

プログラム動作の可視化による プログラムの動作確認支援

杉本 優太*1・伊藤 福晃*1・北 英彦*1・高瀬 治彦*1
Email: 419m225@m.mie-u.ac.jp

*1: 三重大学大学院工学研究科

◎Key Words プログラミング, 可視化, デバッガ, トレーサ

1. はじめに

プログラミング学習は近年、情報化社会が発展を遂げる中で更なる注目を集めている。プログラミング初学者がつまづく点のひとつとして、学習者が作成したプログラムの動作が誤っている場合に修正できないという点がある。その原因として、プログラムが複雑になると、自分で作ったプログラムの動作が誤っている原因となる場所を特定できないことが考えられる。プログラムの動作が間違っている場所を特定する方法として、一般にデバッガが用いられる。しかし、既存のデバッガは初学者向けに作られていないことが多く、初学者にとって利用法習得の負荷が大きいという問題がある⁽¹⁾。プログラムの理解を助けるための手法として、様々なプログラム可視化システムが開発されており、可視化システムにはプログラミング学習に効果があることが知られている^(2,3)。

本研究では、プログラムの動作が誤っている際に、間違えている場所を特定できない初学者を対象として、変数の値や配列の値の移り変わりを把握する支援を目的とする。初学者向けのプログラミング演習担当教員に、変数の値や配列の値の移り変わりを把握するために必要な支援についての聞き取り調査を行い、必要な支援となる機能を備えた可視化支援システムを構築した。そして、初学者に本システムを使用してもらい、アンケート調査によりその有効性を確かめた。

2. 必要な支援

初学者向けのプログラミング演習担当教員に、プログラミング可視化システムを使用して、変数の値や配列の値の移り変わりを把握するために必要な支援についての聞き取り調査を行った。その結果、必要な支援は以下ようになった。

- ① 可視化の自動生成
- ② 命令の実行前後の状態を表示
- ③ 命令の実行前後で変化している部分を強調表示
- ④ 条件文の評価式、プログラム中の計算式の詳細表示
- ⑤ ブロック表示

3. 先行研究

変数の値や配列の値の移り変わりを把握するのに必要な支援と、類似した機能を持つ代表的な可視化システムを紹介する。

3.1 Online Python Tutor

Online Python Tutor (図1) は、学習者が書いたプログラムを、変数の値や配列の値を可視化することでプログラム学習支援を行うシステムである⁽⁴⁾。ソースコード1行ずつの実行後の状態を静的に可視化し、1行ずつ進んだり戻ったりして状態を確認することができる。しかし、一度にひとつの実行状態のみを表示するため、プログラム命令の実行前後で変数の値や配列の値の移り変わりを把握することが難しい。

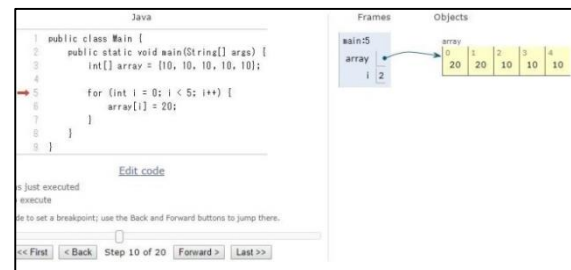


図1 Online Python Tutor

3.2 静岡大学のシステム

静岡大学のシステム (図2) は、アルゴリズムをプログラムで実装した際、アルゴリズムの特長をつかむために可視化を行うシステムである⁽⁵⁾。一度に実行前後の変数の値や配列の値を表示するため、プログラム命令の実行前後で変数の値や配列の値の移り変わりを容易に把握することができる。しかし、学習者が書いたプログラムを自動で可視化するのではなく、講師があらかじめ説明したいプログラムの可視化表示を用意し、表示することでアルゴリズム理解を支援するシステムである。従って、学習者は自分で作成したプログラムについての可視化を行うことができない。

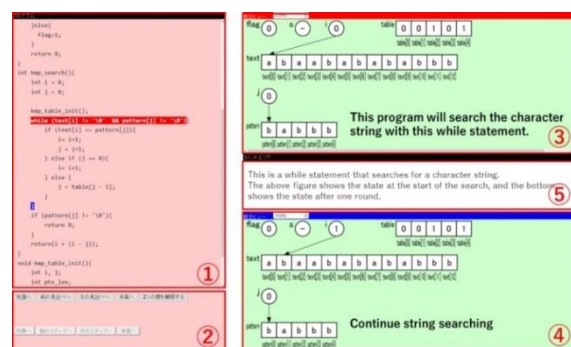


図2 静岡大学のシステム

4. 実装した機能

著者の所属している研究室では、プログラミング演習中に学習者が書いているプログラムの状況をリアルタイムで把握することができるプログラミング演習支援システム PROPEL (PROgramming Practice Easy for Learners) ⁽⁶⁾を開発し、三重大学電気電子工学科のプログラミング演習 I・II の授業^(7, 8)で運用を行っている。本研究の提案するシステムをサブシステムとして付け加える。なお、本システムは、既存の可視化システムである Online Python Tutor を拡張することで実装を行った。

コーディング画面でコーディングされたソースコードは Web サーバに送られる。そして、JSON 形式の実行トレース結果となり Web ブラウザに返されることで可視化を行う。

- ① 可視化の自動生成
- ② 命令の実行前後の状態を表示
- ③ 命令の実行前後で変化している部分を強調表示
- ④ 条件文の評価式とプログラム中の計算式の詳細表示
- ⑤ ブロック表示

これらの機能について、1-3 については実装を行い、4 と 5 は仕様の決定までを行った。以下にそれぞれについて説明する。

4.1 可視化の自動生成

本システムは、既存の可視化システムである Online Python Tutor を拡張することで実装を行い、Online Python Tutor と同様に可視化を自動生成する。本システムの可視化画面を図 3 に示す。画面左側には可視化を行うプログラムのソースコードを、右側に変数の値や配列の値を可視化表示する。

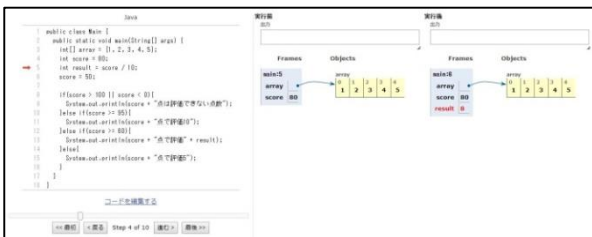


図 3 本システムの可視化画面全体

4.2 命令の実行前後の状態を表示

図 4 に示すように、命令の実行前後の変数の値や配列の値の表示する。

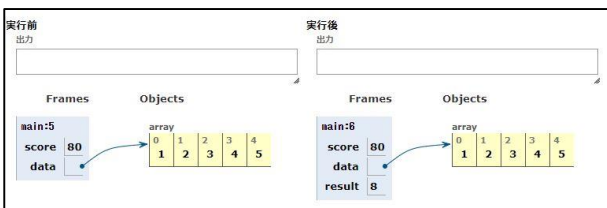


図 4 命令の実行前後の状態表示

4.3 命令の実行前後で変化している部分を強調表示

図 5 に示すように、命令の実行前後で、変数の値や配列の値が変化している部分に色付けをして強調表示を行う。可視化部分に実行前後の状態を表示して変化を示すことで、プログラムのソースコード 1 行の実行前後の動作を確認できるようになっている。

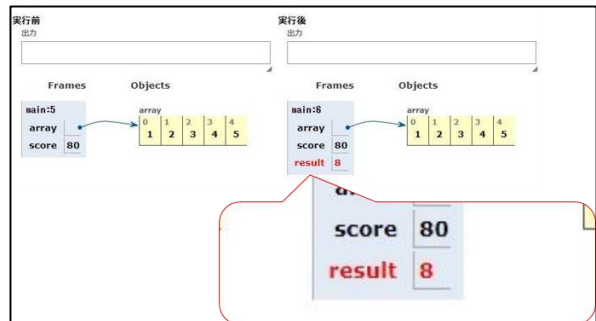


図 5 本システムの可視化部分

4.4 条件文の評価式とプログラム中の計算式の詳細表示

繰り返し文や条件分岐文の評価式や、プログラム中の計算式を表示する機能は、仕様の決定までを行った。イメージ図を図 6 に示す。可視化を行っている命令に関係のある条件判定や繰り返しに必要となる評価式の内容を表示し、その true, false を表示する。同様に、計算式の内容を可視化部分に表示する。これらの機能により、命令の実行前後での変化の理由を把握することを支援する。イメージ図を図 7 に示す。

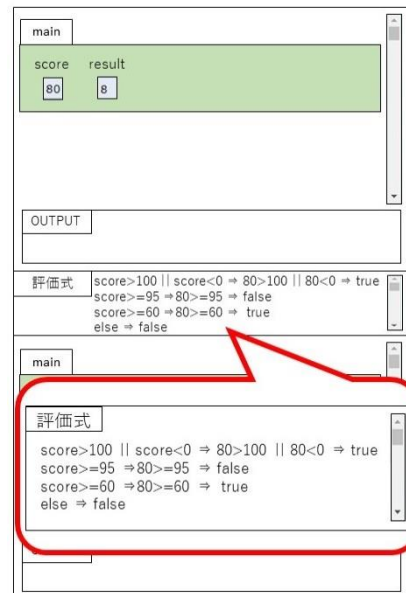


図 6 評価式の詳細表示

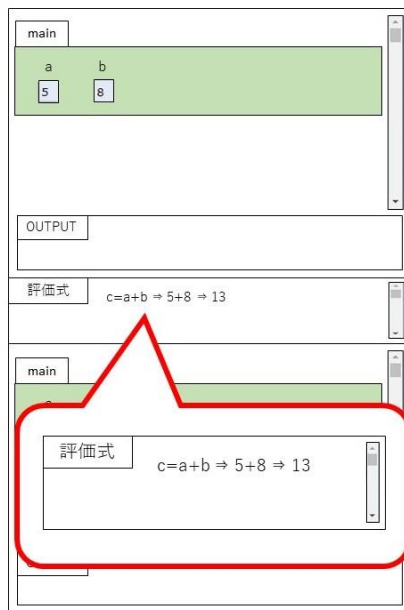


図 7 計算式の詳細表示

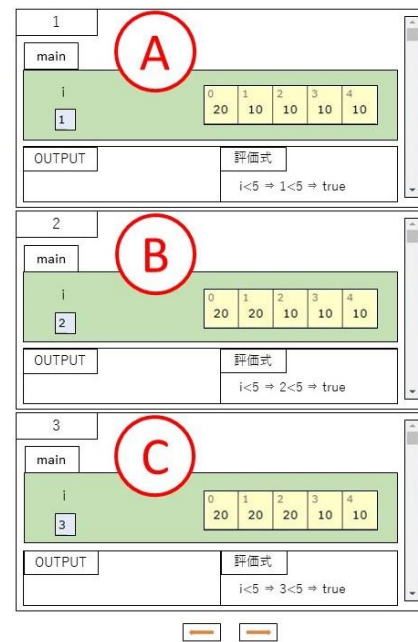


図 9 繰り返し文の可視化表示

4.5 ブロック表示

ソースコード1行の実行前後での可視化だけでなく、繰り返し文や条件分岐等の条件文をひとつのブロックとしてみなし、ひとつのブロックの実行前と実行後の変数の値や配列の可視化を行う機能は、仕様の決定までを行った。イメージ図を図8に示す。条件文をひとつのブロックとして捉え、その実行前後の変化を表示することで、マクロな視点でプログラムの動作を把握することを支援する。

さらに、繰り返し文は1ブロックの実行ごとに3回分の変数と配列の状態を表示する。イメージ図を図9に示す。3回分のブロックの状態を可視化表示することにより、学習者が繰り返し文のブロック実行前後での変数の値や配列の値の変化を把握することを支援する。

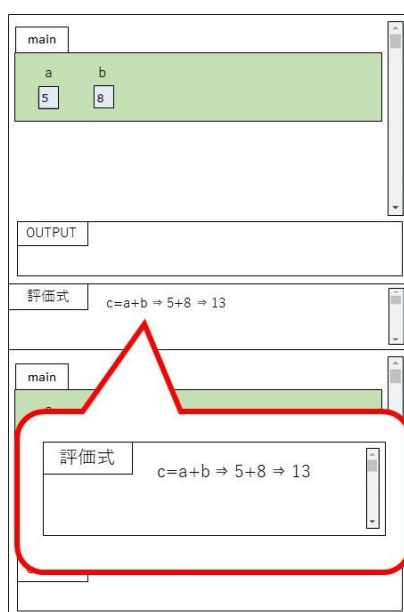


図 8 ブロックの実行前後の状態表示

5. 実験

本システムにより、学習者が、変数の値や配列の値の移り変わりを把握する支援をできているかを確認するため実験を行った。

5.1 実験内容

実験内容は、動作の誤っているプログラムを修正する課題を以下の両システムを使用して行わせるというものである。

- 従来の可視化システム (Online Python Tutor)
- 本システム

両システム使用後にアンケート調査を行うことで評価を行った。アンケート内容は6章に示す。対象者は三重大学工学部電気電子工学科2年のプログラミング演習II受講者64人(2018年度)である。

5.2 結果

アンケートの内容と、アンケートの結果を示す。

アンケート1の内容を図10に、アンケート1の結果を表1に示す。

- 命令の実行前後の状態表示は、従来システムと比べて変数の値や配列の値の移り変わりを把握する際に役立ちましたか。
 1. 役に立たなかった
 2. どちらかといえば、役に立たなかった
 3. ふつう
 4. どちらかといえば、役に立った
 5. 役に立った

図 10 アンケート1の内容

表 1 アンケート 1 の結果

アンケート1						
選択肢	1	2	3	4	5	合計
票数	1	1	27	18	17	64
百分率	2%	2%	42%	28%	27%	100%

アンケート 1 の結果 (表 1) より, 「4 (どちらかといえば役に立った), 5 (役に立った)」の回答が 55%と過半数に達した. これより, 本システムの機能である実行前と実行後の 2 つの状態が表示されることは, 変数や配列の値の移り変わりを把握することに有効であると考えられる.

アンケート 2 の内容を図 11 に, アンケート 2 の結果を表 2 に示す.

- 変化部分の強調表示は, 従来システムと比べて変数の値や配列の値の移り変わりを把握する際に役立ちましたか.
1. 役に立たなかった
 2. どちらかといえば, 役に立たなかった
 3. ふつう
 4. どちらかといえば, 役に立った
 5. 役に立った

図 11 アンケート 2 の内容

表 2 アンケート 2 の結果

アンケート2						
選択肢	1	2	3	4	5	合計
票数	1	0	29	16	18	64
百分率	2%	0%	45%	25%	28%	100%

アンケート 2 の結果 (表 2) より, 「4 (どちらかといえば役に立った), 5 (役に立った)」の回答が 53%と過半数に達した. これより, 本システムの機能である, 実行前後の変化部分に色をつけて示す表示は, 変数や配列の値の移り変わりを把握することに有効であると考えられる.

アンケート 3 の内容を図 12 に, アンケート 3 の結果を表 3 に示す.

- 今後, 本システムと従来システムのどちらを使いたいですか. 使いたくないですか.
1. どちらも使いたくない
 2. 従来システムを使いたい
 3. どちらかといえば, 従来システムを使いたい
 4. どちらかといえば, 本システムを使いたい
 5. 本システムを使いたい

図 12 アンケート 3 の内容

表 3 アンケート 3 の結果

アンケート3						
選択肢	1	2	3	4	5	合計
票数	2	4	6	36	16	64
百分率	3%	6%	9%	56%	25%	100%

アンケート 3 の結果 (表 3) より, 「4 (どちらかといえば, 本システムを使いたい), 5 (本システムを使いたい)」が 81%と過半数に達した. これより, 学習者が自身でプログラムを作成する際, 可視化システムを使用しない方法と従来システムを使用する方法よりも, 本システムを用いて取り組む方法を行いたいという学習者が多いことがわかる.

以上の結果より, 本システムが変数の値や配列の値の移り変わりを把握する支援に有効であることを確認した. また, 従来システムよりも本システムを使用したいという学習者が 80%に達するという結果になった.

6. まとめ

プログラミング初学者が, 自身で作成したプログラムの変数の値や配列の値の移り変わりを把握する支援をすることを目的として, 以下の機能を備えた可視化システムを提案した.

- 可視化の自動生成
- 命令の実行前後の状態を表示
- 命令の実行前後で変化している部分を強調表示

そして実験結果より, 本システムが変数の値や配列の値の移り変わりを把握する支援に有効であることを確認した. また, 従来システムよりも本システムを使用したいという学習者が 80%に達するという結果になった.

参考文献

- (1) 吉村巧朗, 亀井靖高, 上野秀剛, 門田曉人, 松本健一: ブレークポイント使用履歴に基づくデバッグ行動の分析, 電子情報通信学会技術研究報告: 信学技報, vol. 109, no. 307, pp. 8590, 2009
- (2) Sassi Bentrard and Djamel Meslati: Visual Programming and Program Visualization — Toward an Ideal Visual Software Engineering System —, ACEEE International, vol. 1, no. 3, pp. 56-62, 2011
- (3) Christine Alvarado, Briana Morrison, Barbara Ericson, Mark Guzdial, Brad Miller and David L. Ranun: Performance and use evaluation of an electronic book for introductory Python programming, Technical Report GT-IC-12-02, Georgia Institute of Technology, 2012
- (4) Philip J. Guo: Online Python Tutor: Embeddable Web-Based Program Visualization for CS Education, Proceeding of the 44th ACM Technical Symposium on Computer Science Education, pp. 579584, 2013
- (5) Daiki Ihara, Satoru Kogure, Yasuhiro Noguchi, Koichi Yamashita, Tatsuhiro Konishi and Yukihiko Itoh: Algorithm Learning by Comparing Visualized Behavior of Programs, pp. 385390, 2017
- (6) 小川正, 西口大亮, 北英彦: プログラミング演習における iPad などの携帯デバイスの利用による指導の円滑化, コンピュータ利用教育協議会, 2011PC カンファレンス, 2011
- (7) 三重大学ウェブシラバス プログラミング演習 I: <https://syllabus.mie-u.ac.jp/?action=display&id=19567> (2019年1月参照)
- (8) 三重大学ウェブシラバス プログラミング演習 II: <https://syllabus.mie-u.ac.jp/?action=display&id=19574> (2019年1月参照)