エラーを起こさせる初学者向けプログラミングドリルの提案

角田　和広　　増原　英彦　　叢　悠悠　(東京工業大学)

背景・動機

・エラーメッセージを読まない/読めない初学者が多い

原因：初学者にとってエラーメッセージは

"*not understandable*" である [Bekker ITiCSE-WGR '19]

- メッセージが複雑
- 知識が足りない状態でエラーと遭遇する

影響：エラーメッセージが初学者の

"*barrier to progress*" となっている[Bekker SIGCSE '16]

=> エラーメッセージが妨げでなく助けになるようにしたい

手法	内容	問題点	メッセージ内容の改善	エラー箇所発見の支援	初学者のエラー理解の支援	初学者のエラーへの印象の改善
エラーメッセージの改善	専門用語の簡略化 文章の単純化など [Bekker '19]	処理系の変更が必要 効果がないという 実験結果も存在	◎	○	▲	▲
視覚表示の改良	エラー探知箇所を ハイライト表示 [Marceau SIGCSE '11]	エラーを読まずに ハイライト場所を 変更してしまう	n/a	◎	▲	n/a
エラー修正ドリル	エラーのあるコードを 修正をするドリル [Hachisu MCCSIS'14]	メッセージを読まず に解けてしまう ことがある	n/a	○	○	▲
エラー生成ドリル （本提案）	正しいコードから エラーを発生させる ドリル	エラーの読み取りが 必須なため 難易度が高い	n/a	○	◎	◎

提案：初学者向けエラー生成ドリル

・正しいコードとエラーメッセージを見せ、

その**エラーが発生するようにコードを変更**させる形式

- エラーへの深い考察、自主的な試行錯誤を促進　→　より初学者の理解の助けとなる可能性

イメージ図

以下のエラーメッセージが発生するようにコードを書き換えよ

エラーメッセージ：

```
Found:    Double
Required: Int
```

コード：

```
1 def circle(r:Int):Double ={
2   val pi = 3.14
3   r * r * pi
4 }
```

Answer

Double => Int

以下のエラーメッセージが発生するようにコードを書き換えよ

エラーメッセージ：

```
not found: value Yellow
```

コード：

```
1 def nextColor(a:String):String ={
2   if(a == "Green") "Yellow"
3   else if(a == "Yellow") "Red"
4   else "Green"
5 }
```

Answer

"Yellow" => Yellow

課題

・期待する効果の分類、計測方法

・ドリルの出題形式ごとの考察

- 各出題形式の長所/短所
- 各出題形式の組み合わせ

・問題の質の評価/良い問題の作成方法

・初学者が実施できる難易度の考察

- 複雑なエラーメッセージの出題方法

・解答の採点方法/採点するかどうか