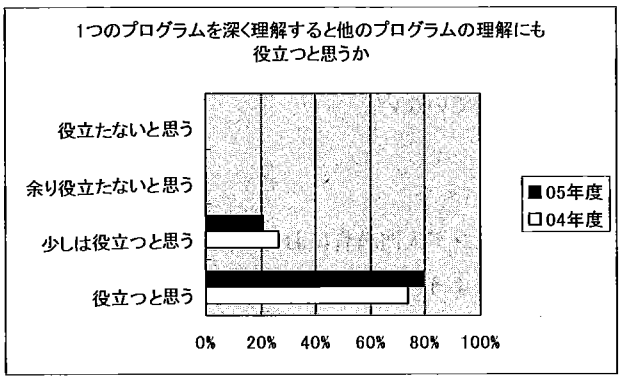

図７　アンケート結果３

４．評価

４．１　アンケート調査による比較

2004年度および2005年度の取り組みに対して，学生がどのように感じているかをアンケートで調査した。

図５の「モデルプログラムを理解できたか」に対する回答は，2004年度の方が理解度が高い。また，図６の「繰り返し学習がためになるか」に対する回答でも2004年度の方が肯定的な回答が多かった。モデルプログラムの本数が2004年度は１本であったのに対して2005年度には３本に増加していることが影響していると思われる。これに対して図７の「他のプログラム理解に役立つか」に対する回答は，2005年度の方が肯定的であった。「これだけは理解して書けるようにしよう！」というモデルプログラムが１本だけの場合と３本ある場合では学生たちの受け止め方，感じ方も相当異なるはずである。

４．２ 定期試験の成績の比較

2004および2005年度の取り組みの成果を客観的に評価するため，2003年度とほぼ同じ問題で2004年度，2005年度の定期試験を実施した。以下のデータからは再履修の学生のデータを除いて，１年次生だけの成績を集計している。

表５に2003年度から2005年度までの定期試験の問題別平均点の推移を示す。３回の試験問題のうち，網掛け表示している2005年度の問２だけが他の年度と異なっている。

従来のやり方の2003年度と比較して2004年度および2005年度の成績がかなり上がっていることが分かる。また，2004年度と2005年度とを比較すると，合計平均点では，2004年度の方が0.53点高い。問題別に見ると，2004年度が高いのは問１，問３，問４，問６であり，2005年度が高いのは問２，問５である。その差が非常に微妙であるため，モデルプログラムの本数を，一番難しい部分（繰り返し）の１本のみとするか，「順次」，「分岐」，「繰り返し」というアルゴリズムの基本の３本とするべきかは明らかとなっていない。

表５ 定期試験の成績の推移

大問	小問	2003年度	2004年度	2005年度	問題内容
問１	(1)	3.1	4.12	3	用語を問う
	(2)	3.17	3.52	3.33	
問２	(1)	2.27	3.6	3.58	2005年流れ図
	(2)	1.93	1.28	3.75	他の年バグ探し
問３	(1)	1.73	1.72	1.83	printf()の出力結果を問う
	(2)	1.3	1.36	1.25	
	(3)	1.2	1.12	0.58	
	(4)	0.6	0.48	0.42	
	(5)	1	1.08	0.33	
問４	(1)	5.5	6	6	プログラムの流れを追い実行結果を求める
	(2)	4.67	5.84	4.75	
	(3)	0.8	1.68	1.5	
	(4)	4.73	5.16	5.75	
	(5)	6.13	7.84	8.5	
	(6)	0.7	1.56	1.5	
	(7)	0.6	1.56	1.5	
	(8)	0.97	0.84	1	
	(9)	1.7	1.8	1.25	
	(10)	1.5	3.36	2.75	
	(11)	0.7	1.44	1.25	
問５	(1)	2.77	2.92	3.42	プログラムの穴埋め
	(2)	2.3	2.84	3.5	
問６		4.43	5.4	5.25	授業評価・意見
合計		53.8	66.52	65.99	平均得点

５．まとめ

プログラミング入門教育において，多数の例題プログラムの中から，最も重要であることがらを含むようにモデルプログラムを設定し，それを集中的に学習することによって，プログラミング全体の理解を深めさせるという試みを２年間に渡って実践した。１年目は繰り返し処理を理解するためのモデルプログラムを１本，２年目は「順次」「条件分岐」「繰り返し」の３つの基本を理解させるためのモデルプログラムを３本決めた。客観テスト（定期試験）を通じて，従来の授業のときの平均点と比較して１年目，２年目の両年度ともに12点以上成績が向上した。また，アンケート調査の結果から，学生たち自信が，「モデルプログラムを重点的に繰り返して学ぶ」ことが役に立ったと感じていることも明らかとなった。

参考文献

1）梶浦，“プレゼンを中心にしたプログラミング教育の実践”，平成12年度情報処理育研究集会講演論文集 pp.79～82，2000.

2）梶浦，“プレゼンを中心にしたプログラミング教育の実践（2）”，平成14年度情報処理教育研究集会講演論文集, pp.198～201，2002.

3）梶浦，“プレゼンを中心にしたプログラミング教育の実践（3）”，平成14年度情報処理教育研究集会講演論文集, pp.112～115，2003.

4）梶浦，“プログラムの繰り返し学習による効果”，信学技報VOL.104，No.342, pp.85～88．2004.

5）梶浦，“プログラミング教育用Web-CAIの試作 “，平成11年度情報処理教育研究集会講演論文集, pp.452～455，1999.

Model Programs for Repetitive Learning of a Computer Programming

Fumio KAJIURA

Dept. of Computer Science and Mathematics,
College of Science and Industrial Technology,
Kurashiki University of Science and the Arts,
2640 Nishinoura, Tsurajima-cho, Kurashiki-shi, Okayama 712-8505, Japan

(Received September 30, 2005)

This paper discusses on the importance and validity of repetitive learning of a few model programs for computer programming which are used as the base of various computer programs. The author applied this idea to an introductory computer programming class in the dept. of Computer Science and Mathematics for two years. By fully acquiring a few model programs, students were able to learn programming more. The term-end examination result of this fiscal year, the last fiscal year and the fiscal year before last are compared, and it becomes clear that repetitive learning of appropriate model programs is very effective for learning computer programming.