入力値の検証コードの実装を促すプログラミング学習支援システムの提案

A Proposal for a Programming Learning Support System Encouraging Input Validation Code Implementation

岩元響希[*1]，水谷 晃三[*1]
Hibiki, IWAMOTO [*1], Kozo MIZUTANI [*1]
[*1] 帝京大学理工学部情報電子工学科
[*1]Department of Information and Electronic Engineering, School of Science and Engineering, Teikyo University.
Email: 203205fe@stu.teikyo-u.ac.jp, mizutani@ics.teikyo-u.ac.jp

あらまし：プログラミング教育では，与えられたプログラムの要件だけではなく，そこから想定される他の要件についても検討，実装できる力の育成も求められる．特に，プログラムへの不正な入力を想定して実装できる力はプログラムの実用性やセキュリティ性を高める能力として不可欠である．そこで本研究では，入力値の検証コードの実装を促す学習支援システムの実現方式を検討した．本稿ではその実現方式と試作したシステムの概要を述べる．

キーワード：プログラミング，学習支援システム，自動評価，入力値バリデーション

1. はじめに

プログラミング能力の育成においては，プログラミング言語の構文やアルゴリズムとデータ構造の理解に加え，与えられたプログラム要件から想定される他の要件についても検討し，実用的なプログラムを作成できる力を身に付けることが求められる．特に，プログラムへ入力される値を複数の観点から想定でき，不正な入力に対処するコードを実装できる能力は，プログラムの異常終了を招く可能性を減らして実用性を高めるだけでなく，欠陥を悪用した攻撃を未然に防ぐなど耐セキュリティ性を高めるうえでプログラマに求められるスキルの一つである．その能力を育成するためには，学習者への意識づけと，学習者への適切なフィードバックが効果的であると考えられる．そこで本研究では，学習者が作成したプログラムの入力値の検証コードの実装の有無を自動的に評価して，その結果をフィードバックして実装を促すためのプログラミング学習支援システムの実現方法を検討する．

2. 方法

プログラミング教育に関する研究においては，学習者が作成したプログラムを自動的に評価して，その結果をフィードバックすることで教育効果の向上を図る試みが行われている[1,2]．これらの研究では，学習者の提出したプログラムがプログラム要件を満たしていることを主な評価基準としている．一方，本研究で実装を促す入力値の検証コードは，プログラム要件としては明示されないが暗黙的に期待される要件（以下，暗黙的要件と呼ぶ）である．その実装を促したり，評価したりする方法が必要である．

ソースコードの記述を解析して評価する方法[3]も検討されている．しかし，本研究のように，暗黙的要件を自ら検討，実装することを求める場合では，学習者によって実装されるコードは千差万別である．そのため，あらかじめパターンを想定しておいてこれを基準に評価する方法や，模範解答のコードと比較して評価する方法では適切に扱うことができない可能性が高い．

そこで本研究では，暗黙的要件を入力値の検証コードの実装に限定したうえで，以下の方法を試みることとした．

1) 学習者に与える課題は，プログラムの要件を 2 段階で提示する方針とする．1 段階目では，プログラムの基本要件のみを提示する．その後の学習状況に応じ，2 段階目として，学習者が自ら検討，実装することが望まれる暗黙的要件を併せて提示する
2) 教授者は，1 段階目および 2 段階目の模範解答となるプログラムのソースコードをあらかじめ用意する
3) システムは，教授者が用意したソースコードを解析してテストケースを自動生成する
4) システムは，学習者の作成したプログラムのソースコードをコンパイルして，3)で生成したテストケースを実行してテストを行う．この結果をフィードバックすることで，入力値の検証コードの実装を促す

3. システムの試作

3.1 システムの基本要件

2.で述べた方法を具体的に実現するための学習支援システムを試作した．対象となる言語は Java とし，Java によるオブジェクト指向プログラミングを一通り学んだ学生を学習者として想定する．システムは Web アプリケーションとして実装する．

3.2 テストケースの生成方法の検討

システム化において課題になるのは，前述の 3) の処理である．本研究では以下の 3 つの方法を検討した．

A) テストケースをアノテーションとして埋込む：模範解答となるコード内に，要件を満たしているかどうかを判定するためのテストケースをアノテーションとして埋込んでおき，システムがこれを解析してテストケースを生成する

B) 大規模言語モデル（LLM）を用いる：LLM を活用して課題文と模範解答となるソースコードからテストケースを生成する

C) 外部ツールにより自動生成する：テストケースを自動生成するツールを用いて模範解答となるソースコードからテストケースを生成する

A) は，ソースコードとは別にテストケースを考えなければならず教授者の負担が増える．B) は検討の結果，本研究の目的に合うテストケースが得られないことがあった．C) については，複数のツールを検証したところ EvoSuite[4]が良好な結果となった．そのため試作システムではこれを用いることとした．

3.3 システムの実装

前述の検討に基づきシステムを実装した．図 1 は教授者がシステムに課題を追加するときの処理フローである．課題追加画面上でソースコードのファイルを指定して追加ボタンを押下すると，システム内部でプログラムをコンパイルするとともに EvoSuite を用いてテストケースを生成する．これらのファイルや生成結果をデータベースなどに登録する．

図 2 は学習者による課題提出時のフローと実際の画面例である．課題提出画面においてプログラムのソースコードを指定してファイルを提出すると，システム内部でコンパイルを行い，あらかじめ生成されているテストケースによりプログラムを実行する．その結果として，テスト成功数を表示して検討すべき暗黙的要件があることを示すことで実装を促す．

4. 評価および考察

試作システムの動作確認と，実際に使用した学習者が暗黙的要件を意識しながら実装を行えるかどうかを確認するための実験を行った．被験者は帝京大学理工学部情報電子工学科 3 年生と同学理工学研究科修士 1 年生の計 6 名である．被験者は本システムの基本的操作を体験しながら提示された課題プログラムの実装に取り組み，事後アンケートに回答する．取り組む課題には，その暗黙的要件として，不正な入力値が与えられた場合に対処することが想定されるものを用意した．

結果，被験者 6 名全員が 2 種類以上の課題に対して，合計 4 回以上の提出を行った． 4 名が 1 種類の課題で全てのテストに成功し，2 名がどの課題でも全てのテストに成功できなかった．アンケートでは，システムの動作や課題の難易度に関する問いのほか，

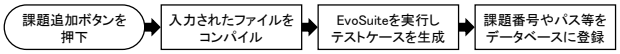

図 1 課題追加機能の処理フロー

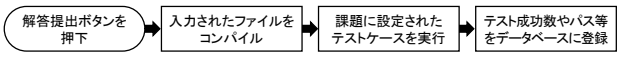

図 2 解答提出機能の処理フロー

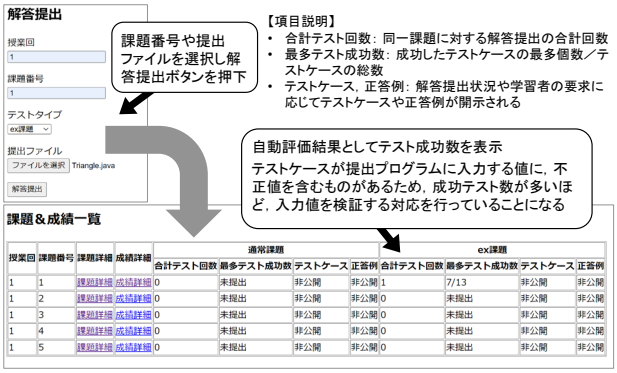

図 3 解答提出機能の動作例

「課題に明記されていない不正な入力値を想定することができたか」という質問を行い，5 名が「できた」，1 名が「できなかった」と回答した．また，「システムの利用を通し，不正な入力値に対処する実装をする力が身に付いたか」という質問に対し，6 名が「身に付いた」と回答した．このことから，本手法がこの能力の育成に寄与したものと考えられる．

一方， EvoSuite によるテストケースの生成は万能ではなく，今回の検証では良好なテストケースが得られた課題にて評価を行った．また，学習者へのフィードバックは提出したプログラムのテスト成功数のみを表示しており，学習効果を得るためにはより丁寧なフィードバックが必要になると考えられる．これらの解決が今後の課題となる．

5. おわりに

本稿では，学習者が作成したプログラムの入力値の検証コードの実装の有無を自動的に評価・フィードバックすることで，この実装を促すためのプログラミング学習支援システムの実現方法を検討した．システムの試作を通じて，本手法がこの目的に寄与する可能性を述べた．

参考文献

(1) 北谷宏紀，井上潮: “Java プログラミング課題のオンライン自動採点システム”，第 6 回データ工学と情報マネジメントに関するフォーラム，F1-1（2014）

(2) 関勝寿：“プログラミングの授業における課題の自動採点”，東洋大学紀要 自然科学篇，Vol.65，pp.31-40（2021）

(3) 西壇諒，高野辰之，小濱隆司，宮川治: “プログラミング演習課題の自動評価システムの研究・開発”，Vol.2019-CE-148，No.8，pp.1-8 （2019）

(4) EvoSuite，https://www.evosuite.org，2024/1/30 参照．