プログラミング学習のソースコードの文に対するフィードバックメッセージの記述に関する考察

2017SE007 有賀千浪
指導教員: 蜂巣吉成

1 はじめに

大学で行われているプログラミング教育では，講義形式で学習後に実習形式で指定されたプログラムを作成し課題に取り組むことで，プログラミングの基礎概念や実用的な力を身につける．学習者が行き詰った際, 教員やティーチングアシスタント（以下，TA）に個別フィードバックを求めるが，多数の学習者に対して少数の教員と TA では個別のフィードバックを十分に行うことが難しい．解法がわからず行き詰まり状態が続くと学習意欲がそがれる原因になる．

既存の自動フィードバックを行うシステムはいくつか存在している．河本 [2] の研究では， コメントを用いてフィードバックメッセージを生成しフィードバックを行う方法が提案されているが， 教員がフィードバックのコメントをどのように記述すれば良いかの指針がない．

本研究では，プログラミング学習のソースコードの文に対するフィードバックメッセージの記述に関して考察する．文を条件分岐や繰り返しなどに分類し，それぞれについて教員がフィードバックメッセージを書く際に何を記述すれば良いのか迷わないように指針を決める．

2 関連研究

フィーバックシステムはいくつか存在するが，その対象はコンパイル可能なほぼ完成したプログラムであり，フィードバック内容はよくある間違いから一般化した修正内容である [3]．学習向けのコンパイルできない編集途中のプログラムに対するフィーバックシステムは少ない．

木鎌 [1] の研究では，フィードバックブロックを作成し，編集途中のフィードバックブロックを学習者の関心の高い箇所として優先的にフィードバックを行う方法を提案している．河本の研究 [2] では，学習者のソースコードと文の説明をコメントで加えた模範解答を正規化後，編集スクリプトを生成する．得られた編集スクリプトに学習者と模範解答のソースコードを加え，修正箇所の行数と行う操作と模範解答にコメントされた文の説明を抽出し，フィードバックメッセージを生成する．自動で生成されるメッセージは，行数を付与し，行う操作を抽象化したもので文の具体的な説明は生成されない．自動で生成されないフィードバックメッセージは教員が記述する．記述方法としては，模範解答の文末に「//」を用いてコメント付与をし，フィードバックメッセージとコメントを区別できるように「@」を用いて記述する．しかし，河本の研究ではフィードバックメッセージの教員の記述部分についての指標がない．

3 学習に適したフィードバックメッセージの考察

3.1 学習に適したフィードバックメッセージの指針

本研究では，河本 [2] の自動フィードバックシステムを利用し，ソースコードの文に対するフィードバックメッセージの記述に関して考察し，教員が記述する際の指標を決める．教員が解答を直接提示するのではなく学習者のヒントとなるフィードバックメッセージにする必要がある．理解度に応じて抽象的な文から具体的な文になるように複数のコメントを書く．コメントが必要な文は文の意味を説明するもので「変数の初期化」「条件分岐」「繰り返し」「式文」「返り値」であり，細かく分類すると「条件分岐」は if 文,「繰り返し」は for 文と while 文,「返り値」は return 文である．抽象度が高いフィードバックメッセージのコメントは書き忘れた文がある場合などを対象に何をする文か提示する必要があり，より具体的なフィードバックメッセージのコメントは間違っている文がある場合や何を書いていいのかわからない場合などを対象に用いる変数や値について提示する．また， 問題に提示されている仮引数に関しては理解度に応じてフィードバックメッセージのコメントに提示する必要がある．それぞれのコメントが必要な文に対して次の例を挙げる．

- if 文の制御文
  1. 何の判定をするか
  2. 何の変数や値を用いて判定するか
  3. 具体的な仮引数などを用いる
- for 文の制御文
  1. 何回繰り返すのか
  2. どういう値から何回繰り返すのか
  3. 具体的な仮引数などを用いる
- 式文 (計算する場合)
  1. 何の計算をする文か
  2. 何の変数や値を用いて計算するか
  3. 具体的な仮引数などを用いる

3.2 学習に適したフィードバックメッセージの例

ソースコード 1 は素数判定を行う関数の模範解答に指針にしたがってコメントを付与したものである．

ソースコード 1 素数判定を行う関数 模範解答例 コメント付き

```
1 int prime(int n)
2 {
3   int i;
```

```
4     int judge;
5     judge = 1; //@素数か判定する変数に初期値を代入 @素数であるときの判定を初期値に代入
6     if(n < 2){ //@判定したい変数が2より小さいか判定 @判定したい変数nが2より小さいか判定
7       judge = 0; //@素数ではない判定 @素数か判定する変数に素数ではない判定
8     }
9     for(i = 2; i < n/2; i++){ //@判定したい変数の半分の数まで繰り返し @判定する変数は2から判定したい数字の半分の数まで繰り返し @判定する変数は2から判定したい数字nの半分の数まで繰り返し
10      if(n % i == 0){ //@判定したい変数が割りきれるか判定 @判定したい変数が判定する数字で割り切れるか判定 @判定したい変数nが判定する数字で割り切れるか判定
11        judge = 0; //@素数ではない判定 @素数か判定する変数に素数ではない判定
12        break; //@割り切れたのでfor文から抜けだしをし
13      }
14    }
15    return judge; //@素数の判定の結果の変数
16  }
```

4 評価

フィードバックメッセージが必要な学習者に対して行われるフィードバックの例を挙げる． 学習者の理解状況は把握しないものとして， 操作の指摘から順に具体的なフィードバックメッセージを挙げる．

ソースコード 2 素数判定を行う関数 解答例 式文挿入前

```
1   int prime(int n)
2   {
3       int i;
4       int judge;
5       judge = 1;
6       if(n < 2){
7           judge = 0;
8       }
9       for(i = 2; i < n/2; i++){
10          if(n % i == 0){
11          }
12      }
13      return judge;
14  }
```

式文の挿入がされずに繰り返しフィードバックが行われると以下の順番にフィードバックメッセージが表示される．

1. 式文に不足がないか確認しましょう
2. 11 行目に素数ではない判定をしましょう
3. 素数か判定する変数に素数ではない判定をしましょう

フィードバックメッセージを基に学習者は 3 のように 11 行目に式文を記述する．次の操作が行えずに再びフィードバックを求めた場合には 12 行目に break 文を挿入するフィードバックが行われる．

ソースコード 3 素数判定を行う関数 解答例 break 文挿入前

```
1   int prime(int n)
2   {
3       int i;
4       int judge;
5       judge = 1;
6       if(n < 2){
7           judge = 0;
8       }
9       for(i = 2; i < n/2; i++){
10          if(n % i == 0){
11              judge = 0;
12          }
13      }
14      return judge;
15  }
```

break 文の挿入がされずに繰り返しフィードバックが行われると以下の順番にフィードバックメッセージが表示される．

1. 式文に不足がないか確認しましょう
2. 11 行目に割り切れたので for 文から抜けだしをしましょう

フィードバックメッセージを基に学習者は 1 のように 11 行目に break 文を記述する．

5 考察

素数判定を行う関数やうるう年の判定を行う関数のように模範解答が複数存在する場合には，差分を取り差分が少ないほうを選択するなどの方法で模範解答の選択をする必要がある．また，教員側が複数の模範解答にフィードバックメッセージのコメントを付与する必要がある．そのため教員の負担が増えることになるが，模範解答の共通部分を再利用することでできるだけ負担を減らす．

6 おわりに

本研究では, 河本の研究 [2] のフィードバック方法のソースコードの文に対するフィードバックメッセージの記述に関して，文の分類を行いそれぞれの文に対してどのような文を書いたらよいのかの指針について考察し，学習者の編集途中のソースコードを想定しフィードバックメッセージを提示した．今後の課題として，実際にプログラミングの演習で用いて学習に適しているか評価することや，複数の模範解答がある際の判定方法への対応が挙げられる．

参考文献

[1] 木鎌汐里:” 学習者の編集途中のソースコードに対する自動フィードバック方法の提案—処理単位に基づいたフィードバックブロックの定義 —”, 南山大学理工学部 2020 年度卒業論文 (2021).

[2] 河本菜々: ” プログラミング学習におけるソースコードの文に対するフィードバックメッセージ生成方法の提案”, 南山大学理工学部 2020 年度卒業論文 (2021).

[3] Singh, R., Gulwani, S. and Solar-Lezama, A.: “Automated Feedback Generation for Introductory Programming Assignments”. ACM SIGPLAN Conference on Programming Language Design and Implementation, pp15-26(2013).