1C5-OS-13b-2

書かせること以外のプログラミング指導から見えてきたこと

Things unveiled by the instruction of programming except writing source codes

松本 慎平 *1 Shimpei Matsumoto　山岸 秀一 *1 Shuichi Yamagishi　加島 智子 *2 Tomoko Kashima　林 雄介 *3 Yusuke Hayashi　平 嶋宗 *3 Tsukasa Hirashima

*1広島工業大学情報学部
Faculty of Applied Information Science, Hiroshima Institute of Technology

*2近畿大学工学部
Faculty of Engineering, Kindai University

*3広島大学大学院工学研究科
Graduate School of Engineering, Hiroshima University

To efficiently support novice programming learners feeling programming difficult, clarifying the cause of preventing programming understanding, and developing a new teaching method appropriate for their understanding degree would be necessary. The objective of this paper is to examine the causes preventing novice programming learner's understanding from the programming education of reading source codes. This paper develops a learning support system for reading source codes. Most types of question in the developed system require learners to answer the proper value of a variable after the execution of a source code of C programming language which is automatically generated. The developed system was utilized in a programming lecture for the novice programmer. This paper obtained student responses from a questionnaire and the learning log data after the students had completed one semester of the instruction in programming, and analyzed these data. From the analysis result, it turned out that different evaluation patterns existed depending on the learner's basic programming skill. The analysis result also suggested inappropriate program descriptions.

1. はじめに

昨今，プログラミングは特に重要な技能として社会的にも広く認識されている．大学等高等教育機関では，プログラミング技能は従来から重要な科目として位置付けられてきた．プログラミングの理解を支援するため，講義では様々な工夫が検討され，そして，そこでの成果は多くの学術雑誌で報告され続けている．しかしながら，プログラミングを不得手とする学習者を支援する有効的な教授法は，調査の限りでは十分に確立されていない．その理由のひとつとして，プログラミングは，論理的思考力，言語力，発想力，数学力など，多様な技能が要求される一方で，プログラミングを不得手とする学習者に対し，どのような技能が不足しているのかを把握する手段が十分に用意されていないことにあると考えられる．プログラミング教育を広く普及させるためには，プログラミングを不得手とする学習者の存在を無視することはできない．したがって，プログラミングへの苦手意識を抑制させることが可能な教材の開発や，従来素養がないと判断されてきたプログラミングを不得手とする層を的確に支援する仕組みが必要であると考えられる．

本稿では，読解の手軽さとトレース・デバッグ学習の基礎力向上に対しての有効性 [江木 09]，プログラミングの基本である順次処理の反復学習の有効性 [Okamoto 10] を踏まえた上で，外的構造の意味に頼らない，プログラムソース自身が保持する内的な構造 (データ依存関係) にのみに基づいた読解学習をプログラミング指導に用い，その結果見えてきたこと，また，プログラミング教育の中で改めて考え直さなければならないと思われることを述べる．読解学習課題は任意の規則で生成されたものであり，言語仕様に関する知識と記憶力・計算力のみの技能に直結する学習課題である．よって，先に課題としてあげた，プログラムを不得手とする学習者の特徴を十分に把握できていないという問題の解決に貢献可能な仕組みであると考えている．加えて，プログラムソース自身が保持する内的な構造にのみ依存する学習課題を題材として学習者の技量を相対的に定量化できれば，内的構造の観点においての指導が可能となるため，従来プログラミングの適性が十分でないとみなされてきた学習者層に新たな観点での教授法を提供可能になると考えている．本稿では，読解学習課題を自動生成し提示可能なシステム [Okimoto 16] を実講義で適用し，学習支援を試みた．運用の結果からは，初学者にとってプログラム読解を困難とする記述を明らかにし，プログラミング教材の作成や教授法検討の際に参考になり得る知見を得ることができた．

連絡先: 松本慎平，広島工業大学情報学部知的情報システム学科，〒 731-5193 広島市佐伯区三宅 2-1-1,
E-Mail: s.matsumoto.gk@cc.it-hiroshima.ac.jp

2. 開発システム

開発したシステムは，C 言語の読解学習を対象としたものであり，ソースコードを題材とした問が自動生成され学習者に提示される．生成されるソースコードは外的な目的のもとでの処理を行うものではなく，外的な意味を持たない処理の連続である．ただし，プログラム自身が持つ構造自体には意味を持っている．本研究で最終的に明らかにしたいことは，1. 内的な意味のみに依存した教材の適用でプログラミングの訓練に有効に働くということ，2. ソースコード読解はプログラミング力向上に寄与するということ，3. ソースコード記述文の困難度水準を量的に定義できるようにすること，の 3 点である．本稿でのこれまでの貢献は，3 に該当するものである．

生成されるソースコードは数十行の短いものである．この点は，“1 メソッドは 1 画面に収まる程度とする” という設計論は一般的であり開発現場で採用されることが多いことから，1 ページ内に収まる処理の読解を行うことができるようになれば，大規模なプログラム読解も抵抗なく対応できるようになるのではないかと考えている．加えて，近年のオープンソースソフトウェアを活用しプログラマは中核に据えたソフトウェア開

発手法が各所で積極的に導入されており，コメント文に頼らないプログラム読解技能の必要性は高いこと，プログラミングの時間のほとんどはコードを読む時間 [Boswell 11] と言われていることなどから，プログラム読解は学習課題として十分に有用であると考えられる．

開発システムは Web アプリケーションであり，Chrome など HTML5 に準拠したブラウザでの動作を推奨するものである．システムは Apache 2.4.7 でデータ入出力を行っており，問題提示には JavaScript ライブラリである jquery 1.7.2，問題データ管理には MySQL 5.6.16 を用いている．アプリケーション自体は，php 5.5.9 で開発されている．図 1 はプログラミング学習者が使用するテスト問題の 1 例であり，この場合は多肢選択方式の問題が提示されている．開発システムが持つ問題生成機能では，任意の規則に基づいて問題を生成できる．“変数の数”，“命令文の数”，“演算対象の変数の数”，“利用演算子 (代入演算子)”，“分岐・繰り返し利用の有無”，“条件判定文の記述に関する制約” など設定を事前に行うことで，条件に合ったソースコードを生成できる．なお，学習者の理解の度合いを確認するため，意味のない代入処理や，意味のない条件判定処理の有無を設定できるようになっている．設定情報を読み取った後，ヘッダ部が記述処理が行われ，変数の宣言記述が行われる．その後，設定された命令文数だけ本文生成処理が呼ばれ，C 言語のプログラムが形成される．制限時間を設定可能なテストは，1 問以上の複数の問題で構成される．一度受けたテストを 2 度受験できないようになっているが，一度受けたテストに含まれる問題は，問題一覧機能から何度も練習できるようになっている．システムの詳細は文献 [Okimoto 16] に譲る．

3. 問題形式

開発システムでは，多肢選択方式の他に，記述方式，並び替え方式，空欄補充方式が用意されており，教授設計に基づき，希望の問題形式で提示可能である．各学習形式の説明と期待される学習効果を以下に記述する．

3.1 記述方式

提示されたソースコード内の全命令を実行し終えた時に変数が保持している値 (整数値) を手入力で回答する方式である．ソースの中で用いられていた変数の中から任意の一つを選出するか，あるいは全ての変数の値の回答を学習者に求める．任意の一つを選出する方式の場合，問われる変数は，学習者によって任意に決定される．

3.2 多肢選択方式

提示されたソースコード内の全命令を実行し終えた時の変数セットが保持している値 (整数値) を回答する方式である．各選択肢には，全ての変数の値が示されており，学習者は，全て適切な値を有する選択肢をひとつ解答する．開発システムでは，正解以外の各選択肢の各変数の値は任意に決定され，また，選択肢は利用者ごとに全く異なったものが生成される．

3.3 整序方式

ソースコードを実行し終えた段階の変数セットの値が与えられており，この値と同様の値が得られるように，事前に用意された命令を適切な順番で整列する方式である．仕様上，正解を含む数パターンの整列順が予め選択肢として用意されており，学習者は，適切な命令の整列を選択肢の中からひとつ解答する．正解以外の各選択肢の各変数の値は任意に決定され，また，選択肢は利用者ごとに全く異なったものが生成される．

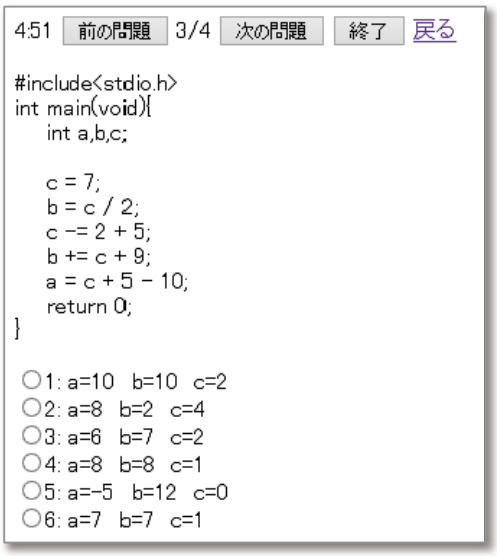

図 1: プログラム読解学習システムの開発システムの動作例

3.4 空欄補充方式

ソースコードを実行し終えた段階の変数セットの値と，任意の 1 行が欠落したソースコードが与えられており，実行後，示された変数セットと同様の値が得られるよう，欠落部に適切な命令を補完する方式である．欠落部は学習者によって任意に決定される．選択式，手入力式両対応であり，手入力回答方式であれば，従来ソースコードを記述される場合と比較して，学習者の入力負荷軽減という目的で有用である．

4. 予備実験及び結果

自動生成されたソースコードの構成に応じたパターンや特徴を明らかにするため，C 言語プログラミングの基本を既に取得している大学 4 年生 10 名の被験者に対して予備実験を行った．言語仕様を完全に理解していれば解ける多肢選択方式による 15 問のテストを 10 分の制限時間で行い，回答ログデータを収集した．プログラミング関連科目の成績を調査したところ，被験者の技能水準は一様であることを確認している．変数は 2-4 個，行数は 3-5 行とし，全て代入文 (複合代入演算とインクリメント，デクリメントを含める) のみで構成されるソースコードを提示した．演算子は四則演算と剰余，演算対象 (右辺) は 1-3 個である．全て単純な仕組みに基づいているが，制限時間内に全て回答するためには滞りなく読解を進めなければならない．実験終了後，項目反応マトリクスを出力した後，4 母数ロジスティックモデル [植野 10] により各問題のパラメータを計算し，各項目のロジスティックモデルのパラメータ及び項目情報量に基づいて結果を分析した．

15 問の項目情報量をまとめたものを図 2 に示す．項目情報量が他と比較して顕著に低い問題が 6 問存在しており，そのうち 5 問には，複合代入演算やインクリメントが共通して記述されていた (図 3 参照)．このことから，これらの要素がプログラミングを得意とする学習であってもプログラム理解効率・可読性を妨げている原因のひとつではないかと仮説を設定した．とりわけ，プログラミングを不得手とする学習者層にとっては，学習を阻害することに強く寄与する不適切な記述であると考えられる．このことを踏まえれば，教授の現場では，これらの記述に対する指導を徹底することや，あるいはコーディン

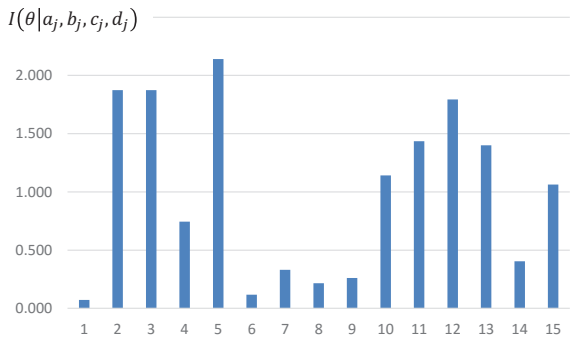

図 2: 問題ごとの項目情報量

int a,b,c,d;
 	
b = 10;
a = 5 * 7;
b += 1 + ++a;
b += 9;
a += 1;
	int a,b,c,d;
	
	d = 3 * 6;
	d += 4;
	d /= 6;
	c = 4;
		int a,b,c,d;
		
		d = 1;
		b = 6;
		d *= b / 8;
		a = b * b;
			int a,b,c,d;
			
			b = 3 * 9;
			b += 1 - 5 * b;
			a = b - 7;
			b += 8;
				int a,b,c,d;
				
				b = 2;
				d = b;
				d += 2 + 9;
				b -= d;

問題(1)	問題(6)	問題(7)	問題(8)	問題(9)

図 3: 項目情報量が顕著に低い問題

グ規約で禁止するなどの対応が必要であると考えられる．

5. 実講義での運用及び結果

情報学を専攻するプログラミング初学者 107 名を対象とした講義内で開発システムを利用し，プログラム読解指導の支援を試みた．本講義は，大学 1 年生のプログラミング未経験者を対象としたものであり，教授されるプログラミング言語は C 言語である．読解学習は，15 回のうち 6 回の講義で行い，各講義時間内の 10 分間を利用しテスト形式で実施された．

問題の読解に必要となる知識量を講義の進行に応じて増加させることで，講義の内容と関連付けた．具体的には，初期の講義の段階では，変数の数は非常に少なく，また，2 項の四則演算のみが出題される．講義の回を重ねるごとに変数の数やプログラム行数が増え，また，剰余演算子や加減算代入，インクリメントやデクリメントなどが問題に追加された．なお，代入文のみの処理の読解を対象とした．

15 回の講義終了後，アンケートを実施し，読解学習の効果を調査した．システムはあくまで読解学習を支援するためだけのものであるので，システム自体の有効性を問うのではなく，教材の中身に対する評価や，読解学習がプログラム学習にどの程度貢献したのかを問う 15 の設問を用意し，6 段階リッカート尺度で学習者から回答を得た．アンケートの各項目は以下に示すとおりである．なお，Q1-6 はプログラミング学習に対する読解教材の貢献を調査するためのもの，Q7-10 はプログラミング技能向上に対する読解学習の期待を調査するためのもの，Q11 は教材の質を調査するためのものである．

- Q1: プログラムの基本的な流れを理解することに寄与した度合い
- Q2: 代入文の仕組みの理解や，代入文の処理の順序の理解を支援することに寄与した度合い
- Q3: 代入文の発展（複合代入演算，インクリメント，デクリメント）の仕組みを知ること，注意を払うことに寄与した度合い

表 1: 各質問ごとの 2 群の平均値

質問	全体 (n=107)	上位 (n=51)	下位 (n=56)
Q1	3.486	3.647	3.339
Q2	3.645	3.784	3.518
Q3	3.439	3.569	3.321
Q4	3.411	3.647	3.196
Q5	3.206	3.373	3.054
Q6	3.673	3.765	3.589
Q7 **	4.075	4.529	3.661
Q8	3.776	3.765	3.786
Q9 **	3.850	4.255	3.482
Q10 *	3.692	3.941	3.464
Q11	3.355	3.333	3.375
Ave.**	3.601	3.783	3.435

- Q4: プログラムに慣れることに寄与した度合い
- Q5: バグ探しや，人のプログラムを適切に読解することに寄与した度合い
- Q6: 読解学習トレーニングシステムが，剰余の演算に慣れることに寄与した度合い
- Q7: 読解学習の「%」の知識習得に対する期待
- Q8: 読解学習の「++」,「−」の知識習得に対する期待
- Q9: 読解学習の「+=」,「-=」の知識習得に対する期待
- Q10: 読解学習の「*=」,「/=」,「%=」の知識習得に対する期待
- Q11: システムがランダムに作り出した命令読解の知識習得に対する期待

アルゴリズムやプログラミングに関係する科目の中で行われたテストや課題の結果から，各被験者のプログラム基礎力を設定した．そして，プログラム基礎力の平均以上 (51 名) と平均以下 (56 名) の 2 群に被験者を分割し，各質問ごとの双方の差を比較した．結果を表 1 に示す．なお，Welch の t 検定の結果，$p < 0.05$ で 2 群に有意差が認められた質問については*を，$p < 0.01$ の場合は**を質問の右側に標記している．

歪度と尖度による正規性の検定の結果，Q1 を除いた全ての項目で正規性が示された ($p < 0.05$)．このことを踏まえ，本稿では被験者の反応を間隔尺度と仮定して，パラメトリックな分析を主として行った．

5.1 全体傾向の分析及び考察

表 1 の全体の平均値から，Q7 と Q9 は他の項目と比較して高い評価を得ていることを確認できる．よって，読解学習は剰余演算子や加減算の複合代入演算の読解技能の向上に強く効果を感じさせていたことが推察される．一方で，Q1，Q3, Q4, Q5, Q11 の結果は比較的低く，特に Q5 の結果は最低であった．Q1，Q4，Q5，Q11 の結果から，大部分の学習者は自動生成された読解教材は基本的なプログラミング知識の修得に向けてある程度の貢献があったと感じていたが，プログラムの基本的な流れを理解すること，プログラミングに慣れること，人のプログラムを読むこと，バグを修正することには効果が弱いと感じていたと考えられる．実際，自動生成されたソースコードは現実離れした記述で構成されたものであったため，評価結果は妥当なものであると考えられる．加えて，本稿で取り扱った読解教材は代入文のみであったことから，制御構文が含まれる実際のソースコードとのかい離により，Q1 の低評価につながったと推察される．これら項目の評価を高めるために

は，現実的なソースコードや，バグ修正を読解教材とする必要があると考えられる．Q11 の “ランダム” に命令を組み立てる要素により代入文の発展に注意を払う必要が生じることから，Q11 の要素は Q3 の結果に強い影響を与えていたと考えられる．自動生成されたソースコードを読解教材とすることは，プログラミングの講義では通常触れることがない難解な仕組みに多々触れることがある．具体的には，現実的なソースコードでは複合代入演算の右辺に計算が来ることはまれであるが，自動生成の場合では大いに起こりうる．このような概念は初学者にとっては極めて高度な仕組みである．ただし，言語仕様の限りにおいては，上述の記述は可能である．よって，複合代入演算など高度な処理を教授するのであれば，多様な技術水準の人に読解されることを学習者に意識させながら教授する必要があると考えられる．あるいは，複合代入演算は読解を困難にする仕組みであると考えられるため，誰にとっても容易な読解を可能とするソースコードを記述すべきとの指針の下では，そもそも複合代入演算は使うべきではないと教授者は指示しなければならないのかもしれない．

5.2 上位・下位群の差に基づく分析及び考察

表 1 平均の行で示される学習者の上位・下位群の差について 2 要因の分散分析を行ったところ，$p < 0.01$ で有意な差があったことが分かった．下位群の評価が上位群より低いことから，本稿での方式により自動生成した読解教材は，プログラミングを不得手とする学習者の支援には十分に貢献できなかったと考えられる．十分なプログラミング技能を有している学習者は，内省するための十分な技能を同時に有していると考えられることから，自らの技能の中で，読解教材の採点結果から十分あるいは不十分な点を把握できる．開発システムはリフレクションの良い機会として機能したことから，上位群の評価は高かったと考えられる．一方で，下位群はその逆であるため，ここでの評価結果は妥当であると考えられる．この考察に基づけば，あらゆる学習者に読解学習システムを受け入れられるようにするためには，採点後，内省を支援できるような，読解のポイントを適切な記述でフィードバックする必要があると考えられる．読解教材の自動生成と共に，フィードバックも自動生成する機能の開発が必要であると考えられる．

表 1 の Q8，Q11 に着目すると，上位群と下位群の差はほとんどない．上位群は下位群より評価が高い傾向があるため，Q8 と Q11 の結果は，全体のバランスから考えると，上位群の評価が例外的に下がっていた結果であると考えられる．プログラミングの発展知識の多くは，効率性の向上というメリットをもたらす半面，理解を難解にさせるというデメリットを併せ持ったものである．Q8 の結果は，インクリメント・デクリメントの知識に対して，上位群であってもデメリット以上のメリットを感じされなかったということを示唆していると考えられる．よって，インクリメント・デクリメントについては，その利点を十分に伝えられない段階では触れるべきではないか，あるいは，内的な構造のみに完結させて教授すべきではなく，外的な構造と対応付けて教授すべき知識であることが示唆される．Q11 の結果については，得られた結果は，読解教材を完全に任意に生成すべきではないことが示唆される．よって，あらかじめ定められた構造の元で問題を生成し，構造の習得を確認するという意図を学習者に伝えたうえで教材を展開すべきであると考えられる．

同様の観点から，Q7，Q9，Q10 で示される有意差は，上位群の高評価によってもたらされたものであると考えられる．以上を踏まえると，Q7，Q9，Q10 については，下位群の満足度を向上させる余地は残されており，その方策として，適切なフィードバックを与えるということが適切な手法ではないかと考えられる．ただし，プログラミング不得意層の支援とは何かについて今一度考えてみた場合，プログラミングに対する苦手意識を少しでも減らし，最低限プログラムに向き合えるようにすることと位置付けるならば，プログラミング不得意層に対して絶望感を与え兼ねない知識には触れるべきではないのかもしれない．プログラミングの高度な仕組みや知識 (発展的な記述法) は，上位群が望んでいる可能性が高く，このことを裏付ける結果は Q7，Q9，Q10 の上位群の高評価及び有意差として表面化したと考えられる．IT エンジニアの慢性的な不足に答えるのならば，発展的な記述は生産性の非効率化やバグ発生の温情となる可能性もあるため，極力使うべきではないかもしれない．C 言語であれば，発展的な記述とは，ポインタや構造体も該当するであろう．一方で，発展的な記述に関する指導を除くならば，上位群の期待には応えられないということを意味する．プログラミングは，熟練の技能と教養的な技能という 2 面性を有した技能であると考えられる．よって，プログラミングを教授する場合，到達目標を示すだけではなく，社会においてどのような状況で用いられるプログラミング技能なのか，という指針も明確に示したうえで，各教材を学習者に提供しなければならないのかもしれない．

6. おわりに

本稿では，外的構造の意味に頼らない，プログラムソース自身が保持する内的な構造 (データ依存関係) にのみに基づいた読解学習をプログラミング指導に用い，その結果明らかになったことを報告した．実験の結果から，初学者にとっては，複合代入演算子がプログラム読解を困難とする記述となっている可能性を確認した．

謝辞

本研究は，独立行政法人日本学術振興会科学研究費助成事業 (若手 (B) 13304922, 基盤研究 (C) 26350296) の助成を受けて実施した成果の一部である．ここに記して謝意を表します．

参考文献

[江木 09] 江木鶴子, 竹内章, プログラミング初心者にトレースを指導するデバッグ支援システムの開発と評価, 日本教育工学会論文誌, Vol.32, No.4, pp.369-381 (2009).

[Okamoto 10] M. Okamoto, K. Terakawa, M. Murakami, K. Ikeda, M. Mori, T. Uehara and H. Kita, Computer Programming Course Materials for Self-Learning Novices, Proc. of World Conference on Educational Multimedia, Hypermedia and Telecommunications, Vol.2010, No.1, pp.2855-2861 (2010).

[Boswell 11] D. Boswell, Trevor Foucher, The Art of Readable Code (Theory in Practice), O'Reilly Media, 2011.

[植野 10] 植野真臣，荘島宏二郎，学習評価の新潮流，朝倉書店 (2010).

[Okimoto 16] K. Okimoto, S. Matsumoto, S. Yamagishi, T. Kashima, A source code reading based learning support system for novice programming education, Proc. of The 21nd International Symposium on Artificial Life and Robotics, PS3, pp.765-768, 2016.