プログラミング演習における シンボルの名前付けに対する指導

北 英彦*1・上村 拓磨*1
Email:kita@elec.mie-u.ac.jp

*1：三重大学工学研究科電気電子専攻

◎Key Words プログラミング演習，プログラミング演習システム，名前付け

1. はじめに

現在多くの大学がプログラミングの講義を行っている．しかし，プログラミング初心者に対するプログラミング演習において，プログラミングスタイルについて十分な指導を行う時間がないというのが現状である．著者らは，今までに学習者にプログラミングスタイルの一つである字下げ，空白の入れ方を自己学習させるためのシステムを開発した(1)．結果として，以前と比べ学習者が字下げ，空白の入れ方を適切にできるようになった．

本研究では，前研究では対象外としたプログラミングスタイルの内の一つである適切なシンボル名を学習者に名付けさせるためのシステムを開発した．なお，本機能は著者らの所属する研究室で開発および運用を行っているプログラミング演習システム PROPEL に付け加える．三重大学電気電子工学科でのプログラミング演習は Java を用いたプログラミングの学習を行っているため，変数名の命名規則は Java のものに従うことにする．

2. プログラミングスタイル

プログラミングスタイルとはプログラムを書くときの規則のことであり，特定の規則に沿ってプログラムを書くことで，他の人が読んでも理解しやすいプログラムを作成することが可能である．カーニハン(2)は「プログラムを書くことということは，正しい構文を使いバクを直し，それなりに動くようにすれば済むわけではない」と言っている．一般に，ソフトウェア開発はひとりで行うのではなく複数人でプロジェクトを組んで行うことが多い．そのため，プログラムを書く上で他の人にも読まれるということを意識して書くことが重要である．

プログラミングスタイルとしては，字下げの仕方，空白の入れ方，コメントの付け方，変数名の付け方，などがある．これらの利点を以下に述べる．

- 字下げ：波括弧{}で囲まれた範囲を 1 段字下げすることで，ブロックの範囲を視覚的に分かりやすくする．
- 空白：演算子の前後に空白を入れることにより演算子を視覚的に分かりやすくする．
- コメント：プログラムの動作には関係ないが，他のプログラマに式の条件式などの意味を伝えるために使用する．
- 変数名：適切な変数名をつけることで変数の役割が分かりやすくなる．Java では Camel 形式，Pascal 形式，Snake 形式を使い分けている．表 1 に例を示す．

表 1　Java で用いている名前の書式

パスカル形式	Lecture SquareOfDistance	クラス名
キャメル形式	lecture squareOfDistance	変数名 メソッド名
スネーク形式	LECTURE SQUARE_DISTANCE	定数

3. 演習時の名前付けの現状

2016 年度のプログラミング演習で学習者が作成したプログラムを調べ，変数名が命名規則に則した形で名付けられているかどうかを確認した．結果を表 2 に記す．以前の発表(3)でもこの表は載せたが，より厳密な形でシンボル名の意味を評価する方針となったため以前提示した数より増えている．また同時に Camel 形式かつ誤字をしている変数名もあるため，重複しているものもある．学習者の数は 78 人である．

結果として，やはり約半数が不適切な変数名を付けている．講師が事前に変数名の名付け方について説明を行っているにも関わらず，このような結果になったということは名前の重要性が理解できていないか，または，理解したつもりでも実践できていないことを示している．

表 2　変数名の付け方の現状

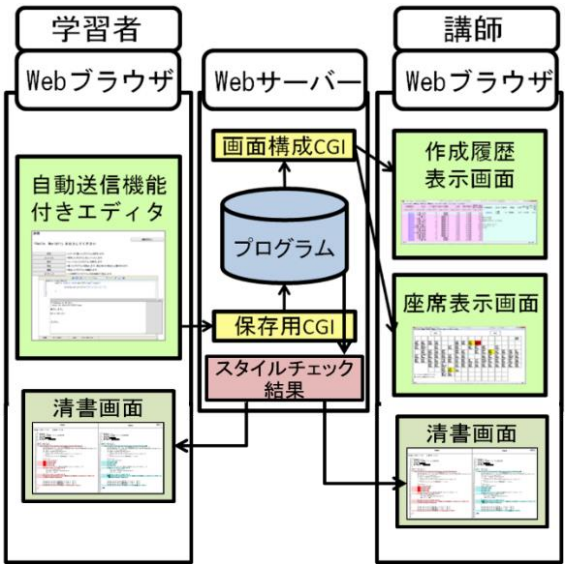
内容		個数	例
変数名の合計		1118 個	
不適切な変数名		613/1118	
	Camel 形式，Pascal 形式でない	474/613	AVERAGE englishpoint
	誤字	44/613	avarage
	意味が不適切	151/613	分散を dispersion
	無意味な名前	30/613	a など一文字

※　Camel 形式でなく，かつ，誤字のものがある

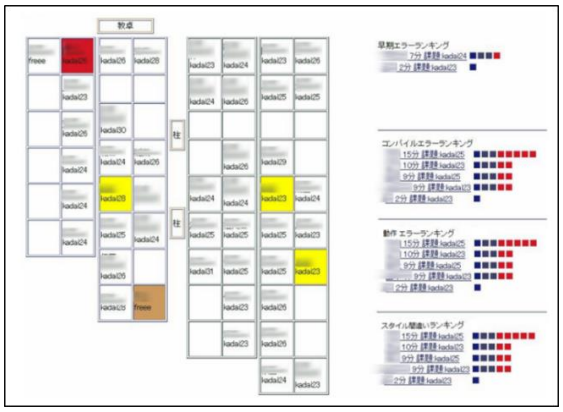
4. プログラミング演習支援システム

著者らの所属する研究室では，学習者のプログラミング状況の把握および学習者への迅速な対処を目的として，プログラミング演習システム PROPEL の開発および運用を行っている．PROPEL のポリシーは，誤字・脱字などの軽微な間違いはできるだけシステムで検出し，その場で学習者に通知し，学習者に考えさせて修正させることである．これにより，講師は動作間違いなどより本質的な間違いの指導に専念することができるようになる．

図 1 (a)にシステム構成図を示す．PROPEL では，演習に必要なプログラムの記述, コンパイル, 実行, 提出，提出したものをシステムが清書したものの閲覧をウェブブラウザ上で行うことができる．エディタには，プログラムをサーバに自動的に送信する機能を埋め込んである．



(a) PROPEL のシステム構成図



(b) 講師用画面

図 1 プログラミング演習システム PROPEL

サーバでは，収集した学習者のプログラムを整理し，図 1 (b)に示すような講師用画面を生成する．この画面では，学習者の座席，取り組んでいる課題などを座席表の形で，コンパイルエラーなどの間違いを修正できずにいる時間をランキング形式で閲覧できる．また，学習者を選択すると，その学習者が作成しているプログラムを学習者の座席まで行くことなくその場で閲覧することができる．講師やティーチングアシスタントは机間巡回中に iPad のようなタブレットを用いて学習者のプログラミング状況を把握し必要があれば指導に行く．

5. システムによる名前付けの指導

表 2 からわかる通り，Java のコーディング規約に従わず，Camel 形式，Pascal 形式を用いていない名前付けが多数されている．これらは書式の確認をするだけなのでシステムで比較的簡単に検出することができる．その他のものは英単語の意味に関係するので単純な方法では検出できないが，本研究ではこれらについても取り組んだ．

5.1 名前の形式違反の指導

表 1 を見てわかる通り Camel 形式と Pascal 形式との違いは先頭のアルファベットが小文字か大文字かである. 学習者に多い間違いとして, 変数名を Pascal 形式で付けていることがある．これについてはシステムで以下の指導を行う．

- 表 1 の内容の説明を載せた Web ページを用意し名前の付け方の規則について学習者がいつでも確認できるようにする．
- 学習者の作成中のプログラムをシステムが定期的に，具体的には 30 秒毎に監視し，プログラムのコード中に変数宣言またはメソッドの定義の開始行があれば，変数名・メソッド名が Camel 形式で書かれているかをチェックする．すなわち, 最初の英単語の語頭が小文字で, 2 番目以降の英単語の語頭が大文字であるかをチェックする．また，大文字で分かち書きされた英単語が英単語リストの中にあるかをチェックする．名前が Camel 形式で書かれていない場合には，名前の付け方の形式に違反していると学習者に通知し，その場で修正させる．

5.2 誤字脱字の指導

学習者の英語の語彙力があまり高くないこともあって，綴り間違いをしているものがある．表 2 のように 1 割近くがこの間違いである. 例えば「average」を「avarage」,「height」を「haight」と間違っているものがある．これについてはシステムで以下の指導を行う．

名前の形式違反のチェックと，同時に，使われている文字列が英語単語リストにあるかどうかをチェックする．リストにない場合には，綴り間違いの可能性を調べるために，英単語リスト中の個々の単語との間で文字列の類似度を表す Levenshtein 距離を計算し，距離が 1 ならば綴り間違いと判断する．Levenshtein 距離が 1 というのは文字列中で 1 文字が違っている，文字列中から 1 文字抜けている，などが相当する．

学習者への通知は，綴り間違いと判断した場合には，学習者の入力した文字列と綴り間違いだと判断した英単語リストの中の単語を併記して提示する．英単語リストの中に類似した単語がない場合には，辞書中にそのような単語がないことを注意する．

5.3 意味が適切でない名前の指導

インターネットで公開されている翻訳ツールや辞書を使って，名前付けをする学習者が多い．検索の一番初めにでてきた単語をよく調べずに使用するのでこのような間違いが起こると考えられる．例えば「分散」と訳す英単語には「variance」「dispersion」などがあるが課題の内容によって使い分けなくてはならない．

これに関しては，課題の文章から形態素解析エンジン MeCab[4]を用いて名詞を抽出し，さらに，課題の文章の表現の特徴から変数名になりそうなものを絞り込む。その後，和英辞書を引いて，変数名になりそうな名詞，および，その英語表現（「分散」のように複数ある場合にはそれらのリスト）を講師に提示し，妥当な英語表現を講師に選択してもらい，それを用いて学習者の付けた名前をチェックする。実際の画面を図 2 に示す。この画面は講師が評価した後の画面である。

図 2　講師の英単語選択画面

デフォルトではすべての単語のチェックが「×」となっている。妥当な英語表現であった場合は「○」、明らかに妥当ではない英語表現であった場合は「×」にチェックしてもらう。明確に判断できない場合はその単語には「△」にチェックしてもらう。これらの評価を用いて学習者の付けた名前をチェックする。

また課題の文章から抽出された単語の他に講師が登録したい単語が存在する事も考えられるため、抽出された単語とは別に講師が思いついた単語も登録できる。

5.4 無意味な名前の指導

課題の内容と関係なく無意味な名前を付ける学習者がいる。特に，それらは一文字で付けていることが多い。

これに関しては，一文字の変数名でもよい課題のときは講師にそれを指示してもらい，それ以外の課題のときは不適切な名前としてチェックする。

6. 実装

システム構成図を図 3 に示す。シンボル名への指導にあたっての 4 つの課題を解決するための基本的なシステムの流れは図の通りである。まずシステムは課題の文書から品詞分解を行い、名詞を取り出す。そして和英辞書（和英 DB）を引き、名詞を英語に変換する。変換した単語を講師に提供する英単語リストとしてまとめ講師に提示する。講師は提供された英単語リスト（図 2）に対して評価を行う。講師が評価した結果を単語、課題文章 DB に保存する。これまでがプログラミング演習の講義を行う前に行う処理の流れである。講義中に学習者がシンボル名の名前付けを行うとシステムはまずそのシンボル名が名前の形式違反や一文字のみのシンボル名ではないかを判断する。その後学習者が名付けたシンボル名に対し誤字脱字や意味が適切であるかを単語、課題文章 DB と比較し評価を行う。その結果を学習者に返す。このような流れで、システムは学習者に指導を行う。

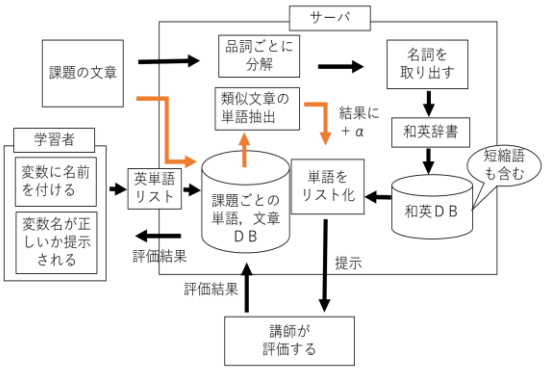


図 3　変数名指導モジュールのシステム構成図

上記の一連の流れの他にシステムは課題の文章を登録する際に類似課題がないかを検索している。これは意味が適切でない名前への指導に対して課題文章のみではすべての適切なシンボル名の把握できない場合があるためである。類似課題が発見されると単語、文章 DB から類似課題で使用された単語と評価結果情報を呼び出し、講師が評価を行う英単語リストに追加する。この流れにより適切なシンボル名として提供される英単語の比率を上昇させる構成になっている。また、類似課題で使用された単語の評

価結果を返すことで講師の負担を増大させない仕組みともなっている。学習者に対しては図 4 のように指導を行う.

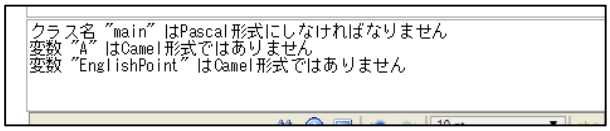


図 4　シンボル名の指摘

7. 運用結果

運用結果としてシステムが行ったシンボル名への指摘回数と内訳について表 6 に示す。また、表 7 にシステムが行ったシンボル名への指摘のうち正しい指摘の数を示す。表 8 に正しい指摘のうち学習者が直せた数を示す。

表 7 からシンボル名への指摘のうち約 64%が正しい指摘だと分かった。正しい指摘が 6 割程度に留まってしまった理由として 2 つ考えられる。

1 つ目が課題ごとに登録されている英単語数が少なく、また、課題のデータベースが非常に少ないということがあげられる。課題ごとに対する単語のデータベースのデータ量を拡張することにより、正しい指摘の精度が上がると考えられる。

2 つ目は「sum」や「num」といった短い単語の誤字脱字判定をする際に Levenshtein 距離の距離判定が 1 となってしまい、これらに対して誤字脱字判定を下してしまっていることがあげられる。対策として短い単語は誤字脱字判定を通さないようにするか、または一度は学習者に提示し、学習者が確認し間違っていなかった場合はチェックなどをしてもらうことによって今後は表示しないなどといったシステムの改善があげられる。

表 3　システムが行ったシンボル名への指摘回数と内訳

内容		回数
指摘回数		481
	誤っている可能性がある	224/481
	綴り間違えがある	114/481
	適切でないシンボル名である（英語の意味が明らかに違うなど）	128/481
	x、 y、 z といった一文字の名前	15/481

表 4　指摘回数の内の正確な指摘回数

内容	回数
正しい指摘	303/481
正しい指摘とは言えない	178/481

表 5　学習者が直せた数

内容	回数
直せた数	215/303
直せなかった数	91/303

表 6　名前の形式違反の指導

内容		システム実装後のシンボル名の個数	システム実装前のシンボル名の個数
シンボル名の合計		867	1118
不適切なシンボル名		91/867	613/1118
	Camel 形式でない	5/91	474/613
	誤字	1/91	44/613
	意味が不適切	87/91	151/613
	無意味な名前	0/91	30/613

8. まとめ

プログラミング演習において，従来は名前付けに関して十分な指導を行うことができず，学習者が書くプログラム中の変数名の約半数が命名規則に則したものではないという問題があった．これらの問題に対処するため本研究では，講師の負担を増やすことなく，システムによって名前付けの指導を行い，学習者自身で適切にシンボル名の名前付けができるようにさせることを目的に行った．このシステムにより学習者は適切に名前付けできる比率が高くなり，一定の効果が得られることがわかった．今後の課題として指摘の精度を高める必要がある.

参考文献

(1) 上村拓磨，北英彦 : プログラミングスタイル習得のための自己学習環境，コンピュータ利用教育議会，2016PC カンファレンス（2016）
(2) Brain W.Kernighan，Rob Pike，訳 : 福崎俊博，プログラミング作法，17-51，株式会社アスキー，2004
(3) 上村拓磨，北英彦 : プログラミング演習におけるプログラム作成時の名前付けに対するシステムによる指導，コンピュータ利用教育議会，2017PC カンファレンス（2017）
(4) MeCab Yet Another Part-of-Speech and Morphological Analyzer: http;//taku910.github.io/mecab/（2017 年 6 月）