

[研究ノート]

## プログラミング初学者における Parson's Puzzles とプログラミングスキルの関係

若山 公威

### 1. はじめに

情報分野を専門とする者に限らず一般情報教育の一環として、プログラミング教育を実施する必要性が指摘されており<sup>[1]</sup>、文系大学においてもプログラミング教育が広がっている。初学者向けに様々なプログラミング言語も用意されているが、初学者向けではないJava言語などの一般のプログラミング言語を教育するうえで、初学者のプログラミングに関するスキル間の関係を調べ、その結果を教育に生かそうとする試みも行われている<sup>[2-7]</sup>。

その中で、Parson's puzzles<sup>[8]</sup> という並べ替え問題についても調査がされている。コード記述やコードトレース問題との関係が調べられているものの、まだ十分な分析がされていない。並べ替え問題はプログラムコード全体を書かせる問題や動作の説明をさせる問題よりも採点を機械的に行うことができるため、LMS (Learning Management System) への適用が容易である。そのため、コードを書いたり説明したりするスキルの測定や練習として並べ替え問題を使うことができると、学習者の演習に有効的に利用することが可能となる。したがって、並べ替え問題を解くスキルと他のプログラミングスキルの関係を調査することが必要である。

本研究では、Java言語におけるParson's puzzlesと他のプログラミングスキルとの関係を測定するための問題を作成し、プログラミング初学者に回

答してもらった実験を行い、その結果を分析した。

## 2. 関連研究

Parsonsらにより Parson's puzzles という並べ替え問題が、初学者向けのドリルとしての用途を想定して提案されている<sup>[8]</sup>。この並べ替え問題は、プログラムを構成する全てまたは一部のコードをランダムな順に提示しておいて、被験者が正しい順に並べ替える問題である。余分なコードを選択肢として追加することも可能である。この Parson's puzzles のツール作成や分析がいくつか行われている。

Dennyらの研究<sup>[5]</sup>では、Parson's puzzles、コード記述、コードトレースの各問題の得点の相関を調べている。各問題とも、繰り返し文 (for) を使った配列要素の取得を行う問題である。この結果、Parson's puzzles とコード記述間に著しい相関がみられた。そして、コード記述とコードトレース間には低い相関がみられ、Parson's puzzles とコードトレース間も低い相関がみられたと報告している。

Simonらの研究<sup>[6]</sup>では、コードを読み取る問題の得点と、コードを書く問題の得点の間に強い相関がみられた。このコードを書く問題には Parson's puzzles も含まれているものの、個別の分析は行われていない。

Lopezらの研究<sup>[7]</sup>では、コードトレース、コード記述、読み取りの各スキル間の階層を調べている。その結果、Parson's puzzles は繰り返しを含むコードトレースより下の階層にあるスキルとしている。ただし、この結果には Parson's puzzles の難易度が影響しているかもしれないと述べている。この問題では、括弧 ( {, } ) の位置をあらかじめ指定しておいたため6点中4.9点 (83%) の高い正解率となったものの、被験者が括弧の位置も答えるようにすると難しくなるかもしれないと述べている。しかし、その後の調査は行われていない。

Ihantolaらは、Parson's puzzles をネット上で体験することができるツール

を作成しているが、プログラミングスキルとの関係については調査していない [9] [10]。

### 3. 実験方法

#### 3.1. 実験で調べるスキル

文献 [7] ではプログラミングに関する様々なスキル間の関係を調べている。そのうち、本実験では以下のスキルを取り上げる。

##### ① Basic

基本構文を理解できるスキル。例えば、プログラミング言語の用語が分かったり、シンタックスエラーを検出したりできるスキル。

##### ② Tracing1

for、whileなどの繰り返しを含まないプログラムコードを自身でシミュレートし、変数の変化状況や、画面表示結果を答えることができるスキル。

##### ③ Tracing2

for、whileなどの繰り返しを含むプログラムコードを自身でシミュレートし、変数の変化状況や、画面表示結果を答えることができるスキル。

##### ④ Writing

プログラムコードを記述できるスキル。

##### ⑤ Explain

プログラムコードの動きを説明できるスキル。

##### ⑥ Parson's1

for、whileなどの繰り返しを含まないParson's puzzles問題を解くことができるスキル。文献 [7] ではSequenceスキルに入れているが、本研究では名称を変更した。

##### ⑦ Parson's2

for、whileなどの繰り返しを含むParson's puzzles問題を解くことができる

スキル。

### 3.2. 実験条件

名古屋外国語大学2012年度1期の「JAVAプログラミング」授業の中間試験と期末試験にて実施した。受講者は3年生および4年生の計27名であり、全員プログラミング言語の初学者である。中間試験はif文とfor文の授業を行った後の9回目の授業時間に、期末試験はwhile文と配列の授業を行った後の14回目の授業時間に実施した。受験者数は、中間試験23名、期末試験18名である。

測定問題は文献 [4] [7] を参考に作成した。ただし、今回の試験ではノートと授業中に配付した資料の持ち込みを許可したため、Basicスキルを問う問題では、単純な文法の知識を問う問題は出さず、シンタックスエラーや実行時エラーとなる部分を指摘させる問題とした。Parson's問題は、中間試験、期末試験とも、余分なコードを追加したタイプの問題とした。

### 3.3. 中間試験の測定問題

中間試験では、3.1節のParson's1を除く6スキルそれぞれ1問ずつ出した。Basic問題は、変数の宣言に関する間違いを指摘させる問題である。

Writing問題は、キーボードから正の整数 $N$ を入力し、その $N$ の約数を求めConsoleに表示するものである。

Explain問題は、2つの正の整数 $n1$ および $n2$ をキーボードから入力し、 $n1$ から $n2$ までの合計を求めConsoleに表示するプログラムを読み取り説明するものである。

Parson's2問題は、付録A.1.に示す。

### 3.4. 期末試験の測定問題

期末試験では3.1節の7スキルそれぞれ1問ずつ出したが、Tracing2問題の正解者はいなかったため、以降の評価処理では対象から省いた。

Basic問題は、ifにおける条件の書き方の間違いを指摘させる問題である。

Writing問題は、以下のような動きをするプログラムを記述するものである。

1. キーボードから人数を入力する。
2. キーボードから人数分の点数を入力する。
3. キーボードから整数を入力し、変数lowに代入する。
4. キーボードから整数を入力し、変数highに代入する。
5. 2で入力した点数のうち、low以上、かつ、high以下の点数のみをConsoleに表示する。

Explain問題は、配列要素10個に入っている整数に関して、0未満、0、0以上のそれぞれの個数を求めるプログラムを読み取り説明するものである。

Parson's1とParson's2の問題は、それぞれ付録A.2.、A.3.に示す。

## 4. 実験結果

### 4.1. 中間試験結果

中間試験の平均点、標準偏差を表1に示す。各スキルとも5点満点である。Explainの平均点が一番低いことが分かる。授業では、教員が出した条件を満たすプログラムを作成する演習のみしか行っていないため、プログラムを自力で読み取り、その意図を理解することには慣れていないことが原因と思われる。Basicに関しては、平均点は一番高いが標準偏差が大きく被験者間のばらつきが大きいことが分かる。

スキル間の相関係数を表2に示す。

表1 中間試験の平均点・標準偏差

	平均点	標準偏差
Basic	3.48	2.35
Tracing1	3.04	1.74
Tracing2	2.22	1.70
Writing	2.48	2.09
Explain	1.70	2.24
Parson's2	2.48	2.21

表2 中間試験各スキル間の相関

	Basic	Tracing1	Tracing2	Writing	Explain	Parson's2
Basic	—					
Tracing1	- 0.09	—				
Tracing2	0.09	0.07	—			
Writing	0.06	0.57**	0.33	—		
Explain	- 0.31	0.18	0.34	0.14	—	
Parson's2	- 0.25	- 0.26	0.44*	0.15	0.35	—

\*  $p < 0.05$ , \*\*  $p < 0.01$

この表から、Tracing1とWriting間に中程度の相関があること、Parson's2とTracing2間にも中程度の相関があることが分かる。繰り返しがあるトレースと、繰り返しがあるParson's puzzles間に相関があることから、問題に用いたプログラムの構造が影響している可能性が考えられる。ただし、今回のWritingプログラムは繰り返しを含んだものであるにも関わらず、Tracing2ではなくTracing1との相関が高いため、他の要素も影響している可能性もある。今回のWritingの問題は整数Nの約数を求めるものであった。この問題で回答されたプログラムを調べた結果、1からNまでの繰り返しをforを使って正しく書いている被験者は15名、forループの中でifを使ってNを割った余りが0であるかどうかの確認をしている被験者は7名であった。つまり、繰り返し処理は正しく使われていても、約数かどうか調べる処理ができない被験者が半数近くいることが分かる。約数の求め方は授業中の演習などで取り上げていないため、プログラミング能力とは別に数学の学力も影響していることも考えられる。

#### 4.2. 期末試験結果

期末試験の平均点、標準偏差を表3に示す。各スキルとも10点満点である。Parson's1の平均点は他の問題よりも大幅に高く、標準偏差が小さくなった。Parson's1の問題は3つの場合分けを行うif-else文を用いたプログラムである。このようなタイプのプログラムを授業中にサンプルあるいは課

題として複数出したため、全体的に点数が高くなり、ばらつきも減ったものと思われる。Basicも平均点は高めであるが、標準偏差が大きく、ばらつきが大きいことが分かる。資料持ち込みの試験であるにも関わらず、中間試験と同様に構文の詳細までの理解ができていない被験者がいると思われる。

スキル間の相関係数を表4に示す。この表から、Parson's2とWriting間と、Parson's2とExplain間にそれぞれ中程度の相関があることが分かる。WritingとExplain間にも中程度の相関がある。WritingとExplainの問題はともに、forのループ内でifを用いて配列要素の値により処理を変更するプログラムである。Parson's2はforの二重ループ内で、ifを用いて変数の値により条件分岐を行うプログラムである。3つともforループ内での処理という共通点があるため、相関が高くなったものと考えられる。

一方、ループをともに含まないParson's1とTracing1間にはほとんど相関がなかった。これは、Parson's1の問題は3つの場合分けを行うif-else文を用いたプログラムであり、授業中に何度か取り上げたプログラムの種類であるが、Tracing1は変数の宣言と代入のみを用いたプログラムであり、種類が全く異なるためと思われる。

表3 期末試験の平均点・標準偏差

	平均点	標準偏差
Basic	6.11	5.02
Tracing1	4.61	2.91
Writing	4.22	4.52
Explain	4.83	4.44
Parson's1	9.22	1.56
Parson's2	4.06	3.90

表4 期末試験各スキル間の相関

	Basic	Tracing1	Writing	Explain	Parson's1	Parson's2
Basic	—					
Tracing1	0.25	—				
Writing	0.38	0.17	—			
Explain	0.42	0.10	0.52*	—		
Parson's1	0.42	0.02	0.07	0.36	—	
Parson's2	0.34	0.00	0.52*	0.51*	0.00	—

\*  $p < 0.05$

## 5. まとめ

本論文では、Java 言語初学者を対象に Parson's puzzles と他のプログラミング能力との関係を分析した。その結果、中間試験では、Parson's2 と Tracing2 間に中程度の相関があった。期末試験では、Parson's2 と Writing 間と、Parson's2 と Explain 間、Writing と Explain 間にそれぞれ中程度の相関があることが分かった。

Lopez らの研究<sup>[7]</sup>によると、Parson's puzzles は繰り返しを含むコードトレースより下の階層で、Writing からは離れたスキルであるとしている。使用された問題は Parson's2 タイプではあるものの、括弧の位置があらかじめ示してあるものであった。これに対して、今回の実験では、括弧の位置をあらかじめ示さないタイプの問題を出した。その結果、期末試験のみではあるが、ともに繰り返しを含む Parson's2 と Writing 間において中程度の相関がみられた。括弧の位置を被験者が決めることにより、Writing スキルが必要な問題に近づいたものと考えられる。中間試験では相関は低くなったが、これは Writing と Parson's2 の問題のタイプが異なっていたためと考えられる。今後、タイプをそろえた場合の調査も必要である。

今回の結果から、Parson's puzzles と Writing の関係について、それほど離れたスキルではない可能性があることが分かる。その場合、Writing スキルを向上させるために Parson's puzzles による演習も有効に働くことが考えられる。

今後は、Java 言語以外のプログラミング言語、とくにプログラミング初学者向けの言語での調査もしていきたい。

## 参考文献

- [1] 情報処理学会情報処理教育委員会: 日本の情報教育・情報処理教育に関する提言 2005, <http://www.ipsj.or.jp/12kyoiku/proposal-20051029.html>, (参照 2012-

08-28).

- [2] Lister, R., Adams, E.S., Fitzgerald, S., et al.: A Multi-National Study of Reading and Tracing Skills in Novice Programmers, SIGCSE Bulletin 36(4), pp.119-150 (2004).
- [3] Whalley, J., Lister, R., Thompson, E., et al.: An Australasian Study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies, Proc. ACE2006, pp.243-252 (2006).
- [4] 山本三雄, 関谷貴之, 山口和紀: プログラミングのスキル階層に関する研究, 情報処理学会研究会報告, 2010-CE-104 No.3, pp.1-25 (2010).
- [5] Denny, P., Luxton-Reilly, A. and Simon, B.: Evaluating a new exam question: Parsons problems. Proc. ICER '08, pp.113-124, ACM (2008).
- [6] Simon, Lopez, M., Sutton, K. and Clear, T.: Surely we must learn to read before we learn to write!, Proc. ACE '09, pp.165-170 (2009).
- [7] Lopez, M., Whalley, J., Robbins, P. and Lister, R. : Relationships between reading, tracing and writing skills in introductory programming, Proc. ICER '08, pp.101-112, ACM, (2008).
- [8] Parsons, D. and Haden, P.: Parson's programming puzzles: a fun and effective learning tool for first programming courses, Proc. ACE 2006, pp.157-163 (2006).
- [9] Ihantola, P. and Karavirta, V. : Two-dimensional Parson's puzzles: the concept, tools and first observations, Journal of Information Technology Education and Innovations in Practice, Vol. 10, pp.1-14 (2011).
- [10] Ihantola, P. and Karavirta, V. : Open source widget for parson's puzzles, Proc. ITiCSE '10, pp. 302-302, ACM (2010).

## 付録

### A.1. 中間試験 Parson's2

次のプログラムを作成したい。

- キーボードから整数を入力し、入力された整数の行数分、Consoleに#を表示する。  
ただし1行目は1個、2行目は2個、と順に#の数を増やす。

選択肢を使って下の四角を埋めよ。ただし、すべての選択肢を使うわけではない。

```
import java.util.Scanner;
class Ex6 {
    public static void main(String args [] ) {
        Scanner scan = new Scanner(System.in);
        int num = scan.nextInt();
        int i;
        int j;
```



```
}
}
```

**【選択肢】**

```
    for (i=1; i<=j; i=i+1) {
    for (j=1; j<=i; j=j+1) {
    for (i=1; i<=num; i=i+1) {
    System.out.print("#");
    System.out.println();
    }
    }
```

## A.2. 期末試験 Parson's1

次のプログラムを作成したい。

1. 英語の点数を入力し、変数`english`に代入する。
2. 数学の点数を入力し、変数`math`に代入する。
3. 英語と数学の点数がともに80以上なら、「合格です」と表示する。

英語と数学の点数どちらか一方が80以上なら、「補習を受けて下さい」と表示する。

それ以外の場合は、「不合格です」と表示する。

選択肢を使って下記プログラムの四角を埋めよ。ただし、すべての選択肢を使うとは限らない。

```
import java.util.Scanner;
```

```
class Ex7 {
```

```
    public static void main(String args [] ) {
```

```
        Scanner scan = new Scanner(System.in);
```

```
int english = scan.nextInt();
```

```
int math = scan.nextInt();
```

```
System.out.println("合格です");
```

```
System.out.println("補習を受けて下さい");
```

```
System.out.println("不合格です");
```

```
    }  
}
```

### 【選択肢】

```
if (english >= 80) {  
  if (math >= 80) {  
    if (english >= 80 && math >= 80) {  
      if (english >= 80 || math >= 80) {  
        } else if (english >= 80) {  
        } else if (math >= 80) {  
        } else if (english >= 80 && math >= 80) {  
        } else if (english >= 80 || math >= 80) {  
        } else {  
        }  
      }  
    }  
  }  
}
```

### A.3. 期末試験 Parson's2

Consoleに右の表示を行いたい。

選択肢を使って下記プログラムの四角を埋めよ。

ただし、すべての選択肢を使うとは限らない。

```
+---+  
+---+  
+---+  
+---+  
+---+
```

```
class Ex2 {  
  public static void main(String args [] ) {  
    int i;  
    int j;
```

```
}  
}
```

**【選択肢】**

```
for (i=1; i<=5; i=i+1) {  
    for (j=1; j<=5; j=j+1) {  
        if (i==1) {  
            if (i==5) {  
                if (i==1 || i==5) {  
                    } else {  
                        System.out.print("+");  
                        System.out.print("-");  
                        System.out.println();  
                    }  
                }  
            }  
        }  
    }  
}
```