5F-07

プログラミングを必要とする専攻科目の観点からのプログラミング導入教育の補完と再学習

辻　康孝†

九州大学大学院工学研究院†

1. はじめに

理工学系分野では，一年次にプログラミング導入教育でその基本概念や文法を学習し，その後，専攻課程で実践的なプログラミングとその利用法を，例えば数値計算のような専攻科目の中で学習する．しかし導入教育では学習言語の文法を網羅的に学習し，各文法の確認のみの演習課題に取り組む場合も多いため，受講学生の多くが実際の数値計算アルゴリズムの実装に必要なプログラミングスキルが身に付いていないことが多い．また初期段階の文法理解が不完全なため，その後の理解も不安定な学生も多い．

本稿では，C 言語による実践的なプログラミングを行う専攻科目の観点から，導入教育での訓練が不十分で文法理解も不安定な学習者を考慮した専攻科目の一授業設計法とその学習成果について報告する．

2. 専攻科目の中でのプログラミング

データ分析スキルが重要視される中，非情報系分野でもプログラミングの知識が必要不可欠なものとなっている．元来，理工系分野では一年次にプログラミングの基礎を学んだ後，アルゴリズムや数値計算等の専攻科目授業が実践的なプログラミングを学ぶ場となっている．さらに専攻によっては，卒業研究でもプログラミングの知識が必要なテーマも多い．情報系分野同様，プログラミングは重要スキルとなっている．

一方，導入教育では，学習言語の文法を網羅的に学習し，各文法項目の確認のみの演習課題に取り組むといった単調な授業（作業）になりがちとなり，学生の学習の動機づけの維持が難しい．また，配列操作等の個々の文法項目の習熟度も低いため，ある程度の長さ（複雑さ）のプログラムの読解や記述が困難な学生も多い．その結果が，数値計算等の専攻科目授業や卒業研究に影響する．

Complementing and re-learning introductory programming course from a perspective of major subjects requiring programming

†Faculty of Engineering, Kyushu University

3. 対象とする学生および専攻科目

著者は，九州大学工学部機械航空工学科（機械工学コース）で数値計算の授業を担当している．本科目は専攻内で実践的なプログラミングを学ぶ機会となっている．カリキュラムの変更に伴い，現在では，以下のようになっている．

・講義名「数値解析応用」（選択・1 単位）
・3 年クォータ科目・秋学期（7 回＋試験）
・評価：各授業回の課題，試験
・Moodle，C，IDE：Xcode(Mac)／Eclipse(Win)

本コース 3 年生は，まず「数値解析基礎」（3 年春期・必修・15 回）で理論やアルゴリズム導出を学び，その中でさらに数値計算のアルゴリズム設計と実装を学びたい学生が受講する．直近 2 年間では 35 名前後（本コースの 20～30%）が受講している．選択科目であることと受講率の低さから，当初はプログラミングが得意な学生が受講していると考えたが，実際はそのような学生は少数であり，プログラミングを再学習する機会と捉えている学生が多かった．

4. 授業設計

4.1 導入教育の理解度に関するアンケート調査

導入教育の指導方法にもよるが，文法理解の不安定さは比較的早い段階でのつまずきに起因すると考える[1]．

そこで本授業では，導入教育の学習内容（C 言語の文法項目）をどの程度理解できたかを問うアンケートを授業初回に実施している．その結果を図 1，2 に示す．図 1 は特に「順次構造（逐次処理）」に関する項目を抽出し，誤ったソースコードを提示し，間違いであることの認識と修正できるかについて尋ねている．図 2 は，その他の項目をまとめたものである．図 1 から，大多数の学生が最も基本的な順次構造が理解できていない可能性がうかがえる．図 2 を見ると，順次構造以降の学習項目を理解できない学生は少数だが，プログラムの中で使いこなす自信がない学生が大多数となっている．ここで図 2 の H を 3，M を 2，L を 1 として，各自の平均を求めたところ，次の結果が得られた（2020/2021 年）．

・順次構造が H の学生の平均：2.85/2.83
・順次構造が L/M の学生の平均：2.27/2.12
この結果から，順次構造を正しく理解できていれば，（簡単なプログラムだったら）その他の基本項目も正しく利用できる可能性が高いことがいえる．その他の特徴として，一次元配列は理解できても実際に使いこなせない学生の割合が高いことが分かった．

図 1 に示した間違いを正しく訂正できた学生割合は，導入教育での学生の理解度を確認する一つの指標として利用できると思われる．

4.2 授業構成

基本的な数値計算では，特に複数の変数に対する代入処理をソースコードの中で正しい順番で記述できるかが重要になる．多くの場合，それらの処理は反復処理の中で実行される（例えばフィボナッチ数列の計算）．そのためには，図 1 で示した順次構造の理解が極めて重要となる．またより実践的な数値計算では，反復処理の中での配列計算がメインとなる．

受講学生の制御構造に関する実践力の不安定さは，前の考察から順次構造を正しく理解できていないことに起因すると考えられる．そこで早い段階に順次構造（なぜ図 1 が誤りなのか）を指導する．一方，他の制御構造は，簡単に復習した後，数値計算の該当する項目の演習課題で身に付かせる． 配列操作に関しては，まず疑似乱数を用いたデータ処理で慣れさせた上で，具体的な数値計算に取組ませる方針を取る．

補完内容としてユーザ定義関数を指導する（C では必然的にポインタの理解も必要）．以上の検討に基づいて，C 言語の再学習と数値計算実装を同時に指導する授業の構成を表 1 に示す．

4.3 授業成果（2021 年度分）

評価は表 1 の各数値計算項目の演習課題と，演習課題に自身で取り組んだことの裏付けとなる基本知識を確認するための試験の両方で行った．両指標とも得点率 80%以上が 11 名（全 35 名），60%以上では 20 名，一方両方とも 50%以下が 8 名であった．やはり図 1,2 の理解度調査の合計値が低い学生ほど，特に演習課題の評価において低い傾向にあった．

また提出されたソースコードには，間違いの元となるあるいは冗長な記述をする学生も多かった．将来的には記述スタイルの指導も取り入れる必要があると考える（例えば[2]）．

5. おわりに

本稿では，プログラミングを用いる専攻科目の観点から，プログラミング導入教育の問題点を指摘した．またプログラミングスキルが不安定な学生を考慮した数値計算科目の一授業設計法を提示した．Python の幅広い普及に伴い，数値計算以外の専攻科目でもプログラミングを取り入れた学習の機会が増加することから，導入教育の理解が不十分な学生に対する学習の補完や再学習が，今後は重要になるものと考える．

表 1: 授業構成（2021 年度は 6 回で実施）

	C 言語再学習(*補完)	数値計算
1	型と変数 標準入出力	ガイダンス
2	**順次構造**	*IDE の使い方 工学情報分野での数値計算
3	数学関数 条件分岐，多分岐	誤差
4	定回反復	乱数 モンテカルロ法
5	不定回反復	ニュートン法，二分法
6	**一次元/二次元配列** *ポインタ変数	数値積分
7	*ユーザ定義関数	常微分方程式の数値解法

```
#include <stdio.h>
int main ()
{
  double r, height;
  double V = 3.14*r*r*height;
  r = 1.0;
  height= 5.0;
  printf("V=%f ￥n" , V);
  return 0;
}
```

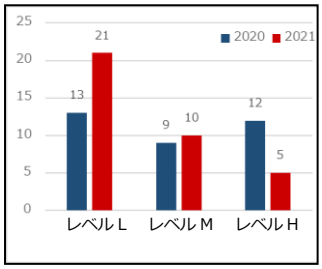

図 1 順次構造に関するアンケート（L：なぜ間違っているのか理解できない M：訂正できない H：正しく訂正できた，2020 年 N=34，2021 年 N=36，履修中止者も含む，2020 年と 2021 年では質問が一部異なる）

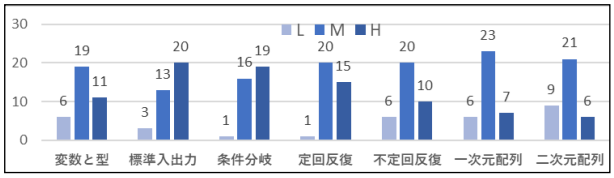

図 2 理解度アンケート（2021 年度のみ，2020 年度省略，L：理解できなかった M：理解できたが，簡単なプログラムの中でも正しく使う自信がない H：簡単なプログラムでなら正しく使うことができる）

参考文献

[1]辻 康孝，“大人数クラスにおけるプログラミング演習の実施と学習者の学習意欲の維持”，基幹教育紀要，vol.4，pp.77-88，2018.
[2]Lloyd Cawthorne, “Invited viewpoint: teaching programming to students in physical sciences and engineering”, J. of Mater. Sci., vol.56, pp.16183–16194, 2021.