7ZH-07

仕様をもとに入力と期待される出力を整理する機能をもった初学者向けプログラミング学習支援*

片桐 健伍 †
静岡大学大学院 総合科学技術研究科

酒井 三四郎 ‡
静岡大学 情報学部

1 はじめに

初学者がプログラミングを学習するにあたっていくつかの課題がある．例えば，関数 (メソッド) の引数に対して期待する出力を十分に予測できないという課題が考えられる．これは，入力値と期待される出力の組み合わせ (以下，成功条件と呼ぶ) を仕様などの情報から整理し，それに沿った開発を行うことで回避できる．しかし，初学者が成功条件を整理し，自らが実装したコードと一致していることを自力で確認することは難しい．また，初学者が検討した成功条件は必ずしも仕様を満たしているとは限らない．これが適切なものであるかを判定・提示することによって実装されるコードの精度がより向上し，より大きな学習効果が得られるのではないかと考えた．学習者自身が仕様をもとにして整理した成功条件がコードと一致しているかを容易に確認できる機能と，学習者が設定した成功条件が仕様に沿ったより良いものであるかを判定する機能をもった初学者向けのプログラミング学習支援ツール neko を開発する．

2 先行研究

本研究に類似している手法として，テスト駆動開発 [1] が挙げられる．テスト駆動開発では分析手法などを用いた精巧な単体テストが求められるが，本研究における成功条件は必ずしも精巧なものである必要はなく，この点においてテスト駆動開発と異なる．

本研究の先行例として，入力値と結果の組み合わせからプログラミング学習を行うという点で CodingBat[2] が挙げられる．CodingBat では，入力値と期待される出力の組み合わせが教材の提供者によって設定されるが，本研究では学習者が自ら入力値と期待される出力の組み合わせを考える．この手法によって，正確なプログラミング学習を支援できると考えている．

また，すでに先行して学習者自身が仕様をもとにして整理した入力と期待される出力がコードと一致しているかを確認できる機能を有したソフトウェアを開発し，効果の検証を行っている [3]．本研究ではこれを発展させ，より良い成功条件を設定することができたかを判定する機能を追加した．この機能の追加によって学習効果がさらに高まることを期待している．

3 提案ツール

本研究では，1 節で挙げた問題を解決し，初学者のプログラミング学習を支援するためのツール neko を開発した．今回開発した neko には，次のような機能がある．

1. 提示された仕様を学習者自身が整理する機能
2. 学習者が検討した成功条件と自身のコードが一致しているかを確認する機能
3. 学習者が検討した成功条件が仕様に沿ったものであるかを確認する機能

提示された仕様は，学習者がこれを読んで neko 上に入力する必要がある．入力された内容は neko 上において成功条件を整理する際やコーディングを行う際に確認することができ，学習者がこれに沿ったコードを実装することを期待している．実装したコードの実行時には，成功条件に基づいて単体テストコードを自動で生成し実行する．正常に実行が完了した場合，その出力を整形して neko 上に表示する．実際に neko を使用している画面の例を図 1 に示す．

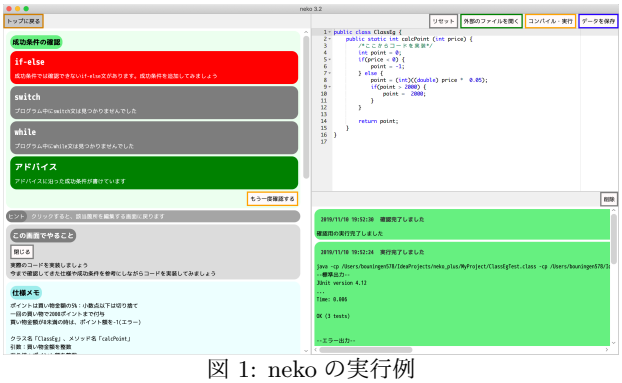

図 1: neko の実行例

図 1 に示した画面では，整理した仕様やプログラミングを行うエディタなどと共に，成功条件を判定した結果が画面左上に表示されている．

学習者が作成したコードは，学習者が設定した成功条件には十分に沿ったものであると考えられる．ただし，学習者が設定した成功条件そのものが仕様の一部を見落としている場合，実装されたコードは仕様に沿っているとは言えない．これに対処するため，neko には成功条件が仕様に沿った良いものであるかを確認する機能が付けられている．これは，学習者のコードから neko が判断するものと，教師が事前に設定したアドバイスから判断するものがある．neko が学習者のコードから判断する際には，学習者のコードに含まれている if 文などの条件分岐から，学習者が設定した成功条件がこれらを検討できるかを判断する．また，教師が事前に設定したアドバイスから判断する際には，教師が学習にあたって必要と考える成功条件に対して学習者が設定した成功条件が不足していないかを判断する．それぞれの情報をもとに判定した結果が図 1 で示した画面の左上に表示される．成功条件が不足していると判断された場合には，成功条件をより良いものとするために成功条件の見直しが提案さ

*Learning support for programming beginner with functions to organize inputs and expected outputs based on specifications

†Kengo KATAGIRI, Graduate School of Integrated Science and Technology, Shizuoka University

‡Sanshiro SAKAI, Faculty of Informatics, Shizuoka University

れる．これによって，学習者の実装したコードが仕様に沿っていなかったり，仕様の一部を見落とすことを抑止できると考えている．

4 評価実験

本研究では，neko を使用したほうが使用しないよりも正確なコーディングができるという仮説の検証のため，評価実験を行った．

4.1 実験方法

被験者は，学部 1 年生 5 名で全員がプログラミングの講義を受講している．本実験では，まず neko を使用することなくプログラミングの課題に 2 問解答する．その後，neko を使用して先ほどと同難易度で内容は異なる課題に 2 問解答する．neko を使用した課題に解答する際には，良い成功条件が設定できたかを判定する機能を使用する群 (A 群) と，使用しない群 (B 群) に被験者をランダムに分けた．得られた結果は，以下の観点をもとに評価を行う．

- より仕様にあった正確なコードが書けるようになったか
- より良い成功条件が設定できるようになったか
- より短時間でコーディングができたか

さらに，課題に解答する前後にはプログラミングについてや実験に関するアンケートを行う．以上の内容について 3 つの観点の検証を行うことで仮説を検討する．

4.2 実験結果と考察

4.1 節で示した，評価実験の計画に沿って実験を行った．各群ごとの，課題の結果と解答に要した時間を表 1 に示す．課題はすべて 3 点満点で，解答に要した時間はファイルを開いてから実装が完了し保存するまでの時間を示している．カッコ内に示した時間は，所要時間のうち成功条件の作成や確認のための実行などを除いたコーディングのみに要した時間を示している．

表 1: 各群のテストの採点結果と解答所要時間

	neko を使用しない		neko を使用した	
	第 1 問	第 2 問	第 1 問	第 2 問
A 群	-	-	2.7/3 点 14:38(04:11)	2.7/3 点 12:33(04:34)
B 群	-	-	3.0/3 点 15:38(04:53)	1.0/3 点 15:18(06:18)
全体	1.8/3 点 15:20(12:55)	2.6/3 点 10:34(09:07)	2.8/3 点 15:39(05:01)	2.0/3 点 13:52(05:56)

表 1 の結果から，neko を使用しなかった時に比べて，neko を使用した時の方が平均点は良く，コーディングに要した時間のみを見ると neko を使用した時の方が短かった．よって，neko を使用した方がより効率よく適切なコードを書けていた．また，A 群の方が平均点は高いことから，より良い成功条件であるかを判定する機能も，正確なコーディングに一定の貢献をしていると考えた．

各被験者ごとの neko を使用する問題における作成した成功条件の数を表 2 に示す．カッコ内の値は自身が作成したコードを初めて実行した後に追加した成功条件の数を示している．

表 2: 各被験者ごとの作成した成功条件の数

ID	群	第 1 問	第 2 問
1	A	4(0)	8(1)
2	A	4(1)	6(0)
3	A	3(1)	4(0)
4	B	2(0)	3(1)
5	B	4(0)	6(0)

表 2 の結果から，成功条件の判定を行った A 群はコンパイル後に提案された成功条件の追加を全員が行っていた．このとき，被験者 2 と 3 は第 1 問での指摘を受けて第 2 問では十分な成功条件を初めから設定することができた．このことから，より良い成功条件であるかを判定する機能は，被験者に有益な情報を提供できており，かつより良い成功条件を書けるようになったと言える．

また，各被験者に対するアンケートでは neko の使用感について肯定的な意見を多く得た．さらに，仕様を読んでそれを成功条件といった形にまとめてからプログラミングを行うことの効果を多くの被験者が実感していた．よって，neko を通じた初学者のプログラミング学習や，今回提案した入力値と期待される出力の組み合わせ (成功条件) を仕様などの情報から整理し，それに沿った開発を行う手法は一定の効果があると考えられる．

5 おわりに

本研究では，プログラミング学習支援として仕様から整理された入力値と期待される出力がコードと一致しているか，仕様に沿った良い成功条件を設定できているかを確認できる初学者向けのコーディング支援ツール neko を開発した．評価実験の結果から，neko によって学習者が短い時間で効率的にプログラミングを学習するという点で一定の効果があることが分かった．また，新たに実装したより良い成功条件であることを判定する機能によって学習者が設定する成功条件の質が向上した．今後，成功条件の判定機能の精度向上や機能の充実を図り，プログラミング初学者のさらなる学習効果向上を目指したい．

参考文献

[1] Kent Beck: *Test-driven development: by example*, Addison-Wesley Professional (2003).

[2] Nick Parlante: *CodingBat*, `https://codingbat.com/java` (2019.01.07).

[3] 片桐 健伍, 酒井三四郎: 仕様をもとに入力と期待される結果を整理する機能をもった初学者向けプログラミング学習支援, 第 81 回 情報処理学会 全国大会講演論文集, pp.549-550 (2019).