プログラミング能力向上を目的とした プログラムテストの学習環境に関する研究

高桑稔*1・西口大亮*1・北英彦*1
Email: takakuwa@ce.elec.mie-u.ac.jp

*1：三重大学大学院 工学研究科 電気電子工学専攻

◎Key Words プログラミング教育，プログラムテスト，自己学習，演習システム

1. はじめに

教育現場のプログラミング演習において，講師は学習者に対してプログラムを作成する課題を与えて，学習者が実施する．学習者のレベルとして大きく分けると次の 3 つの段階に分けることができる(1)．

① 自力でプログラムを作成できる学習者
② 教科書の例を参考にすればプログラム作成をできる学習者
③ 講師による指導を必要としている学習者

これらのレベルの学習者のうち，講師は③の学習者を中心とした指導をしなければならず，①と②の学習者への十分な指導が行なわれていない．そのため，①や②の学習者のような，ある一定以上のレベルの学習者のプログラミング能力向上がうまくできていないという問題点がある．

そこで解決策としてプログラムの作成が終わっている学習者のプログラミング能力向上を目的とし，プログラミング技術を向上させることができるサポートを行うことができるようにする．

本研究では講師の作業を増やさずに，現状にて指導できていない学習者のプログラミング能力向上を促すために，プログラムテストについて学習することができる環境を提供することを提案する．

2. プログラミング能力向上に必要な要件

プログラミング能力向上に必要な要件の一つとして，品質のよいプログラムを書くことができるかどうかという点がある．プログラムの品質(2)を上げるために必要となってくる要件としては以下のものがある．

- 機能性：求められている機能を実装しているか
- 信頼性：機能が正常に動作するか
- 使用性：分かりやすいか，使いやすいか
- 効率性：目的達成のために使用する時間やリソースを効率よく使用しているか

ここでプログラムテストについては，専門書が多数出版されている(3-6)ことからも分かるように，プログラムの信頼性を高めるための重要な工程である．しかしながら，多くのプログラミング言語の入門書にはプログラムテストの項目がないため，多くのプログラミング初学者は正しいプログラムテストを実施することができていない．そのため，学習者自身が完成したと考えているプログラムであっても実際には求められた動作をしない場合がある．

3. プログラミング演習支援システム

筆者らは現在，プログラミング演習システム PROPEL（PROgramming Practice Easy for Learners）を開発している(7)．PROEPEL には図 1 のように，学習者のシステムと，講師用のシステムの 2 つがある．学習者は Web ブラウザ上のエディタでプログラムの記述からコンパイル，実行を行うことが出来る．それらの内容は 30 秒ごとに Web サーバに送信され保存される．そして，講師側はリアルタイムで学習者のプログラムの進行状況を知ることができる．

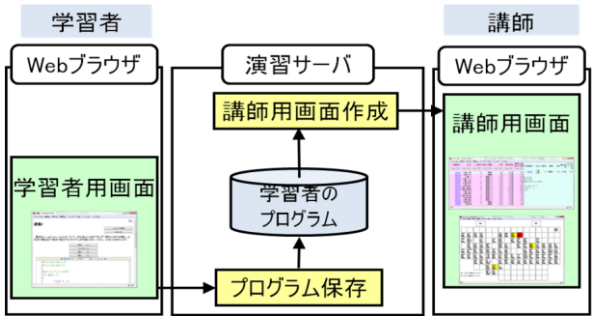

図 1 PROPEL のシステム構成

4. プログラムのテスト

従来のプログラミング演習では学習することができていないプログラムのテストについて説明する．プログラムのテストには，求められた内容からテストデータを作り機能をテストするブラックボックステストとプログラムの構造からテストデータを作りプログラムの構造を確認するホワイトボックステストがある．

プログラミング演習の授業において出題される問題においては，要求仕様が厳密には示されていないため，テストデータを用いるブラックボックステストは適さない．一方でホワイトボックステストは，学習者自身がコードを記述するため，自己学習に向いているテスト技法といえる．よって今回はホワイトボックステストを実施することができる学習環境

を提供することを目的とする．

ホワイトボックステストはプログラムのソースコードに基づいて，構造や制御，データの流れに着目して行うテストであり，いくつかのテスト網羅基準がある．命令網羅テスト，分岐網羅テスト，条件網羅テストの順に厳しいテストとなる．テストとしては網羅率を確認し，プログラムの命令をどの程度網羅的に実行したかを確認する．網羅率は 100%にすることが望ましいが，網羅基準が厳しい場合には網羅率を 100%にすることが難しい場合も存在する．以下にそれぞれの網羅基準について記述する[8]．

① 命令網羅テスト（C0）

少なくとも一回はすべての命令を通過する網羅基準である．網羅テストとしては最も簡素なもので，プログラムテストとしては不十分である．

② 分岐網羅テスト（C1）

分岐網羅テストとは，少なくとも一回はすべての分岐を通貨するという網羅基準の，ホワイトボックステストである．二分岐は真と偽，多分岐はすべての分岐，ループ構造は本体を通過するという条件でテストを行う．

分岐網羅テストの例を図 2 のような構造をもったプログラムで示す．図 2 のプログラムはそれぞれ入力された x と y の値に対して，それぞれの値が 0 であるかどうかを判定するプログラムである．分岐網羅テストでは，少なくとも一回はすべての分岐を通過する必要があるので，x を判定する分岐，y を判定する分岐，それぞれ真と偽の分岐を通過することで，全ての分岐を通過したこととなり網羅率が 100%となる．

具体的に分岐網羅テストを行う例としては，図 3 のようにテスト 1 として x=0，y=0 を入力した場合では，それぞれの分岐について真の分岐を通過する．図 4 のようにテスト 2 として x=2，y=3 を入力した場合では，それぞれの分岐について偽の分岐を通過する．以上のテストにより，全ての分岐を通過したため網羅率は 100%となる．

③ 条件網羅テスト（C2）

少なくとも一回は，複合条件の個々の条件で，すべての可能な結果が得られるという網羅基準である．プログラムの内容によっては，莫大なテストの回数が必要となる，また 100%にするのも困難となるため，今回のプログラムテスト学習としては不適と言える．

また条件網羅よりも厳しい網羅基準があるが，これも条件網羅と同様に，莫大なテストの回数が必要となるため，プログラムテストの学習としては適さない．

5. 提案

本研究では，第 1 章で述べた問題を改善するために，プログラムの作成が自力でできる学習者を対象とし，学習者自身が作成した用意された課題のプログラムに対してプログラムテストを行うことができるような環境を用意し，プログラムテストについて学習者自身が学習できる開発を行う．今回，提案するプログラムテストの内容としては，間違いがなく，意図した通りに動作しているかを学習者自身に確認させることを目的としてホワイトボックステストの分岐網羅テストを行えるようにする．

本研究では PROPEL 上で学習者が提出する前にプログラムテストを行える環境を組み込むことを提案する．

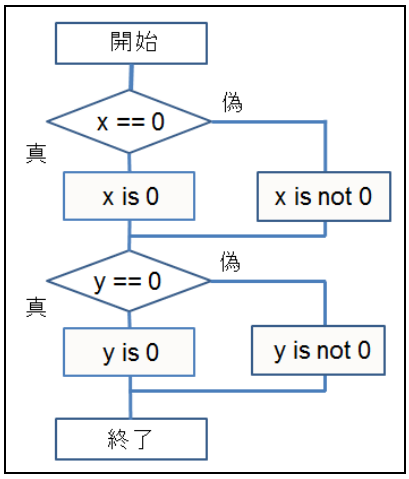

図 2 分岐網羅テストの例

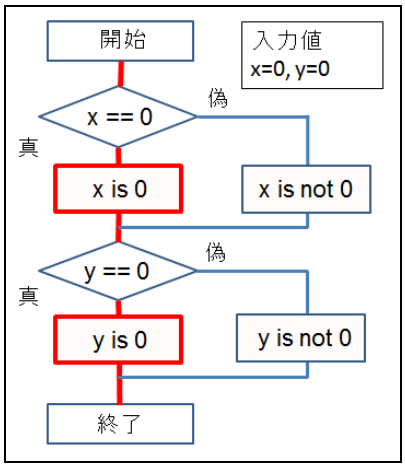

図 3 x=0, y=0 を入力した場合

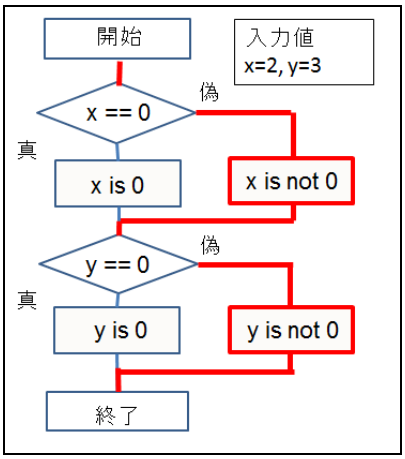

図 4 x=2, y=3 と入力した場合

5.1 学習方法

まず学習者が分岐網羅テストを学ぶための手法として，分岐網羅テストを自己学習することができる学習環境を用意する．実際に学習環境において分岐網羅テストを行うことで，学習者は分岐網羅テストを理解することができる．

5.2 プログラムテストの方法

分岐網羅テストを行うために学習者が作成したプログラムに動作追跡用のコードを埋め込み，実行した際にどこの分岐を通っているかを確認できるようにする．

プログラムを実行する際に分岐が発生する条件としては if 文や for 文などが存在する．if 文の場合，条件式を評価することによりプログラムが 2 つ以上に分岐をする．for 文の場合，制御式を評価した値が 0 でない限りループ本体が繰り返し実行される．このような条件分岐とループによる繰り返しの処理を通っているかを確認する．

5.3 プログラムのフローチャートの自動生成

第 4.2 章で述べたように，分岐網羅テストはプログラム中の全ての分岐を通過することで，網羅率が 100%となる．そこでプログラムをフローチャート化し，図 5 のように通過していない分岐を色を付け，線を太く表示することで，学習者は通過していない分岐を視覚的に一目で分かるようになる．

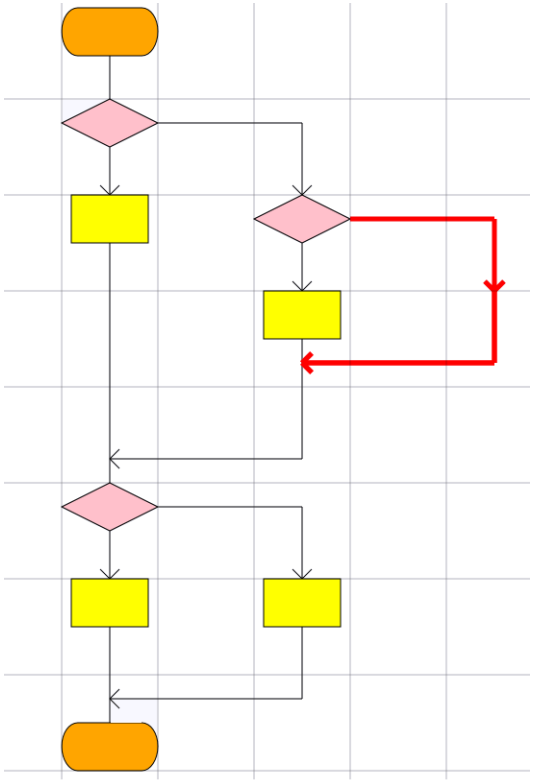

図 5 通過していない分岐を示すフローチャート

5.4 プログラムテストの学習の流れ

実際に学習者にプログラムテストを実施してもらう際の流れについて説明する．

① 学習者にプログラムテスト（分岐網羅テスト）についての説明を提供する．

② 学習者は自身の作成したプログラムのテストを行うために，プログラムを見ながら，どのような入力値を入力すればすべての分岐を網羅することが出来るのか考え，図 6 のように，入力すべき変数名と入力値，またそれらによって得られることが予想される出力を入力する．

③ この時すべての分岐を網羅するために複数回の入力が必要である場合は複数回入力する．

④ すべての入力に対して実行が終了すると，分岐網羅テストの達成率及びフローチャートを図 7 のような形で学習者にフィードバックする．図 7 の場合では，12 行目の if 文において偽の分岐方向をテストしていないためコメントとして，またフローチャート上で警告を出している．

⑤ その際の達成率が 100%になるまでテストを繰り返す．

⑥ 達成率が 100%になれば，図 8 の画面で，学習者自らが作成したテストケースと実際の実行結果の出力は一致しているか，詳細に見ることができる．

プログラムテストの流れを図 9 に示す．

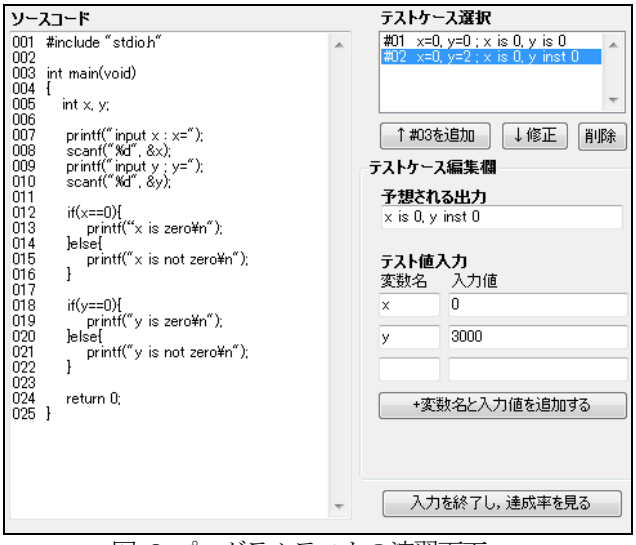

図 6 プログラムテストの演習画面

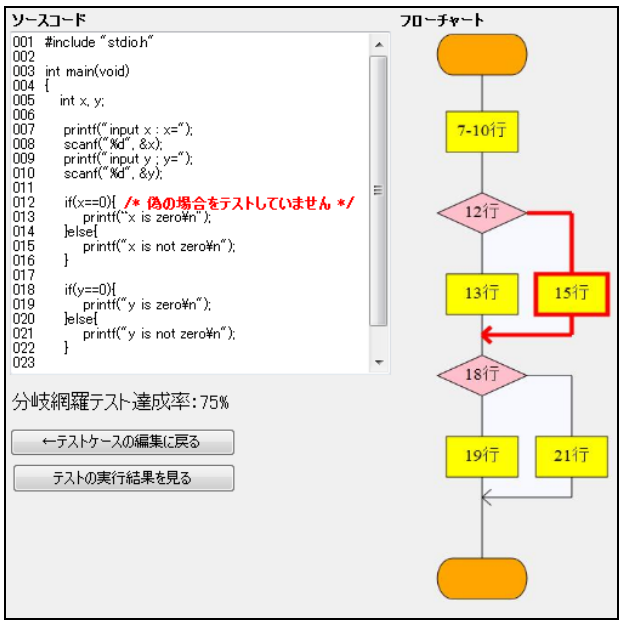

図 7 テスト結果のフィードバック

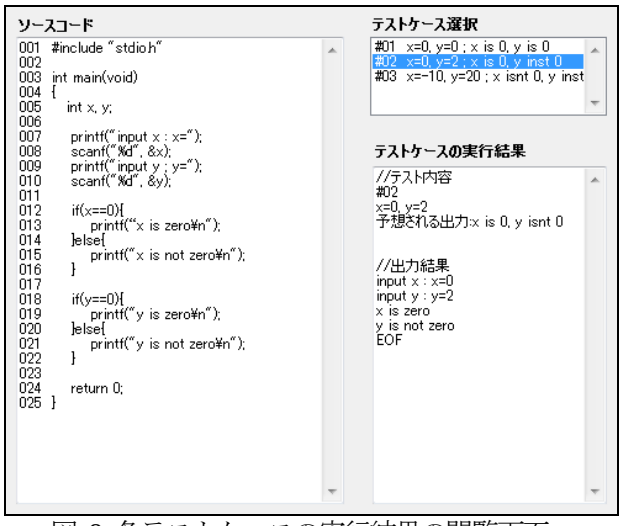

図 8 各テストケースの実行結果の閲覧画面

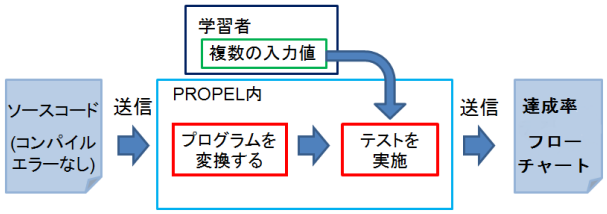

図 9 プログラムテストの流れ

5.5 期待できる効果

今回提案した自己学習環境により，学習者は分岐網羅テストの実施方法を学習することが可能となり，学習者のプログラミング能力の向上が期待される．

6. 今後の課題

今回提案した手法について，実装したシステムを実際のプログラミング演習にて．学習者に使用してもらい，評価する予定である．

また今回のシステムでは，分岐網羅テストのみしか学習することができていない．前述した通り，テスト技法には様々なものがあるので，分岐網羅テスト以外のテスト技法も学習することができる環境を提供することが望まれる．

7. まとめ

プログラム演習の講義において，講師はプログラム作成ができていない学習者に対して中心に指導を行なっている．そのため，プログラム作成ができている学習者に対してはプログラミング能力向上のための指導ができていないという現状がある．

そこでプログラミング能力向上の手段の一つとして，プログラムの信頼性を高めるために必要なプログラムテストを学習することができる環境を提供することを提案した．

学習者は自身が作成したプログラムについて，複数回のテストを実施し，システムにより与えられる分岐網羅の達成率及びフローチャートを元に網羅率を 100%となるようにテストを行う．

本研究により，プログラミング演習において指導が行き渡っていない学習者に対してプログラムの信頼性を高めるという観点においてさらなるプログラミング能力向上の効果を得られることができると考えている．

参考文献

(1) 西口大亮：習熟度に応じたプログラミング演習の支援に関する研究，平成 24 年度修士論文（2012）
(2) 日本規格協会， 2012 年版 JIS ハンドブック 66-1 ソフトウェア (2012)
(3) Boris Beizer : Software Testing Techniques（2003）
(4) Cem Kaner, Jack Falk, Hung Quoc Nguyen，テスト技術者交流会訳：基本から学ぶソフトウェアテスト（2001）
(5) 高橋寿一：知識ゼロから学ぶソフトウェアテスト（2005）
(6) Glenford J. Myers : The Art of Software Testing, pp.43-46（1999）
(7) 伊冨昌幸，小島佑介，高橋功欣，北英彦：プログラムの作成状況を把握する機能を持つプログラミング演習システム，2010PC カンファレンス（2010）
(8) Lee Copeland，宗雅彦訳：はじめて学ぶソフトウェアのテスト技法（2005）
(9) 日本規格協会，JIS X 0121 情緒利用流れ図・プログラム網図・システム資源図記号（2011）