4ZF-02

初心者の不完全プログラムを対象とした理解度推定に関する研究

堀川直希　　太田剛
静岡大学大学院情報学研究科

1. まえがき

初学者が作成したプログラムの多くは何らかの知識の不足や勘違いが原因となり、さまざまの間違いを犯している不完全なプログラムとなる事が多い。そこで、Neil ら [1]は初心者の犯す間違いに注目し、初心者 10 万人、17,144,721 の java ファイルを対象として、間違いの種類や頻度について調査した。

2. 目的

Neil らの調査では間違いをした理由、どのような過程を経て間違えたかについては触れていない。そのため、本研究では間違えた理由、間違いをする過程について調査を行うと共に、初学者が間違いを犯す要因を明らかにすること、学生が何を考えていたのかを推測し、その推測が正しいかどうかを、明らかにすることを目的とする。

3. 調査方法

不完全プログラムの調査は 2011 年度の静岡大学の学生 96 人が 1 年次のプログラミングの授業で課題を実施した際の 10,633 の java ファイル編集の様子を PPV[2]で観察した。調査対象課題は第 5 回の講義の課題であり、プログラムに慣れてきたころを対象としている。課題内容はタートルプログラムを利用して 3 匹の亀を競争させるプログラム、流れ星を書くプログラム、今までに書いた絵を回転させるプログラムの 3 つである。また、今年度の java を用いたプログラミングの講義中である静岡大学の学生 106 名と作成中のプログラムの観察を教室内で TA として行った。そして、不完全プログラムを作成した学生にどのように考え、不完全プログラムの原因となる記述をしたのか聞き取り調査を行った。これらの調査によって不完全プログラムの原因の収集を行った。また、ミスが確認されたプログラムと、そのプログラムがコンパイルされる直前にコンパイルされたプログラム(以下、直前プログラムと呼ぶ)を比較して何が起きているのかを分析した。

4. 調査結果

過去の学生の編集の様子を調査した結果、特徴的なミスが 2 件、while 文の周辺で発見された。これらをセミコロン追加ミス、インクリメント忘れと呼ぶことにする。ミスの例を図 1 に示す。

```
while(K=20);{
  kameTaro.fd(150);
  kameJiro.fd(150);
  kameSaburo.fd(150);
  i++;
}
```

```
while(K=20){
    kameTaro.fd(150);
    kameJiro.fd(150);
    kameSaburo.fd(150);
}
```

図 1：不完全プログラムの原因例
左:セミコロン追加ミス、右:インクリメント忘れ

セミコロン追加ミスは小かっこの直後にセミコロンを記述したことによって無限ループとなる問題であり、17 人の学生、74 ファイルが発見された。セミコロン追加ミスは文献 [1]においても取り上げられているミスである。プログラミングの授業内でセミコロン追加ミスをした学生 2 名に聞き取り調査を行った結果、「println 文でかっこの後にセミコロンを付けるため while 文でも付けた」、「なんとなく付けた」という回答であった。

インクリメント忘れは条件式にかかる変数がインクリメントされないことによって無限ループとなる問題であり、50 人の学生、566 ファイルが発見された。

インクリメント忘れは文献[1]は取り上げていないミスである。また、プログラミングの授業内で学生から原因の聞き取り調査を行うことができなかった。これは、多くの学生たちは無限ループ等の予想外の動作をする場合、質問をせずにプログラム内の変数の値を編集し、挙動を観察する事が多く、自己解決してしまうため、授業内でインクリメント忘れを行った学生を発見できなかったためである。

5. 考察

セミコロン追加ミスをした学生に聞き取り調査を行った結果、println 文やメソッド呼び出しなどの小かっこの後にセミコロンを付ける事と while 文の文法規則を混同してしまい、while 文の小かっこの後にセミコロンを付けたと思われる。また、同テキスト内の println 文やメソッド呼び出しなどを見たことによって、セミコロンを小かっこの後に付けたことが考えられる。セミコロン追加ミスをした学生は if 文でも while 文と同様に小かっこの後にセミコロンを付ける場合があった。セミコロン記述ミスをした学生 17 人の内 6 人が if 文でも while 文と同様に小かっこの後にセミコロンを記述したことが

A Study on estimating the extent of understanding of a novice programmer using incomplete programs
Naoki Horikawa and Tsuyoshi Ohta, Graduate School of Informatics , Shizuoka University

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.

確認された。[1]また、セミコロン追加ミスをした過去の学生 17 人の内、15 人は while 文を書き始める際、小かっこ記述直後にセミコロンを付けていた(図 2)。そのため、15 名の学生には小かっこの後にはセミコロンを付けるという考え方が生まれていた事が考えられる。これらのことから、セミコロン追加ミスはセミコロンの意味を理解していないため生じる問題であり、セミコロンの意味を教えれば解決する問題であると考えられる。また、セミコロン追加ミスは Neil ら[1]の調査によっても 20264 ファイルが発見されており、静岡大学の学生特有のミスでは無い事が解る。

int length;
length = random(100);
	int length;
	length = random(100);
	
	while(length < 200);
	length++;

図 2：時間軸に沿ったセミコロン追加ミスの例
左:直前プログラム、右:コンパイル時にセミコロン追加ミスが確認されたプログラム

インクリメント忘れを行った学生の多くは while 文を書き始める際に、while 文の構造や変数、条件式などに気を取られた結果、インクリメントを記述し忘れたと考えられる。これは、多数のことに気を取られたため、インクリメントの事が考えから抜け落ちた事と思われる。また、著者が学生の観察をした授業では while の条件式を true として意図的な無限ループをさせる例題や課題が多い。そのため、学生はループをさせる場合、while(true)とイメージする事が多く、条件式の考えが理解されにくい。その結果、指定回数ループさせる場合、条件式の更新を意識しないため、無限ループをするプログラムになりやすいと考える(図 3)。そのため、インクリメント忘れ問題は意図的な無限ループをさせる問題の比率を減らし、特定の回数ループさせる問題の比率を増やす事によって、問題の発生率を減らすことができると考えられる。

6. まとめ

本論文では不完全なプログラムの原因となる初学者の考え方を調査した。また、過去の学生のプログラム編集の様子から、コンパイルした不完全なプログラムと直前プログラムを調べる

while(true){
sleep(0.1);
star.fd(5);
star.rt(10);
update();
}
	while(i<=10){
	sleep(0.1);
	star.fd(5);
	star.rt(10);
	update();
	}

図 3：時間軸に沿ったインクリメント忘れの例
左:直前プログラム、右:コンパイル時にインクリメント忘れが確認されたプログラム

ことによって、学生が何を考えていたのかを推測し、その推測が正しいかどうかを、今年度同じ誤りを犯した学生に聞き取り調査することで確かめた。その結果、セミコロン追加ミスとインクリメント忘れの二つの特徴的なミスが発見された。

セミコロン追加ミスは println 文やメソッド呼び出しなどの文法規則が while 文や if 文の文法規則と混同することによって発生する問題であると考えられる。これはセミコロンの意味を理解していないために発生する問題であり、セミコロンの意味を教えることによって発生率を下げる事が出来ると考えられる。

インクリメント忘れは while 文の範囲の指定、条件式の設定や使用する変数の初期化などの多数のルールに気を取られた結果、発生しやすくなる問題であると考えられる。また、静岡大学では意図的な無限ループをさせる例題や課題が多い。そのため、意図的な無限ループをさせる問題の比率を減らし、特定の回数ループさせる問題の比率を増やす事を提案し、インクリメント忘れの発生率がどのように変化するか観察をする必要がある。また、インクリメント忘れは文献[1]では取り上げていないため、指導方法、例題や課題の選び方によって生じる間違いであるのかを調査するため必要がある。

また、今回の調査では 1 つの課題についてのみ調査を行っている。そのため、今後は全体の課題を対象として全体での問題の発生率、while 文や if 文を習った直後の問題の発生率、講義を進めることによって問題の発生率がどのように変化するかを調査する必要がある。

参考文献

[1]Neil C. C. Brown and Amjad Altadmri:“Investigating Novice Programming Mistakes : Educator Beliefs vs Student Data,”Proc. of the 10th annual conf. on International computing education research,pp.43-50(2014).

[2]松澤芳昭,岡田健,酒井三四郎:”Programming Process Visualizer:プログラミングプロセス学習を可能にするプロセス観察ツールの提案”,情報教育シンポジウム(SSS2012),Vol.2012,No.4,pp.257-64(2012).

[1] if 文では間違いを犯したが、while 文で間違いを犯さなかった学生が 4 名確認された。if 文でセミコロン追加ミスを犯した学生 10 名、23 ファイルが発見された

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.