プログラミングスタイルを指導するための演習システム

三重大学大学院 工学研究科 間座秀幸 木村英朗* 北英彦 高瀬治彦 林照峯
kanza@ce.elec.mie-u.ac.jp

1. はじめに

プログラミング能力を身に付けるためには，プログラミング言語の各機能について学んだ後に，その具体的な使い方を習得する必要があり，そのために．プログラミング演習が行われている．

プログラミング演習における学習者の第一の目標は，課題で指定された動作をするプログラムを作成することである．しかし，それだけでなく，正しく動作するプログラム，また，修正しやすいプログラムを書くためには，読みやすいプログラムを書くことが重要である．読みやすいプログラムとするためには，一貫したプログラミングスタイルを用いる必要がある．

従来のプログラミング演習においては，正しく動作するプログラムを作成させることに重点がおかれ，演習時間が十分でない，あるいは，講師の手がそこまで回らないなどの理由で，プログラミングスタイルに関する指導が十分には行われていないことが多い．

本研究では，プログラミングスタイルを指導するための演習システムを提案する．プログラミングスタイルには，守るべき項目がいくつかあるが，その中でも最も基本である「字下げを適切に行うこと」と，計算機での処理が簡単である「マジックナンバーを使わないこと」の 2 つを本研究の対象とする．

2. 学習者のプログラミングスタイルの現状

プログラミングスタイルに関して注意すべきこととして，以下のものがある[1,2]．

- 字下げを適切に行うこと
- マジックナンバーを使わないこと
- コメントを適切に付けること
- 変数，関数などに適切な名前を付けること

本研究では，先に述べたように最初の 2 つを対象とする．

まず，学習者がプログラミングスタイルにどれぐらい注意をはらっているかに見るために，下記の講義のプログラム作成の演習において，学習者が実際に作成したプログラムに対して調査を行った．

講義：2005 年度前期 プログミング演習Ⅱ
対象：電気電子工学科 2 年生 87 人
プログラミング言語：C 言語

なお，この講義の演習では，正しく動作するプログラムを作成させることに重点がおかれ，講師は学習者が作成した個々のプログラムに対して，プログラミングスタイルに関する指摘を行うことはできていない．

*平成 18 年 4 月からトランスコスモス株式会社

学習者が作成したプログラムに対して，字下げが適切に行われているかを調べた結果を表1に示す．学習者が作成したプログラム全体のうち，55%のプログラムで字下げが適切に行われていないことが分かった．

プログラミングスタイルの中で最も基本であり，正しく使うことが難しくないと思われる字下げであっても，約半数の学習者が正しく使えていないことが分かった．

表 1. 字下げの使用に関する現状

課題番号	提出されたプログラム	字下げが不適切なプログラム	
課題1	84 個	46 個	55%
課題2	84 個	41 個	49%
課題3	83 個	54 個	65%
課題4	82 個	41 個	50%
合計	333 個	182 個	55%

学習者が作成したプログラムに対して，マジックナンバーが使われているかどうかを調べた結果を表 2 に示す．5 月 12 日の 1 つ目の課題（課題 5）では，問題に配列の要素数にマクロを用いるよう指示されていたため，マジックナンバーを使っている学習者はいなかった．しかし，同じ日に出題された次の課題（課題 6）では，課題 5 と同様な問題だったにも関わらず，27%の学習者がマジックナンバーを使用していた．その後，1 ヶ月後（課題 7）では 84%の学習者が，2 ヶ月後（課題 8）では全ての学習者がマジックナンバーを使用していた．

プログラミングの初心者はマジックナンバーを使うことによって発生する問題について実感を持つことが難しく，マクロの使い方や使う目的を一度説明しただけでは，「マジックナンバーを使ってはいけない」という意識を持たせることができないということが分かった．

表 2. マジックナンバーの使用に関する現状

演習日 課題番号	提出されたプログラム	マジックナンバーを使用したプログラム	
5 月 12 日 課題 5	87 個	0 個	0%
5 月 12 日 課題 6	86 個	23 個	27%
6 月 9 日 課題 7	85 個	71 個	84%
7 月 7 日 課題 8	84 個	84 個	100%

3. 字下げの指導

学習者に字下げを指導するためには，字下げの仕方に関する明確なルールが必要である．著者らが所属する学科ではプログラミング言語として C 言語を教えているので，本研究では C 言語を対象とする．

字下げの仕方に関する明確なルールを定めるためには，以下の 2 つを決める必要がある．

- 字下げをする箇所，すなわち，どこで字下げをし，どこで字下げをもどすか
- 字下げをするときどれだけの深さにするか

そこで，代表的なプログラミングスタイルである K&R スタイル，BSD スタイルについて調べた [3]．この 2 つは，制御文の有効範囲を一段字下げするという点で共通しているが，そのときの字下げの深さは異なっている．

本研究では，字下げの箇所については，上記の 2 つのスタイルと同じ制御文の有効範囲で一段字下げをするというルールを採用する．また，字下げの深さについては，共通したものがないので，プログラムを作成した学習者の意図に従って決めることにする．すなわち，学習者の作成したプログラムをシステムが解析することで，字下げの深さについての学習者の意図を読み取り，それをルールとして用いる．

本研究の演習システムでは，学習者の提出したプログラム中に，これらのルールに合わないものがあった場合に，それを学習者に指摘しプログラムの修正を促す．

4. マジックナンバーの指導

マジックナンバーとは，定数，配列の大きさなど，プログラム中に現れる数値を表す数字列のことをいう．例えば，図1のプログラム中の網掛けになっている数字列がマジックナンバーである．

```
/* 5 教科の平均が 60 点以上だと合格と表示
   するプログラム */
#include <stdio.h>
int main(void)
{
    int i, sum=0;
    int vx[5];
    for(i=0; i<5; i++) {
        printf("%d 教科目：",i+1);
        scanf("%3d",&vx[i]);
        sum += vx[i];
    }
    if((sum/5)>=60) {
        printf("合格\n");
    }
    return(0);
}
```

図 1. マジックナンバーが使われているプログラム

プログラム中にマジックナンバーがあると，以下に示す理由により，プログラムの修正が困難になる．

- マジックナンバーは，その数値の持つ意味が分かりにくい．例えば，図1に示すプログラムの中の合格基準を示す「60」がこれにあたる．
- 同じ意味をもつマジックナンバーがプログラム中に何個もあると，その数値を変更する時に手間がかかる．例えば，図1に示すプログラムでは，「5」という数値が 3 箇所で出現している．これらは全て教科の数を意味しているが，もし教科の数を変更する場合には，3 箇所すべてを変更しなければならなくなる．
- プログラム中に同じ数値だが違う意味を持つマジックナンバーがある場合には，プログラムを修正する時に，一つ一つその意味を確認しなければならない．

プログラム中に表れた数値ではあってもマジックナンバーと見なしてはいけないものがある．以下にそれを列挙する．

- マクロ定義，定数宣言，列挙体宣言で使われている数値を表す数字列
- 注釈内で使われている数値を表す数字列
  `/* 5 教科の平均点 */`
- 文字列で使われている数値を表す数字列
  `“5 教科の平均点%.2f”`

また，この他にも，名前ではなく数値で表現した方が分かりやすいことがある．この場合は，本研究では，マジックナンバーとは考えないことにする．以下のものがそれらにあたる．

```
/* 5 教科の平均が 60 点以上だと合格と表示
   するプログラム */
#include <stdio.h>
#define NUMBER 5      /* 教科数 */
#define BOUNDARY 60   /* 合格点 */
int main(void)
{
    int i, sum=0;
    int vx[NUMBER];
    for(i=0; i<NUMBER; i++) {
        printf("%d 教科目：",i+1);
        scanf("%3d",&vx[i]);
        sum += vx[i];
    }
    if((sum/NUMBER)>=BOUNDARY) {
        printf("合格\n");
    }
    return(0);
}
```

図 2. マジックナンバーが使われていないプログラム

- 0, 1 の数値
- switch 文で用いる case に続く数値

```
switch(sw) {
    case 1: ～
    case 2: ～
```

- 変数や配列の初期化

```
int x = 100;
int a[NUM] = {10, 20, 30, 40, 50};
```

本研究では，マジックナンバーを使わないようにするための方法として，マクロを使う方法を採用する．マクロを使ってマジックナンバーを使わないようにした例を図 2 に示す．このプログラムは図 1 に示したマジックナンバーが使われているプログラムを書き直したものである．このように，数値の代わりに分かりやすい名前を付けることにより，プログラムが読みやすさが向上する．

5. プログラミング演習システム

本研究で提案するプログラミング演習システムでは，学習者の提出したプログラムに対して，以下の 2 点についてチェックを行う．

- 字下げが適切に行われていること
- マジックナンバーが使われていないこと

演習システムでチェックした結果は分かりやすい形で学習者に提示する．それぞれのチェック項目について，学習者のプログラムに不適切な行があった場合には，それを学習者に指摘しプログラムの修正を促す．また，字下げやマジックナンバーに関する知識が不十分な学習者のために，学習者の要求に応じてそれらに関する説明やヒントを提示する．本演習システムの処理の流れを図 3 に示す．

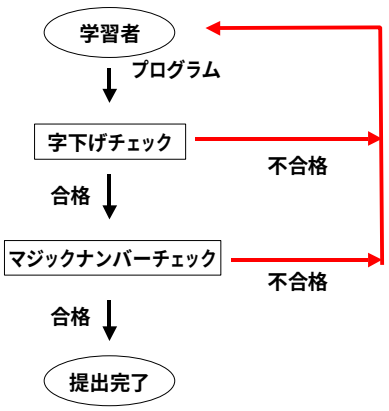

図 3 演習システムの処理の流れ

5.1 字下げのチェック

本演習システムでは，学習者の意図する字下げの深さを，学習者の作成したプログラムから読み取るために以下の流れで字下げのチェックを行う．プログラム中のある行が字下げされるべき回数をその行の字下げのレベルと呼ぶことにする．

① 各行の字下げの深さを調べる．
② 各行の字下げのレベルを調べる．
③ 各行を字下げのレベルで分類する．
④ 字下げのレベルごとに，基準となる字下げの深さを多数決で決定する．
⑤ 字下げの各レベルに対して，基準の字下げの深さが適切であるかどうかを調べる．
⑥ 基準の字下げの深さに一致しない行があるかどうかを調べる．

図 4 に学習者に提示する字下げに関するチェック結果を示す．

(a) 学習者の作成したプログラムから読み取った，字下げの各レベルに対する基準となる字下げの深さを表示する．
(b) 字下げのレベルの基準となる字下げの深さが適切でないときは，その字下げのレベルに該当する行の字下げの深さを修正するように指導する．該当する行を青色の文字で表示する．
(c) 基準となる字下げの深さに対して，字下げが適切でない行があるときは，その行の字下げの深さを修正するように指導する．該当する行を赤色の文字で表示する．

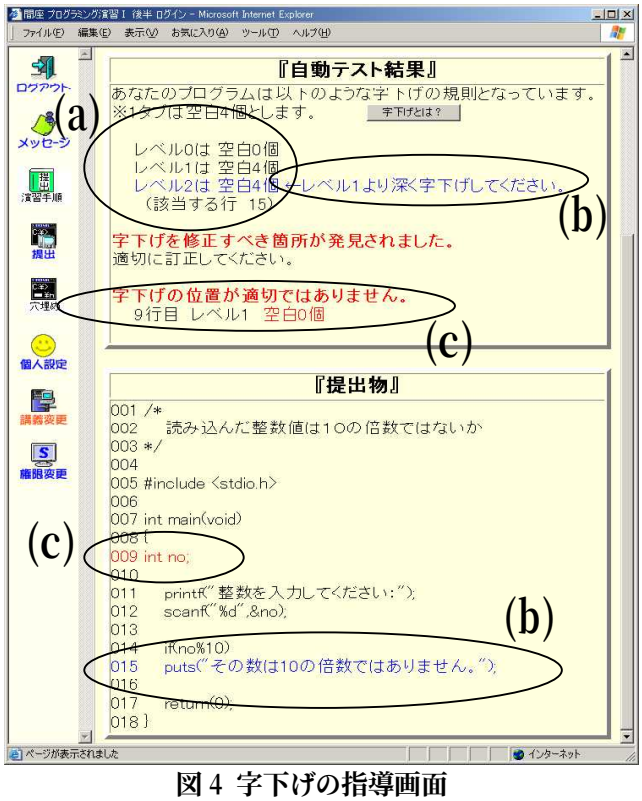

図 4 字下げの指導画面

5.2 マジックナンバーのチェック

本演習システムでは，学習者の作成したプログラム中にマジックナンバーがあった場合には，図 5 中の線で囲まれた箇所で示すように，マジックナンバーとそのマジックナンバーがある行の番号をマゼンタ色で表示する．

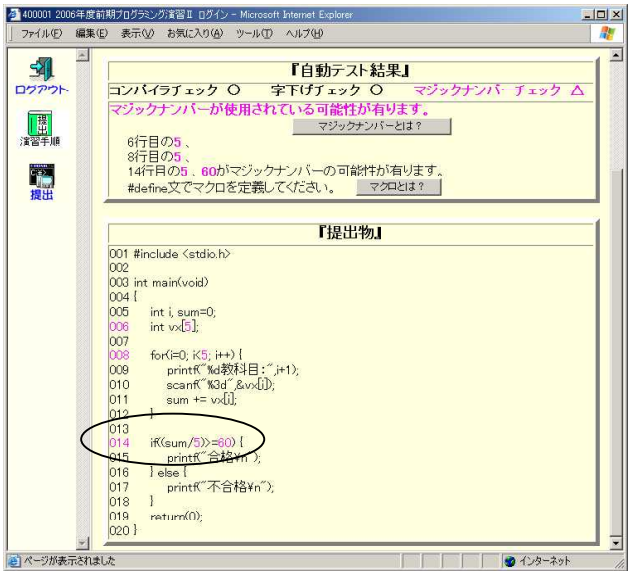

図 5 マジックナンバーの指導画面

6. 運用結果

本研究で提案するプログラミング演習システムの有効性を確認するために，プログラミング演習の講義で実際に運用を行った．

講義：2006 年度前期 プログラミング演習 II
対象：電気電子工学科 2 年生 91 人

1 回の演習で，2 問を順番に出題した．演習システムに字下げを指摘された学習者の人数と字下げを修正できた学習者の人数を示す．

問題 1 プログラムを提出した学習者 83 人
字下げを指摘された学習者 21 人
字下げを修正できた学習者 16 人
問題 2 プログラムを提出した学習者 63 人
字下げを指摘された学習者 9 人
字下げを修正できた学習者 6 人

演習後に行ったアンケートの結果を以下に示す．

① 今日の演習をはじめる時点で，字下げを適切に使用すべきだと考えていましたか．
はい 22 人 どちらかといえば，はい 37 人
どちらかといえば，いいえ 14 人 いいえ 18 人

② 今日の演習の中で，あなたが作成したプログラムの中に字下げに関して不適切な部分があることを，プログラミング演習システムに指摘されましたか．
はい 33 人 いいえ 49 人 未提出 8 人

③ プログラミング演習システムに字下げに関して不適切だと指摘されたところは修正することができましたか．
はい 26 人 いいえ 17 人
指摘されていない 41 人 未提出 6 人

④ これから，プログラムを作成するときには，字下げを適切に使用しようと思いますか．
はい 54 人 どちらかといえば，はい 29 人
どちらかといえば，いいえ 7 人 いいえ 1 人

⑤ 今日の演習をはじめる時点で，マジックナンバーをなるべく使用しないようにプログラムを作成しようと考えていましたか．
はい 11 人 どちらかといえば，はい 17 人
どちらかといえば，いいえ 25 人 いいえ 37 人

⑥ 今日の演習の中で，あなたが作成したプログラムの中に，マジックナンバーを使用している可能性があると，プログラミング演習システムに指摘されましたか．
はい 60 人 いいえ 18 人 未提出 12 人

⑦ プログラミング演習システムにマジックナンバーを使用している可能性があると指摘されたところは修正することができましたか．
はい 22 人 いいえ 46 人
指摘されていない 17 人 未提出 6 人

⑧ これから，プログラムを作成するときには，なるべくマジックナンバーを使わないようにしようと思いますか．
はい 32 人 どちらかといえば，はい 39 人
どちらかといえば，いいえ 15 人 いいえ 5 人

アンケート結果の②③から，演習システムの指摘により，学習者は，自分が作成したプログラムの字下げが適切ではない箇所を把握することができたこと，また，その箇所を自分自身で修正することができたことが分かる．

アンケート結果の⑥⑦から，マジックナンバーについては，指摘された箇所を学習者は自分自身で適切に修正できなかったことが分かる．マジックナンバーについては，説明やヒントを提供するだけでは不十分で，講師による説明が必要だと考えられる．

アンケート結果の①④から，このシステムの使用前後に学習者が字下げを適切に使用しようと意識が変化したこと，また⑤⑧から，マジックナンバーを使わないようにしようと意識が変化したことが分かる．

7. まとめ

本研究では，講師のプログラミングスタイルの指導を支援するための演習システムを開発した．プログラミング演習の講義で実際に運用を行うことで，その有効性を確認した．

参考文献

[1] 林晴比古，C / C++によるプログラミングスタイルブック，ソフトバンクパブリッシング株式会社，2000
[2] Brian W. Kernighan, Rob Pike, 福崎俊博訳，プログラミング作法，アスキー，2000
[3] 経済産業省 組み込みソフトウェア開発力強化推進委員会 独立行政法人情報処理推進機構ソフトウェア・エンジニアリングセンター，コーディング作法ガイド，http://sec.ipa.go.jp/index.php，2005