デバッグ演習への意欲向上を目的とした順位付けをしない得点提示式演習の提案

宮島 健太　篠埜 功

デバッグ学習では，事前に用意されたバグを含むプログラムのデバッグを行う演習を実施することがある．その一例としてデバッグコンテストがあるが，演習に競争要素を導入すると演習への意欲が低下することがある．本研究では，演習への意欲を向上させることを目的とし，順位付けを行わずに演習得点のみを提示するデバッグ演習を提案する．提案手法に基づき，得点提示式演習システムを Web アプリケーションとして実装した．提案手法の有効性を評価するため，実装したシステムを用いて，芝浦工業大学情報工学科の学生 11 名を対象に，提案手法，得点を提示しない形式，コンテスト形式の 3 つのグループに分け，演習への意欲が向上するかどうかを確認する実験を行った．本論文では，その結果を報告する．

1　はじめに

ソフトウェア開発においてデバッグ作業は不可欠であり，開発コスト全体の 50%〜75%を占めている [9]．しかし，現在のプログラミング教育では，デバッグ学習が行われることは少ない [12] [6]．デバッグ作業はプログラミング初学者にとって困難であり，プログラミング学習のついでに習得できるものではないため，別途指導するべきである [8]．

デバッグ学習では，事前に用意されたバグを含むプログラムのデバッグを行う演習（以下「デバッグ演習」という）を実施することがある．その一例としてデバッグコンテストが挙げられる．コンテストにおいては通常，点数で順位付けをして参加者に提示するということが行われ，コンテストをプログラミングやデバッグの教育に導入する研究もこれまでに多数行われた [18] [21] [16] [13]．しかし，プログラミング演習にこのような競争要素を導入すると演習への意欲が低下することがある [7] [21]．

そこで本研究では，デバッグ演習への意欲を向上させることを目的とし，順位付けを行わずに得点のみを提示するデバッグ演習（以下「得点提示式演習」という）を提案する．また，提案手法に基づいた演習システムを Web アプリケーションとして実装する．さらに，提案手法の有効性を評価するため，実装したシステムを用いて，芝浦工業大学情報工学科の学生 11 名を対象に，デバッグ演習への意欲が向上するかどうか確認する実験を行う．

2　関連研究

近年，デバッグ演習に関する研究が盛んに行われており，デバッグ学習の手法としてだけでなく，プログラミングやアルゴリズムの学習の手法としても研究されている．また，デバッグ演習と意欲との関連について調査した研究が少し行われている．以下ではこれらについて本研究と比較しながら議論する．

2.1　デバッグ演習によるデバッグ学習

本研究で提案するデバッグ演習は，学習者の意欲を向上させることを目的としており，デバッグ能力を向上させるような支援は行わない．これまでに行われ

---

* An Approach to Motivating Students in Debugging Exercises by Feedbacking Scores without Ranking
This is an unrefereed paper. Copyrights belong to the Authors.

Kenta Miyajima, Isao Sasano, 芝浦工業大学大学院 理工学研究科 電気電子情報工学専攻, Electrical Engineering and Computer Science, Graduate School of Engineering and Science, Shibaura Institute of Technology.

てきたデバッグ演習に関する研究では，デバッグ能力を向上させることを目的としたものがいくつかあり，ここではそれらについて概観する．

山本ら [17] は，授業を座学と演習のセットで行う必要があると考え，体系的デバッグ手順を学習するための座学教材と，その座学で学習した手順を辿ることを支援するシステムを開発した．彼らの学習方式でプログラミング初学者を対象に実験を行った結果，座学が体系的デバッグ手順に関する知識の獲得に有効であることが示された．また，システムを用いて座学で学習した手順を辿らせることが，体系的デバッグ手順に関する知識の定着に有効であることが示された．

Lee ら [10] は，デバッグ演習においてデバッグのヒントや正誤判定，誤答した際の解説により学習の支援を行うシステムを開発した．デバッグのヒントはバグの内容に関する説明であり，学習者がヒントを要求したタイミングで提示される．3 回誤答した場合にはバグを修正するための手順が示される．この学習支援システムを用いてプログラミング初学者を対象に実験を行った結果，システムを用いてデバッグを練習したグループの練習後のデバッグテストの得点が，学習支援無しでデバッグを練習したグループよりも有意に高いことが示された．

これらの研究は，本研究とは目的も内容も異なっており，直接の関係はないが，将来的にはこれらの研究を参考にしてデバッグ支援機能を追加することも考えられる．

2.2 デバッグ演習によるプログラミング学習

プログラミングやアルゴリズムの学習においてデバッグをする機会を設けることで，デバッグをする機会のない学習と比べて学習量が増加したりアルゴリズムの理解度が向上するなど，良い影響があることを示した研究がいくつかある．

岩渕ら [14] は，構文の説明やバグを含んだソースコードなどからなる教材を用いた学習法を提案した．バグを含んだソースコードを修正できないまま学習開始から 15 分が経過した場合，学習者は修正後のソースコードとバグの解説を見ることができるようになる．この学習法でプログラミング初学者を対象に実験を行った結果，プログラミングに関する学習が能動的になり，学習量が増加することが示された．

倉持ら [19] は，アルゴリズム学習においてコードリーディングの代わりにバグ修正問題を解く手法を提案した．この手法でプログラミング初学者を対象に実験を行った結果，バグ修正問題を解いた場合のアルゴリズム理解度確認問題の正答数がコードリーディングを行った場合よりも有意に高いことが示された．また，学習時間はバグ修正問題を解いたときの方が有意に長くなっており，自律的な学習時間が伸びたことが示された．

長瀧ら [20] は，アルゴリズム学習を目的としたデバッグ演習の手法を提案した．この演習では，学習者はバグが含まれているソースコードとアルゴリズム名を提示され，アルゴリズムを思い出すことでバグを発見し，修正する．この手法でプログラミング初学者を対象に実験を行った結果，この手法がアルゴリズムの理解に役立ったという意見が多く得られた．

これらの研究は本研究と相互に補い合うものと考えられ，例えば，本研究と同様に上記の研究における学習手法において学習者に得点を提示することにより，学習量の増加やアルゴリズムの理解度向上にさらなる効果が見込めるかもしれない．これらに関する調査は今後の課題の候補の 1 つである．

2.3 デバッグ演習への意欲の調査

デバッグ演習とデバッグ学習意欲やデバッグ演習に対する意欲との関連について調査した研究がこれまでいくつか行われてきた．

空谷 [15] は，プログラミング初学者を対象に，出題機能とプログラムの編集及び実行の機能のみを持つシステムを用いてデバッグ演習を実施した．演習後にアンケートを行った結果，プログラミング科目の成績が低い学生は成績が高い学生に比べ，演習への意欲が低いことが示された．

Rein ら [13] は，大学院生とプロのプログラマを対象に，デバッグにかかった時間と正確さによって評価されるデバッグコンテストを実施した．コンテスト後のインタビューでは，全ての参加者が今後，デバッグコンテストに参加することに興味があると述べてい

る．また，コンテスト後のアンケートでは，ほとんどの参加者がデバッグ学習の意欲が向上したと述べている．

これらの研究では，提案手法とそれ以外の手法の比較が行われていない．また，アンケートは演習後の一度のみ実施されており，演習による効果が明らかになっていない．これに対し本研究では，提案手法と得点を提示しない形式及びコンテスト形式を比較する実験を行っている．また，アンケートは複数回実施し，演習による効果の調査になっている．

3 得点提示式演習

本研究で提案する得点提示式演習では，学習者は一定時間の間，デバッグ問題に取り組む．演習時間中，各問題で解答を提出した際，デバッグできているかどうかの正誤判定が行われる．制限時間内であれば各問題に何度でも取り組める．制限時間終了後もしくは全問正解後，学習者に得点が提示される．

本演習は以下の流れで行う．

1. 事前準備
   教師は，各問題に関して以下のような情報を演習システムに登録する．
   - プログラムの仕様を説明した文章
   - 複数の実行例（入力と出力の組み合わせ）
   - バグを含むソースコード
2. 出題
   教師によって登録された情報を基にシステムが学習者に問題を出題する．問題の出題においては，プログラムの仕様を説明した文章，複数の実行例，バグを含むソースコードが学習者に提示される．実行例は教師が登録したもの全てが提示される．
3. デバッグ
   学習者は，提示されたバグを含むソースコードのデバッグを行う．この際，コンパイラやデバッガ，エディタなどの使用に制限はなく，学習者は自由にデバッグすることができる．
4. 正誤判定
   システムが，登録された実行例を基にデバッグできているかどうかの判定を行う．各実行例において，登録された入力を標準入力として学習者のプログラムに与えたときの標準出力と，登録された出力を比較する．教師によって登録された全ての実行例において正解と判定した場合，デバッグできていると判定する．
5. 得点提示
   制限時間終了後もしくは全問正解後，システムが得点を計算し学習者に提示する．

問題の出題において学習者に教師が登録した全ての実行例を提示するのは，多くの初学者が実行例以外の入力はテストしなくてもいいと認識している [11] ことから，提示された実行例全てについて正しく動作するにもかかわらず正誤判定において不正解になった場合に，不正解の理由が分からず詰まってしまうのを回避するためである．

4 得点提示式演習システムの機能

提案手法に基づき，デバッグ演習システムを実装した．本システムのソースコードは `https://www.cs.ise.shibaura-it.ac.jp/jssst2024/` で公開している．本システムは Node.js [5]，Web アプリケーションフレームワーク Express.js [3]，及びテンプレートエンジン EJS [2] を用いた Web アプリケーションである．

4.1 出題

教師によって登録された情報を基に問題を出題する．出題画面の例を図 1 に示す．画面左上は問題文であり，その中にプログラムの仕様を説明した文章が含まれている．画面左下は実行例であり，1 つずつ表示される．学習者はドロップダウンリストから選択することで，複数ある実行例を切り替えて表示することができる．実行例の選択画面の例を図 2 に示す．画面右側は教師が登録した，バグを含むソースコードである．

本システムでは，学習者がプログラムの実行結果を確認できるような機能は提供していない．そのため，学習者は自身でプログラム実行環境を用意する必要がある．学習者は「初期ファイルダウンロード」ボタンを押すことで各問題のソースコードをダウンロー

ドすることができる．学習者は自身の環境でデバッグし,「ファイルの選択」ボタンを押してそのファイルを指定し,「ファイルアップロード」ボタンを押すことで，デバッグ後のソースコードをエディタに表示することができる．Web アプリケーションのエディタ部分には Monaco Editor [4] を利用しており，ブラウザ上で直接編集することもできる．

4.2 正誤判定

各問題において，教師によって登録された実行例を基に学習者がデバッグできているかどうかの判定を行う．登録された入力を標準入力として学習者が提出したプログラムを実行した時の標準出力が，登録された出力を連続する記号列として含んでいた場合，その実行例において正解と判定する．プログラムの実行には npm パッケージ compile-run [1] を利用する．この判定処理を登録された全ての実行例に対して行い，全ての実行例において正解と判定した場合，デバッグできていると判定する．

4.3 制限時間管理

得点提示式演習システムには演習開始ボタンがあり，学習者がそのボタンを押した時点からの経過時間で制限時間内かどうかを判定する．制限時間が過ぎると，解答の提出ができなくなる．

5 評価実験

得点提示式演習の有効性を評価するため，前節で述べた得点提示式演習システムを用いて，芝浦工業大学情報工学科の学生 11 名を対象に実験を行った．この実験では，得点を提示しない形式，得点提示式演習，コンテスト形式の 3 つの手法で比較を行った．被験者を 3 グループに分けるため，実験の最初に得点を提示しない形式での演習を行った．この演習の得点を基に被験者を 3 つのグループに分け，前半セッション，中間アンケート，後半セッション，最終アンケートの順に行った．各セッションでは，グループごとにそれぞれ異なる手法で演習を行った．

5.1 演習内容

本実験においては各演習の制限時間は 15 分とし，被験者はこの時間内で全 5 問のデバッグ問題に取り組んだ．出題する問題の難易度は 5 段階とし，各レベルから 1 問ずつ出題する．バグを含むソースコードについて，いくつかの項目に関するレベルごとの分類を表 1 に示す．

表 1 において,「なし」はその項目を含まないソースコードであることを示しており，例えば if 文はレベル 1 の問題のソースコードには含まれない．バグを含むソースコードは，仕様通りのソースコードを書き，それをコンパイルは通るが仕様を満たさなくなるように数行変更することで第一著者が作成した.「変更箇所」という項目は，変更した行数を示している．

得点は，デバッグに成功した問題の難易度，誤答回数，及び残り時間を用いて計算する．また，この得点計算式は前半セッション前に被験者に提示した．

- 難易度基礎点
  難易度基礎点は，レベル $\times 1,000$ 点とする．
- 誤答による減点
  誤答による減点は，デバッグに成功した問題のみを対象とし，問題ごとに行う．誤答 1 回につき 100 点の減点とする．ただし，各問題 6 回目以降は減点せず，最大でも各問題 500 点の減点とする．
- 残り時間による加点
  残り時間による加点は，最後に正解した時の残り時間を基に加点する．全問正解により演習を終了した場合，1 秒につき 1 点の加点を行う．制限時間により演習を終了した場合，2 秒につき 1 点の加点を行う．

5.2 グループ分け前の演習

グループ分けの前に行う演習では，演習システムを用いた 15 分の演習を 1 回行う．この際，得点は被験者に提示されない．演習終了後，被験者を平均得点が同程度になるように，得点を提示しないグループ 3 名，提案手法グループ 4 名，コンテストグループ 4 名の 3 グループに分けた．

図 1　本研究の得点提示式演習システムにおけるデバッグ問題出題画面のスクリーンショット

表 1　評価実験の各演習で提示するバグを含むソースコードについて，いくつかの項目に関するレベルごとの分類

レベル	関数	if 文	for 文と while 文	配列	ポインタ型変数	変更箇所
1	main 関数のみ	なし	なし	なし	なし	1 行
2	main 関数のみ		なし	なし	なし	1 行
3	main 関数のみ				なし	1 行
4	main 関数以外に 1 つ					1 行
5	main 関数以外に 1 つ					2 行

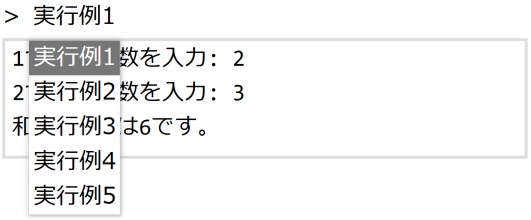

図 2　本研究の得点提示式演習システムにおいて実行例を切り替えるドロップダウンリストの 1 例のスクリーンショット

5.3　前半セッションと後半セッション

前半セッション及び後半セッションはそれぞれ 35 分とし，同様の流れで行う．各セッションでは演習システムを用いた 15 分の演習を 2 回行う．各演習の終了後，被験者が得点履歴やランキングを見る時間を取り，余裕を見て各セッションは 35 分間で実施した．これらのセッションでは，5.2 節で述べた 3 グループに分かれ，それぞれ異なる手法で演習を行う．得点を提示しないグループでは，得点提示をせずに，それ以外は得点提示式演習と同じように演習を行う．提案手法グループでは，得点提示式演習通りに演習を行う．コンテストグループでは，得点提示式演習にランキング機能を追加して演習を行う．ランキング機能では，前半セッション及び後半セッションにおける合計 4 回の各演習の得点でのランキングと，各被験者の各演習終了時点でのそれまでの演習における最高得点のランキングの計 5 つのランキングが被験者に提示される．各ランキングでは，被験者全員の得点と順位が表示される．その際，自分以外の被験者は匿名で表

示される．

5.4 アンケート

前半セッション後と後半セッション後にアンケートを行った．前半セッション後に行う中間アンケートの項目を表 2 に示す．選択肢は全てのアンケート項目において「全く当てはまらない」「あまり当てはまらない」「どちらともいえない」「やや当てはまる」「とても当てはまる」の 5 つとした．後半セッション後に行う最終アンケートの項目は表 2 の項目に，表 3 の項目を加えたものとした．表 3 において，Q7 と Q8 は提案手法グループ及びコンテストグループに対する質問であり，Q9 と Q10 はコンテストグループのみに対する質問である．

6 実験結果と考察

この節では，前節で述べた評価実験の結果を示し，考察を行う．演習の得点と意欲の関連を調査するため，各被験者の演習の合計得点を用いて，全被験者 11 名を上位 5 名と下位 6 名に分けた．その結果，得点を提示しないグループでは上位 1 名，下位 2 名，提案手法グループでは上位 2 名，下位 2 名，コンテストグループでは上位 2 名，下位 2 名となった．

6.1 アンケート結果

中間アンケートの結果を表 4 に，最終アンケートの結果を表 5 に示す．これらの表の各数値は，5 つの選択肢「全く当てはまらない」から「とても当てはまる」に対して，1 から 5 の数値を割り当てたときの，各グループの上位と下位の回答の平均値である．この上位と下位は上記で述べた被験者全体を分けたものである．

6.2 デバッグ演習への意欲

この節では，表 4 と表 5 で示した Q1 の回答結果について述べる．

各グループで上位と下位に分けずに中間アンケートと最終アンケートで差があるかどうか有意水準 5%で対応のある t 検定を行ったところ，どのグループにおいても有意差は見られなかった．また，中間アンケート及び最終アンケートについて，上位と下位に分けずにグループ間で差があるかどうか有意水準 5%で Mann-Whitney の U 検定を行ったところ，中間アンケートと最終アンケート共に，どのグループ間においても有意差は見られなかった．

6 節の冒頭で分けた得点上位 5 名それぞれについて，中間アンケートと最終アンケートで回答が同じであった．また，表 4 と表 5 において，各グループの得点上位の回答平均値は，得点を提示しないグループ，コンテストグループ，提案手法グループの順に高い．

6 節の冒頭で分けた得点下位 6 名のうち，中間アンケートと最終アンケートで異なる回答をした被験者が 2 名いた．表 4 と表 5 において，各グループの得点下位の回答平均値について，中間アンケートと比べて最終アンケートでは，得点を提示しないグループは減少し，提案手法グループは増加した．これにより，最終アンケートにおける各グループの得点下位の回答平均値は，提案手法グループ，得点を提示しないグループ，コンテストグループの順に高い結果となった．

今回の実験は短時間であったが，長期に渡ってデバッグ演習を行った場合，得点が低い被験者のデバッグ演習への意欲を向上させる手法として，今回行った 3 つの手法の中では提案手法が最も有効であるという結果が得られることが期待される．

6.3 デバッグ学習の意欲

この節では，表 4 と表 5 で示した Q2 の回答結果について述べる．

各グループで上位と下位に分けずに中間アンケートと最終アンケートで差があるかどうか有意水準 5%で対応のある t 検定を行ったところ，どのグループにおいても有意差は見られなかった．中間アンケートについて，上位と下位に分けずにグループ間で差があるかどうか有意水準 5%で Mann-Whitney の U 検定を行ったところ，得点を提示しないグループとコンテストグループ間で p 値が 0.041 となり，得点を提示しないグループの方がコンテストグループよりも有意に高い結果となった．また，最終アンケートについても同様の検定を行ったところ，得点を提示しないグループとコンテストグループ間で p 値が 0.035 と

表 2　中間アンケートと最終アンケートの共通の項目

	項目
Q1	今後，前半セッション及び後半セッションの形式のデバッグ演習に取り組みたいと思った．
Q2	今後，デバッグについて学びたいと思った．

表 3　最終アンケートでの追加項目

	項目
Q3	プログラミングは得意である．
Q4	デバッグは得意である．
Q5	ランキング上位だった場合，積極的に順位を上げたいと思う．
Q6	ランキング下位だった場合，積極的に順位を上げたいと思う．
Q7	演習終了時に得点が提示されることで，演習への意欲が向上した．
Q8	演習終了時に得点が提示されることで，デバッグをより学びたいと思った．
Q9	ランキング機能によって，演習への意欲が向上した．
Q10	ランキング機能によって，デバッグをより学びたいと思った．

表 4　中間アンケートの結果

	得点を提示しない		提案手法		コンテスト	
	上位	下位	上位	下位	上位	下位
Q1	5.0	4.5	3.0	4.0	4.5	3.0
Q2	5.0	5.0	4.0	4.5	3.0	3.5

表 5　最終アンケートの結果

	得点を提示しない		提案手法		コンテスト	
	上位	下位	上位	下位	上位	下位
Q1	5.0	4.0	3.0	4.5	4.5	3.0
Q2	5.0	5.0	4.0	4.5	3.0	4.0
Q3	3.0	2.5	4.0	2.5	4.5	3.0
Q4	2.0	2.0	3.5	3.5	3.0	3.0
Q5	5.0	4.0	4.0	3.0	4.0	4.5
Q6	5.0	4.5	3.5	4.5	3.0	3.5
Q7			4.5	4.0	4.5	4.0
Q8			4.0	4.5	3.0	4.0
Q9					4.5	4.0
Q10					3.0	4.0

なり，得点を提示しないグループの方がコンテストグループよりも有意に高い結果となった．

6 節の冒頭で分けた得点上位 5 名それぞれについて，中間アンケートと最終アンケートで回答が同じであった．また，表 4 と表 5 において，各グループの得点上位の回答平均値は，得点を提示しないグループ，

提案手法グループ，コンテストグループの順に高い．

6 節の冒頭で分けた得点下位 6 名のうち，中間アンケートと最終アンケートで異なる回答をした被験者が 1 名いた．表 4 と表 5 において，各グループの得点下位の回答平均値について，中間アンケートと比べて最終アンケートでは，コンテストグループは増加した．しかし，最終アンケートにおける各グループの得点下位の回答平均値は中間アンケートと同じく，得点を提示しないグループ，提案手法グループ，コンテストグループの順に高い．

上記のように，デバッグ学習の意欲については提案手法は有効ではないかもしれない．

6.4 得点とランキングの提示が意欲に及ぼす影響

コンテストグループの全ての被験者が，表 3 における Q7 と Q9，Q8 と Q10 で同じ回答をしており，得点とランキングの提示について，意欲に及ぼす影響に差が見られなかった．

得点の提示については，得点計算式を調整することで，意欲に及ぼす影響をより良いものにできるかもしれない．例えば，誤答による減点を行わないことで，正誤判定を気軽に行うことができるようになり，演習への意欲が向上するかもしれない．

7 まとめと今後の課題

本研究では，演習への意欲を向上させることを目的とし，順位付けを行わずに演習得点のみを提示するデバッグ演習を提案した．また，提案手法に基づいた演習システムを Web アプリケーションとして実装した．実装したシステムを用いて，芝浦工業大学情報工学科の学生 11 名を対象に，提案手法，得点を提示しない形式，コンテスト形式の 3 つの手法で比較する実験を行った．その結果，提案手法によって，得点が低い被験者 2 名のうち 1 名のデバッグ演習への意欲が向上した．また，得点が低い被験者のデバッグ演習への意欲に関するアンケートの回答平均値においては，比較した 3 つの手法の中で提案手法が最も高かった．

今後の課題として，長期に渡ってデバッグ演習を行った場合の効果を調べることや，得点計算式の違いによる効果の差を調べることが挙げられる．

謝辞 本研究を進めるにあたって，実験に協力していただいた皆様に感謝申し上げます．

参 考 文 献

[1] : compile-run, `https://www.npmjs.com/package/compile-run`.
[2] : EJS – Embedded JavaScript templates, `https://ejs.co/`.
[3] : Express.js, `https://expressjs.com/`.
[4] : monaco-editor, `https://microsoft.github.io/monaco-editor`.
[5] : Node.js, `https://nodejs.org/`.
[6] Beller, M., Spruit, N., Spinellis, D., and Zaidman, A.: On the Dichotomy of Debugging Behavior among Programmers, *Proceedings of the 40th International Conference on Software Engineering*, Association for Computing Machinery, 2018, pp. 572–583.
[7] Fischer, K., Vaupel, S., Heller, N., Mader, S., and Bry, F.: Effects of Competitive Coding Games on Novice Programmers, *Educating Engineers for Future Industrial Revolutions*, Springer International Publishing, 2021, pp. 464–475.
[8] Fitzgerald, S., McCauley, R., Hanks, B., Murphy, L., Simon, B., and Zander, C.: Debugging From the Student Perspective, *IEEE Transactions on Education*, Vol. 53, No. 3(2010), pp. 390–396.
[9] Hailpern, B. and Santhanam, P.: Software debugging, testing, and verification, *IBM Systems Journal*, Vol. 41, No. 1(2002), pp. 4–12.
[10] Lee, G. C. and Wu, J. C.: Debug It: A debugging practicing system, *Computers & Education*, Vol. 32, No. 2(1999), pp. 165–179.
[11] Murphy, L., Lewandowski, G., McCauley, R., Simon, B., Thomas, L., and Zander, C.: Debugging: The good, the bad, and the quirky — a qualitative analysis of novices' strategies, *ACM SIGCSE Bulletin*, Vol. 40, No. 1(2008), pp. 163–167.
[12] Perscheid, M., Siegmund, B., Taeumel, M., and Hirschfeld, R.: Studying the advancement in debugging practice of professional software developers, *Software Quality Journal*, Vol. 25, No. 1(2017), pp. 83–110.
[13] Rein, P., Beckmann, T., Geier, L., Mattis, T., and Hirschfeld, R.: Competitive Debugging: Toward Contests Promoting Debugging as a Skill, *Proceedings of the 2022 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, Association for Computing Machinery, 2022, pp. 172–184.
[14] 岩渕悠太, 高島健太郎, 西本一志: 意図的に埋め込まれたバグによるプログラミング学習支援, 情報処理学会研究報告. *HCI,* ヒューマンコンピュータインタラクション, Vol. 2020-HCI-187, No. 12(2020), pp. 1–6.
[15] 空谷知之: 商業高校におけるデバッグ教材の開発と課題, 教育情報研究, Vol. 37, No. 1(2021), pp. 43–52.
[16] 坂本一憲, 鷲崎弘宜, 深澤良彰: プログラミング初学

者向けコンテストシステム, 日本ソフトウェア科学会大会論文集, Vol. 30(2013), pp. 46–53.
[17] 山本頼弥, 野口靖浩, 小暮悟, 山下浩一, 小西達裕, 伊東幸宏: 場当たり的なデバッグを行ってしまう学習者に体系的デバッグ手順を指導する授業パッケージと学習支援システムの構築, 教育システム情報学会誌, Vol. 35, No. 1(2018), pp. 21–37.
[18] 前田新太郎, 古池謙人, 東本崇仁: ロボットプログラミングを題材にした競争型知識共有プラットフォームの提案と実装, 人工知能学会研究会資料 先進的学習科学と工学研究会, Vol. 91(2021), pp. 14.
[19] 倉持雄樹, 坂本一憲, 鷲崎弘宜, 深澤良彰: バグ修正とコードリーディング-アルゴリズム学習に適しているのはどちらか-, 情報処理学会研究報告. *CE,* コンピュータと教育, Vol. 2021-CE-160, No. 4(2021), pp. 1–7.
[20] 長瀧寛之, 伊藤亮太, 大下福仁, 角川裕次, 増澤利光: アルゴリズム学習における間違い探し形式の演習課題を自動生成する手法の提案と評価, 情報処理学会論文誌, Vol. 49, No. 10(2008), pp. 3366–3376.
[21] 富永浩之, 川崎慎一郎: 競争型学習を取り入れた入門的 C プログラミング演習 - 運用実験での実行テスト系列の利用状況 -, 情報処理学会研究報告. *CE,* コンピュータと教育, Vol. 2010-CE-104, No. 4(2010), pp. 1–12.