N-022

初学者のためのプログラミング助言支援システムの開発
Development of A Programing Support System for Beginners

大島 康裕 †　　六沢 一昭 ‡
Yasuhiro Ohshima　　Kazuaki Rokusawa

1　はじめに

本稿は、プログラミング初学者を対象としたプログラミング支援システムについて述べたものである。

このシステムは、プログラムの誤りやコーディングの問題点などを助言の形で指摘する。プログラムを書き終わった直後にこのシステムの助言を受けることにより初学者はより早くより良いプログラムを作成することを期待する。

プログラミング演習のアシスタントの経験から以下のような知見を得た。初学者のプログラムは、単純であるが一見しただけでは発見しにくい間違いを含むことが多い。そのため、間違いを修正して正しいプログラムを完成させるのに長い時間がかかってしまう。間違いを発見しにくい理由は、主にコーディングが整っておらず可読性が低いことである。

本稿では、まず初学者のプログラム及びプログラミングの問題点を述べ、支援システムに必要な機能を検討する。そして、開発したシステムを説明し、実行例を示す。

2　初学者のプログラミング

2.1　初学者の誤り/問題点

初学者のプログラムの誤りの多くには共通点がある。実際に初学者が書いたプログラムを以下に示す。

```
1   #include<stdio.h>
2   int main(void)
3   {
4       int a=5,a1=10,a2;
5       a3=a+a1;
6       print("%d\n",a2);
7       return   (0)
8   }
```

図 1: 初学者のプログラム例

このプログラムには以下の誤り/問題点がある。

- 変数名や関数名の綴り (変数 a3, 関数 print)
- 似ている変数名が多く使われている。 (a, a1, a2)
- 全角スペースが使用されている。 (7 行目)
- ; (セミコロン) が抜けている。 (7 行目)

一方、初学者はプログラムの可読性を考慮せずプログラミングを行いがちである。その結果、読みにくいプログラムとなってしまうことが多い。このため上述した記述間違いに気がつきにくく、間違いを修正するのに多くの時間を費やしてしまう。

2.2　初学者のコーディング

初学者のプログラムの可読性が低い主な理由は、コーディングに一貫性が無いことである。具体的には、以下の記述に一貫性がないことが多い。

- スペース
  演算子の前後や , (カンマ) の後のスペースの有無
- 括弧
  制御文における中括弧 {} の有無
- 改行の位置
- 字下げ

初学者のコーディングに一貫性がない理由として以下が考えられる。

- とりあえずプログラムが動作すれば よい とする。
- 読みにくいプログラムでも問題を感じない。
- 読みにくいプログラムを書いたという認識が無い。
- 書いたプログラムをもう一度見直すということが無い。
- 他のプログラムからコピー&ペーストするだけで自分のプログラムを作ってしまう。

2.3　支援に求められるもの

プログラムを書き終わった直後に、支援システムを使用して誤りの検出や低い可読性の指摘を得られれば、上述した問題は解決できるだろうと考える。この支援システムは、プログラムの誤りを分かりやすく指摘し、整っていないコーディングについては明確に注意を促すことが必要であろう。

3　システム

本システムは、プログラミングを学ぶ初学者を対象とする。対象プログラミング言語はC 言語である。このシステムは、プログラムの誤り/問題点と、コーディングの問題点を助言の形で指摘する。

†千葉工業大学 大学院 情報科学研究科 情報科学専攻
‡千葉工業大学 情報科学部 情報工学科

3.1 誤り/問題点指摘

与えられたプログラムを GNU indent を用いて整形し、整形したプログラムを解析する。解析には Perl 処理系による正規表現を用いる。

以下について誤り/問題点を検出し指摘を行う。

- 関数
  printf fprintf など
- 制御構造
  if else for while など
- 変数名や関数名および宣言部
- 計算式 (条件式含む)

以下は実際の誤りの例である。

- ; (セミコロン) がない。
- , (カンマ) を書くべきところに . (ピリオド) が書かれている。
  例: `printf("%d\n".n);`
- ”” (ダブルクォーテーション) が抜けている。
  例: `printf(HelloWorld\n);`
- ”” (ダブルクォーテーション) の数が合わない。
  例: `printf("Hello world"");`
- 変数名や関数名の綴りが間違っている。
- 宣言されていない変数を使っている。
- 関数の引数が間違っている。

3.2 コーディング指摘

[1] が採用している Kernighan & Ritchie スタイルを規範とし、以下を調べ指摘を行う。

- 演算子 (= や + など) の前後のスペース
- , (コンマ) の後のスペース
- for, while の中括弧
- 改行の位置

4 実行例

図 2 に実際にシステムが行う指摘の例を示す。誤り/問題点のある行の次の行に、指摘を示す助言が (赤字で) 表示される。

5 まとめ

演習のアシスタントの経験から初学者のプログラム及びプログラミングの問題点を検討した。そして、これらを解決すべく開発したプログラムの誤り/問題点とコーディングの問題点を助言の形で指摘するシステムについて述べた。

指摘対象プログラム

```
void check(int x, int y)
{
    printf("Hello World %d %d\n", x, y)
}
int main(void)
{
    int a = 2,b = 3,c;
    double max = 1.0;
    c=a+b;
    check(a,max);

    return(0);
}
```

システムによる指摘例

```
 1 void check(int x, int y)
 2 {
 3     printf("Hello World %d %d\n", x, y)
3 行目 ; がありません。
 4 }
 5 int main(void)
 6 {
 7     int a = 2,b = 3,c;
7 行目 , の後にはスペースを入れましょう。
 8     double max = 1.0;
 9     c=a+b;
9 行目 演算子の前後にはスペースを入れましょう。

10     check(c, max);
10 行目 関数"check()"の引数の型は int, int ですが、
呼び出し側の引数の型は int, double です。
2 番目の引数が誤っています。
11
12      return(0);
13  }
```

図 2: 指摘例

参考文献

[1] 柴田 望洋, 新版 明解 C 言語 入門編, ソフトバンククリエイティブ, 2004.

[2] 松本 翔太, 松原 俊一, Martin J. Dürst, “大学のプログラミング教育における可読性の影響の調査,” 5ZF-3, 情報処理学会第 74 回全国大会, 2012.

[3] 石嶋 博行, 山田 優里, 粂谷 歩, 高野 辰之, 宮川 治, “プログラミング教育におけるコーディング手順矯正システムの開発,” 5ZF-7, 情報処理学会第 74 回全国大会, 2012.