

芦屋大学論叢 第86号

(令和8年7月31日)抜刷

技術科教員養成で行った計測・制御プログラミングにおける
効果的なデバッグ手法の選

岩 城 賢 一
藤 本 光 司
野 口 聡

技術科教員養成で行った 計測・制御プログラミングにおける効果的なデバッグ手法の選定

岩 城 賢 一 (1)

藤 本 光 司 (2)

野 口 聡 (3)

(1) 芦屋大学大学院博士前期課程

(2) 芦屋大学大学院教育学研究科教授

(3) 新見公立大学健康科学部講師

1. はじめに

1.1 研究背景

技術・家庭科（技術分野）（以降は技術科と表記）では2種類のプログラミング教育の題材があり、それぞれ「課題を解決する力の育成」をねらいとしている¹⁾。しかし、アンケート調査²⁾によるとプログラミング指導時に課題として「指導・授業展開の難しさ」、「教員の専門性の不足」が挙がっており、D(2)双方向性のプログラミングでは60%以上、D(3)計測・制御のプログラミングでも50%以上の中学校教員が課題に感じている。また、中学校段階のプログラミングに取り組んだ生徒の理解度において「一部生徒が理解している」、「ほとんどの生徒が理解していない」¹⁾という意見が90%の教員から挙がっている。このことから教員はプログラミングに関して専門知識が不足しているため適切に指導できず、生徒も理解できないため、難しさを感じていると推察する。

プログラミングの難しさについて小泉ら(2017)³⁾は「プログラミングが難しい」と述べた生徒は「センサから得られる情報のバリエーション」や「場面ごとに実行すべきルーチン」が分かっていないと指摘している。これに対して、「分岐処理するときの参照するセンサの現在の状態」や「実行中のプログラムの内容と位置」といった、コンピュータが処理中に見えている情報を把握することが有効と考える。これらの情報が把握できれば、生徒自身が想定している処理と実際にコンピュータが行った処理の違いが分かるはずである。その違いに気が付き、適切にプログラムを修正できれば、コンピュータに期待通りの処理をさせられるため、難しさが低下すると予想した。

コンピュータが見えている情報の把握として、ソフトウェア開発でも用いられるデバッグ手法が有効と考えられる。プログラミングにおいて、間違いを修正する行為をデバッグと呼ぶ。また、そのデバッグを効果的に行うノウハウとしてデバッグ手法がある。その手法を活用することで、プログラミングで躓いたとしても間違いに気が付き、適切に修正することで課題が解決できるはずである。

間違いを修正する方法について、技術科の検定教科書には多くの情報が記載されているが岩城ら(2025)⁴⁾はデバッグする方法の説明がないことを確認している。

そのため、技術科のプログラミング教育においてデバッグの推進とデバッグ方法の提案に向けたデバッグ手法の選定が必要であると考えた。

1.2 研究目的

本研究の目的は中学校のプログラミング教育で実施可能かつ効果的なデバッグ手法の選定である。デバッグ手法はソフトウェア開発において多く存在している。しかし、技術科で許された費用や時間で実施可能なデバッグ手法を選定する必要がある。そこで今回は、技術科のプログラミング教育に向けた予備実験として技術科教員養成コースを受講している大学生に対し、デバッグ手法を活用した計測・制御のプログラミングの授業（以降は本授業と表記）を行うことにした。

本授業の結果から「どのデバッグ手法が最も効果的か」（考察Ⅰ）と「デバッグ手法はプログラミング教育に有効か」（考察Ⅱ）を問い、さらに今後の研究に向けて「技術科にデバッグ手法を適用する際に注意すべきこと」（考察Ⅲ）を問うことにした。

2. 研究方法

2.1 本授業の内容

本授業は計測・制御のプログラミングの学習を目的にしている。そのため、PCでプログラミングするだけでなく、センサやアクチュエータの使い方や、それらの連動について学べるようにした。本授業の計画は表1の通りである。本授業には8名の学生が履修した。

本授業はプログラミング初心者の履修も想定しているため、まずは入門とする座学から始めることにした（1～4時間目）。その後はソフトウェア開発環境を用いて、電気回路の製作とプログラムの制作を行わせた（5～8、11～12時間目）。学習効果の確認として、授業の節目に課題（課題①、課題②、課題③）と試験レポートに取り組ませた。デバッグ手法については活用できる場面で説明した。

表1 授業の計画（全15回）

授業順番	内容（単元）
1～4	プログラミングで登場する用語の説明やソフトウェア開発環境の扱い方を学ぶ（入門用）
5～8	IOポートの扱い方を学ぶ
9～10	課題①、課題②を行う
11	PWM出力を学ぶ
12	AD変換を学ぶ
13	課題③を行う
14～15	試験レポートを行う

2.1.2 用いた教材

計測・制御のプログラミングの学習として電気回路の製作とプログラムの制作が行えるようにEPS 32（マイコン）を搭載した開発ボードとブレットボードを組み合わせた機材を用意した。簡単な電気回路を構築しながら、Thonny（IDE）でMicro Pythonをプログラミングできるソフトウェア開発環境になっている。

2.1.3 授業中の流れ

入門用の授業後（表1の5～8、11～12時間目）は、電気回路の製作とプログラムの制作を行わせた。その授業（90分間）の内、序盤は本時で学ぶ機能の動作原理や使い方を説明した。また、その機能を使うときに活用できるデバッグ手法も説明した。残りの時間は説明した機能の使い方を覚えるために演習問題を出題し、電気回路の製作とプログラムの制作を行わせた。

なお、授業で用いた資料の全ては、学生らにプリントとして配布し、課題に取り組む際の参考情報とした。

2.1.4 課題の内容

本授業では節目に課題に取り組ませた。どれも「以下の動作結果になるように電気回路を実装し、プログラミングすること」とし、求める動作結果は表 2 の通りである。求める動作結果を満たせば、具体的な電気回路とプログラムは自由とし、学生自身に「どうすれば課題が解決できるか」を考えさせた。

表 2 課題で求める動作結果

	求める動作結果
課題①	・スイッチを On すると LED が点滅(On-Off の繰り返し)する ・スイッチを Off すると LED が消灯(Off のみ)する ・何度スイッチを切り替えても上記の処理を行う
課題②	・2つのスイッチを両方 On すると LED が点灯(On)する ・2つのスイッチのいずれかが Off だと LED が消灯(Off)する
課題③	・可変抵抗器のボリュームに応じて LED が点滅する ・LED の点滅周波数は 50 Hz とする ・点滅は消灯（ほぼ）から点灯（ほぼ）まで可変とする

2.2 用いたデバッグ手法

本授業で扱う計測・制御のプログラミングはプログラムと電気回路で構成されており、その連動で動作することから複雑である。バグの原因もプログラムだけでなく、電気回路に起因している可能性があり、特定が難しい。そのため、デバッグ手法においてもソフトウェア用の手法だけでなく、電気回路用の測定器を利用した手法も必要になる。

このような複雑なものをデバッグする場合は、闇雲にするのではなく、ソフトウェアと電気回路ごとにバグがない状態作り、連動を段階的に確認しながら、完成度を高めていくことが大切である。さらに初学者でも「バグを発見さえできればその 97%を修正できる」⁹⁾ と言われていることから、本授業ではバグを探すための方法を用いることにした。

2.2.1 想定するデバッグの流れと確認内容

デバッグの流れとして図 1 の「範囲 S」や「範囲 E」のような狭い範囲から順番に実施し、完成度が高まった後は範囲を広げ（「範囲 S-E」）、最終的な動作結果である「範囲 R」までデバッグを行う。

デバッグを行う各範囲として「範囲 S」はソフトウェア内で完結する処理を指す。これには、四則演算、順次処理、分岐処理、反復処理や print 命令が含まれる。この範囲のデバッグで確認することは、「入力したプログラムが正しく実行しているか」である。

「範囲 E」は電気回路を指す。この範囲のデバッグで確認することは、「マイコンやセンサ、アクチュエータを含む各電子部品が電氣的に正しく接続されているか」。

「センサの状態が変化した場合に電気信号も正しく変化しているか」、「電気信号の変化を起こした際にアクチュエータが正しく動作するか」である。「範囲 S-E」はソフトウェアと電気回路の連動を指す。この範囲のデバッグで確認することは、「センサの状態がソフトウェアで正しく計測できるか」、「ソフトウェアからアクチュエータを正しく制御できるか」である。「範囲 R」は最終的な動作結果を指す。この範囲で確認することは「課題は解決できているか」である。

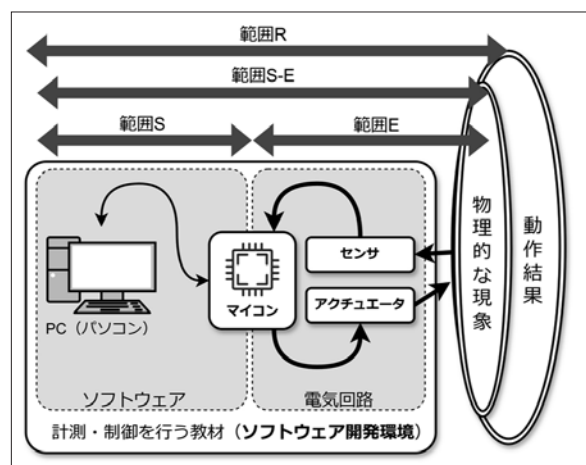


図 1 デバッグを行う範囲

2.2.2 導通テスト（デバッグ手法①）

本手法はデジタルテストの導通チェックモード，もしくは電圧測定モードを用いて，電子部品同士が電氣的に接続されていることを確認する方法である．図 1 の「範囲 E」で使用する．

2.2.3 電圧変化の確認（デバッグ手法②）

本手法はデジタルテストの電圧測定モードを用いて，センサやアクチュエータの電気信号を確認する方法である．図 1 の「範囲 E」，「範囲 S-E」で使用する．センサをデバッグする際は，センサの状態を変化させた後，計測端子の電気信号が意図した状態になっていることを確認する．アクチュエータをデバッグする際は，制御端子に電圧を印加させた後，意図した制御になっていることを確認する．なお，電圧の印加する方法として「範囲 E」では開発ボードの 3 V 3 (3.3 V) 端子または GND (0 V) 端子をアクチュエータの制御端子に接続し，「範囲 S-E」では当該端子を制御するプログラムを制作して行う．

2.2.4 コメントアウト（デバッグ手法③）

本手法はプログラムの一部を無効化することで，試行錯誤を促し，バグの原因を絞り込む方法である．図 1 の「範囲 S」で使用する．この手法はデバッグに限らず，作成中のプログラムを部分的に実行したい場合や分岐処理の条件式を強制的に成立させたい場合にも有効でもある．

2.2.5 print 命令（デバッグ手法④）

本デバッグ手法は print 命令で任意の文字列や変数を，任意のタイミングで IDE に表示することで，プログラムの動作順序を確認する方法であり，江木らのトレース⁶⁾と同様である．図 1 の「範囲 S」，「範囲 S-E」，「範囲 R」で使用する．この手法を用いることで様々な情報を表示できるが，print 命令の使い方を試行錯誤する必要がある．

2.3 収集する情報の内容

本授業で説明したデバッグ手法の感想と課題への取り組み方や躰きについて履修した学生にインタビューと質問紙によるアンケートを行った．収集する情報は表 3 の通りであり，詳細は以降で説明する．なお，履修者が 8 名であり，統計分析が適切に行えない．そのため，質問紙の回答は補足情報とし，インタビューの回答を中心に分析と考察を行った．

情報を収集する場面では本研究が期待している結果に迎合する回答を禁止するため，予め「授業の成績に影響しないので素直に回答して欲しい」と伝えた．また，回答結果の扱いについても「本研究のみに使用する」と伝えてから実施している．

表 3 収集する情報一覧とタイミング

収集する情報の内容	収集方法	収集時期
デバッグ手法の感想	半構造化 インタビュー	全課題（①～③）終了後
課題ごとの躰き	質問紙	課題①，課題②，課題③の終了時
手法ごとの頻度と効果	質問紙	課題①，課題②，課題③の終了時
手法ごとの順位	質問紙	課題①，課題②，課題③の終了時

2.3.1 インタビュー「デバッグ手法の感想」の内容

このインタビューにより「デバッグ手法は課題の解決に寄与したか」を把握する。質問項目は表 4 の通りであり、一人当たり 15 分程度行った。各質問項目の目的として「質イ 01」では、この授業の全体を通じてデバッグを必要とする場面（躓き）があったかを確認する。「質イ 02」では、課題に取り組んだ全体の傾向として、デバッグ手法の活用の有無とその理由を確認する。「質イ 03～質イ 08」では、課題ごとの回路製作とプログラミングで、各デバッグ手法の使用の有無とその理由を確認する。

インタビューの回答内容はボイスレコーダーで録音し、文字起こし後、逐語録を作成し、M-GTA 7) を行った。M-GTA は「人間の行動の何らかの変化と多様性の説明」に適しているため、デバッグ手法を学んだ影響を確認できると考えた。なお、分析テーマは「本授業の課題に対して学生はどのように対処したか」とし、分析焦点者は「本授業を履修し、課題①～③まで取り組んだ学生」とした。分析の手順として、逐語録から分析テーマに関係する内容を解釈し、要約である一次コードを設定した。全員分の逐語録に一次コードを設定した後は、一次コードごとの主旨を確認し、類似と対極を区別して二次コードを設定した。

概念は「全ての二次コードは課題解決に至る作業工程や判断に関係する」という観点で二次コードを分類し、生成した。その分類した二次コードの主旨を包括するように概念名と定義を設定した。各概念の関係性についても同じ観点で整理し、作業者の思考プロセスに合うよう、矢印で時系列を表現した。カテゴリは概念が意味する特徴と課題の取り組み中に起こった場面对応付けしてまとめ、包括するように名称を設定した。

表 4 「デバッグ手法の感想」の質問項目

番号	質問項目
質イ 01	この授業で上手くいかないことはありましたか？どんなところで躓きましたか。
質イ 02	試行錯誤を促すためにデバッグ手法を用いることを想定していましたが、利用しましたか？
質イ 03	課題①で回路製作を行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？
質イ 04	課題②で回路製作を行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？
質イ 05	課題②で回路製作を行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？
質イ 06	課題②でプログラミングを行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？
質イ 07	課題③で回路製作を行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？
質イ 08	課題③プログラミングを行いました。その中でどのデバッグ手法が良かったですか？ また、効果のないデバッグ手法はありましたか？

2.3.2 質問紙 1 「課題ごとの躓き」の内容

この質問紙により「実施した課題ごとに、どのような躓きが発生したのか」を把握する。質問項目は安達ら(1995)⁸⁾の質問紙から“つまずきの箇所”を引用した。この質問紙で想定している躓きは抽象化プロセス 4 項目 (1～4)、構造化プロセス 5 項目 (5～9)、その他 (主に操作段階) 3 項目 (10～12) であり、回答者には上位 3 位まで選択させた。選択肢は表 5 の通りである。なお、回答は第 1 位に 3 点、第 2 位に 2 点、第 3 位に 1 点の重みづけをし、選択肢ごとに得点化して分析した。

表 5「課題ごとの躓き」の選択肢

選択肢	
1. 例題や課題の意味（何を要求されているのか）がわからない	7. どのステートメント（命令語）を使えばよいかわからない
2. 何を入力として与えればよいかわからない	8. ステートメントの書き方がわからない
3. 何を結果として出力すればよいかわからない	9. ステートメントをどういう順番に並べればよいかわからない
4. 考え（どのような方法で解決すればよいか）がわからない	10. プログラムが正しく動くかどのように調べたらよいかわからない
5. どのような変数を使えばよいかわからない	11. プログラムのキーボードから入力操作がよくわからない
6. 手順（どのような順序で処理すればよいか）がわからない	12. プログラムの保存、読み込みの方法がよくわからない

2.3.3 質問紙 2「手法ごとの使用頻度と効果」の内容

この質問紙により「どのデバッグ手法は効果的か」を把握する。回答は 4 件法（4：とても当てはまる～1：全く当てはまらない）とした。質問項目は表 6 の通りである。

表 6「手法ごとの使用頻度と効果」の質問項目

質問項目	
デバッグ手法①「テストを使った導通テスト」について教えてください	
質問紙 2-01	頻繁に使用しましたか
質問紙 2-02	効果的でしたか
デバッグ手法②「テストを使った電圧変化の確認」について教えてください	
質問紙 2-03	頻繁に使用しましたか
質問紙 2-04	効果的でしたか
デバッグ手法③「コメントアウトによる簡易的な削除」について教えてください	
質問紙 2-05	頻繁に使用しましたか
質問紙 2-06	効果的でしたか
デバッグ手法④「欲しい情報を print 命令で確認」について教えてください	
質問紙 2-07	頻繁に使用しましたか
質問紙 2-08	効果的でしたか

2.3.4 質問紙 3「手法ごとの順位」の内容

この質問紙により「どのデバッグ手法が最も効果的だったか」を把握する。回答者には課題に取り組んだときに役に立ったデバッグ手法を上位 3 位まで選択させた。選択肢は表 7 の通りである。なお、回答は第 1 位に 3 点、第 2 位に 2 点、第 3 位に 1 点の重みづけをし、選択肢ごとに得点化して分析した。

表 7「手法ごとの順位」の選択肢

選択肢
①テストを使った導通テスト
②テストを使った電圧変化の確認
③コメントアウトによる簡易的な削除
④欲しい情報を print 命令で確認

3. 研究結果

3.1 収集した情報の結果と分析

3.1.1 インタビュー「デバッグ手法の感想」の内容

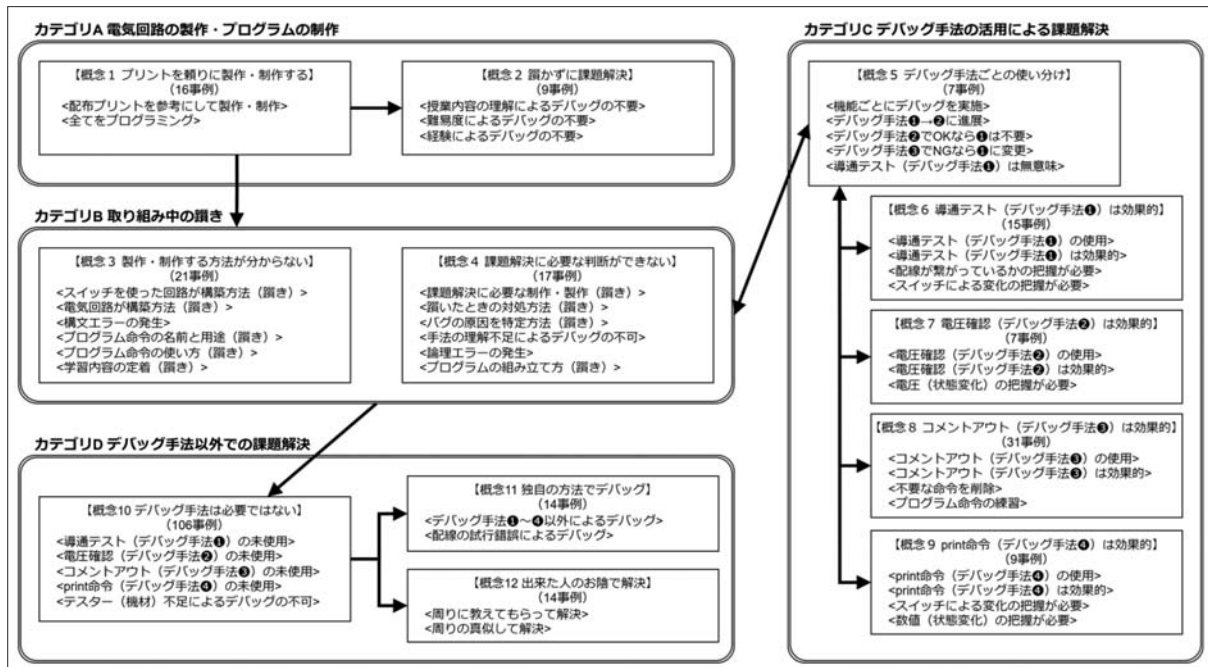


図2「デバッグ手法の感想」で生成されたM-GTAによる結果図

M-GTAの結果、分析テーマ「本授業の課題に対して学生はどのように対処したか」の思考プロセスを図2に示す。図中の< >は二次コード，【 】は概念，A～Dはカテゴリを意味する。それぞれ、46の二次コード、12の概念、4つのカテゴリで構成されている。概念の定義と具体例および事例数については表8の通りである。なお、デバッグ手法ごとの使用または未使用の二次コードについては回答内容がYesかNoの2択であるため、表8に具体例を記載していない。インタビューの属性として表4の「質イ02」より、デバッグ手法を活用した5名（表8のA～E）と活用していない3名（表8のF～H）に分かれた。

ストーリーラインとして、学生らは課題に取り組むときに、まず「カテゴリA 電気回路の製作・プログラムの制作」を行うが、授業で配布された【概念1 プリントを頼りに製作・制作する】。その製作・制作で課題が解決できれば、【概念2 躓かずに課題解決】になる。このパターンは課題解決に必要な力量が課題開始前に備わっており、デバッグを必要としなかった。

しかし、何かしらで、「カテゴリB 取り組み中の躓き」が起き、【概念3 製作・制作する方法が分からない】か【概念4 課題解決に必要な判断ができない】に陥った。どちらに躓くかは課題の内容や個人の力量により異なるため、一定ではない。躓いた後の対応として「カテゴリC デバッグ手法の活用による課題解決」または「カテゴリD デバッグ手法以外での課題解決」に分かれていた。

「カテゴリC デバッグ手法の活用による課題解決」では、躓いている状況に応じて【概念5 デバッグ手法ごとの使い分け】をしていた。1種類のデバッグ手法を用いて課題を解決した場合もあれば、複数のデバッグ手法を用いたなど、活用の仕方は多様である。また、デバッグ手法は【概念6 導通テスト（デバッグ手法①）は効果的】、【概念7 電圧確認（デバッグ手法②）は効果的】、【概念8 コメントアウト（デバッグ手法③）は効果的】、【概念9 print命令（デバッグ手法④）は効果的】の通り、効果的であった。この

パターンは課題で躰いたものの、デバッグ手法を用いたことで課題を解決している。しかし、＜導通テスト（デバッグ手法①）は無意味＞のようにデバッグ手法を用いたものの課題解決に至らない事例もあった。

一方で「カテゴリ D デバッグ手法以外での課題解決」として【概念 11 独自の方法でデバッグ】または【概念 12 出来る人に助けてもらって解決】に分かれていた。このパターンも課題で躰いたものの、何かしらの手段を用いて課題を解決していた。

表 8 「デバッグ手法の感想」で生成された概念の定義と具体例

概念	定義	具体例	事例数
1	プリントを頼りに製作・制作する	授業で配布されたプリントを組み合わせて一通り製作・制作した。プリント（演習問題などの教材）あつまりまくって、くっつけた（組み合わせた） B 100 一旦こう思ったことを「だー」って書いて（プログラミングして）みて E 45	16
2	躰かずに課題解決	課題の難易度よりも学生の力量が高いため、躰くことがなかった。今回はできてる気がするんですよ B 98 授業の時にまあ、分かりやすく説明してもらってたんで D 91	9
3	製作・制作する方法が分からない	電気回路やプログラムの製作・制作の仕方が分かっていない状態。もしくは動作しない状態。下のウインドウ（Thonny のシェル）で不都合（構文エラー）を教えてくれる A 52 「どの関数を使えばいいのかわかるのか」っていうのを、授業ごとに忘れちゃったりして、思い出すのとか C 28	21
4	課題解決に必要な判断ができない	課題に取り組む過程でどのように製作・制作すれば解決できるか判断ができない状態。電氣的に?がってないから動かないのか?そのプログラミングが間違っているのかっていう判断が、あまり自分の中でできてなかった F 31 何からすればいいかわからない H 35	17
5	デバッグ手法ごとの使い分け	デバッグ手法ごとに適した場面が異なる。使い分けて活用されていた。見たいところ以外を一旦（コメントアウトして）切り離して A 45 やはりコメントアウト（デバッグ手法③）。それでも、「ちょっとあれ？」ってなった時は導通テスト（デバッグ手法②）とか C 105-107	7
6	導通テスト（デバッグ手法①）は効果的	電気回路製作において配線が繋がっているかを確認するときに活用されていた。導通テストを使うとスイッチを切り替えたときに、回路がどう繋がるか分かる。 B 82-92 音が鳴るとかが、すごいわかりやすくて D 75	15
7	電圧確認（デバッグ手法②）は効果的	電気回路製作において可変抵抗などの電圧値を確認する必要がある場合に活用されていた。数字（電圧値）が変わるのが出てくれたんで、わかりやすかった E 168-170	7
8	コメントアウト（デバッグ手法③）は効果的	プログラミングの全般で活用されていた。ワンタッチかツータッチぐらいで消したいところを消して、見たいところ見るみたいなのができる A 103 他の関数を使うときとか、試してみるとときとか C 162	31
9	print 命令（デバッグ手法④）は効果的	スイッチや可変抵抗の状態を把握する必要がある場合に活用されていた。センサ類が混ざってくる数値を見れたほうがやすい A 99 スイッチの状態が分かるようにしています B 124	9
10	デバッグ手法は必要ではない	課題の解決にデバッグ手法は必要ではない。※各デバッグ手法の未使用に関する具体例は省略。 使うかなと思った時に、ちょっと他の人が（テスターを）使ってた A 47	106
11	独自の方法でデバッグ	授業で学んだデバッグ手法以外の方法でデバッグして課題を解決した。（ジャンパー線をブレッドボードのどこかの穴に）挿してダメだったら、もう他の（場所）を挿してみたいな感じで A 23 自分で工夫したりもしてるし C 142	14
12	出来る人のお陰で解決	課題を解決している人から教えてもらったり、その成果物を真似るなどして自分の課題を解決していた。その友達の（製作した電気回路）を見てたから回路は間違ってたなと F 90-92 「こうしてみ？」みたいな言われて G 35	14

3.1.2 質問紙1「課題ごとの躓き」の結果と分析

本授業の履修者が8名に対し、回答者数が課題①で4名、課題②で3名、課題③で5名と十分な回答数ではないが、順位（表9）を求めた。

その結果、第1位が「7. どのステートメント（命令語）を使えばよいかわからない」だった。なお、操作段階（選択肢10～12）については順位が低いため、躓いていないと考えられる。

表9 「課題ごとの躓き」の順位

選択肢	課題①	課題②	課題③	総合
1. 例題や課題の意味（何を要求されているのか）がわからない	3	3	4	3
2. 何を入力として与えればよいかわからない	8	6	2	6
3. 何を結果として出力すればよいかわからない	8	6	4	9
4. 考え（どのような方法で解決すればよいか）がわからない	3	1	1	2
5. どのような変数を使えばよいかわからない	2	3	9	3
6. 手順（どのような順序で処理すればよいか）がわからない	3	3	4	3
7. どのステートメント（命令語）を使えばよいかわからない	1	1	2	1
8. ステートメントの書き方がわからない	3	6	4	6
9. ステートメントをどういう順番に並べればよいかわからない	3	6	4	6
10. プログラムが正しく動くかどのように調べたらよいかわからない	8	6	9	10
11. プログラムのキーボードから入力操作がよくわからない	8	6	9	10
12. プログラムの保存、読み込みの方法がよくわからない	8	6	9	10

3.1.3 質問紙2「手法ごとの使用頻度と効果」の結果と分析

本授業の履修者が8名に対し、回答者数が課題①で4名、課題②で3名、課題③で5名と十分な回答数ではないが、平均値（表10）を求めた。

その結果、全体としてデバッグ手法の「頻繁に使用しましたか」が中間値(2.50)を超えている場合は「効果的でしたか」も中間値を超えていた。そのため、使用していたデバッグ手法の効果を感じていた可能性がある。しかし、課題②のデバッグ手法①とデバッグ手法②の「頻繁に使用しましたか」の平均が中間値よりも低かった。理由として課題②は、課題①と同日に行ったため、課題①で既に出来上がった電気回路を改造することで容易に完成でき、デバッグが不要になったと考えられる。

表10 「手法ごとの使用頻度と効果」の平均

質問内容	課題①	課題②	課題③
デバッグ手法①「テストを使った導通テスト」について教えてください。			
質紙2-01 頻繁に使用しましたか	2.50	1.67	3.00
質紙2-02 効果的でしたか	3.00	2.33	3.40
デバッグ手法②「テストを使った電圧変化の確認」について教えてください。			
質紙2-03 頻繁に使用しましたか	2.25	1.67	3.00
質紙2-04 効果的でしたか	3.00	2.33	3.20
デバッグ手法③「コメントアウトによる簡易的な削除」について教えてください。			
質紙2-05 頻繁に使用しましたか	3.25	3.33	2.80
質紙2-06 効果的でしたか	3.50	3.33	2.80
デバッグ手法④「欲しい情報を print 命令で確認」について教えてください。			
質紙2-07 頻繁に使用しましたか	3.00	2.67	2.20
質紙2-08 効果的でしたか	3.25	3.00	2.20

3.1.4 質問紙3「手法ごとの順位」の結果と分析

本授業の履修者が8名に対し、回答者数が課題①で4名、課題②で3名、課題③で5名と十分な回答数ではないが、順位（表11）を求めた。その結果、全ての課題でコメントアウト（デバッグ手法③）が第1位だった。

表11 「手法ごとの順位」の結果

選択肢	課題①	課題②	課題③	総合
①テストを使った導通テスト	2	3	2	2
②テストを使った電圧変化の確認	4	4	2	4
③コメントアウトによる簡易的な削除	1	1	1	1
④欲しい情報を print 命令で確認	3	2	2	2

3.2 本研究の結果と考察

3.2.1 考察Ⅰ「どのデバッグ手法が最も効果的か」

「どのデバッグ手法が最も効果的か」について、本授業ではコメントアウト（デバッグ手法③）が最も効果的なデバッグ手法だった。その根拠として表8より【概念8 コメントアウト（デバッグ手法③）は効果的】の事例数が最も多く、表11でも第1位であることから判断した。コメントアウトが最も効果的になった理由として、表9より「9. どのステートメント（命令語）を使えばよいかわからない」が第1位であったことや、表8より【概念3 製作・制作する方法が分からない】の具体例に「下のウインドウ（Thonnyのシェル）で不都合（構文エラー）を教えてくれる」があり、構文エラーが頻発していることが要因と考えられる。

以上より、構文エラーが頻発している状況では、コメントアウト（デバッグ手法③）が最も効果的なデバッグ手法だったと言える。

3.2.2 考察Ⅱ「デバッグ手法はプログラミング教育に有効か」

「デバッグ手法はプログラミング教育に有効か」については有効であったと考えられる。その根拠として図2および表8に【概念7 導通テスト（デバッグ手法①）は効果的】、【概念7 電圧確認（デバッグ手法②）は効果的】、【概念8 コメントアウト（デバッグ手法③）は効果的】、【概念9 print 命令（デバッグ手法④）は効果的】があり、デバッグ手法を活用して課題を解決していたことが確認できたからである。また、表10より「頻繁に使用しましたか？」の平均が中間値を超えていた場合は「効果的でしたか？」も中間値を超えていたことから示唆される。しかし、表10の課題②でデバッグ手法①とデバッグ手法②の「頻繁に使用しましたか？」が中間値よりも低いことや、考察Ⅰの「構文エラーが頻発している状況ではコメントアウト（デバッグ手法③）が最も効果的なデバッグ手法だったと言える。」より、全てのデバッグ手法が、どの場面でも効果的という訳ではなく、発生した躓きに対して効果的と思われるデバッグ手法が活用されると考えられる。

以上より、授業で制作させる内容や制作中に想定される躓きに、効果的なデバッグ手法を説明することが大切である。

3.2.3 考察Ⅲ「技術科でデバッグ手法を適用する際に注意すべきこと」

「技術科でデバッグ手法を適用する際に注意すべきこと」としては表8より【概念8 コメントアウト（デバッグ手法③）は効果的】の具体例「ワンタッチかツータッチぐらいで消したいところを消して、見たいところを見るみたいなのができる」より、簡単に利用できる利便性の高いデバッグ手法が望ましいと考えられる。

特にデバッグ手法で用いる機材に台数制限があると【概念 10 デバッグ手法は必要ではない】の具体例「使うかなと思った時に、ちょっと他の人が（テスターを）使っていて」のようにデバッグ手法が利用できないことがあり得るため、嫌厭される可能性がある。

4. まとめ

4.1 結論

本研究の目的は中学校のプログラミング教育で実施可能かつ効果的なデバッグ手法の選定である。今回はその予備実験として、技術科教員養成コースの大学生に対して授業を行い、デバッグ手法の評価を行った。その結果、プログラミング教育で実施可能なデバッグ手法であっても、学習者に制作させる内容や制作中に発生した躓きに対して効果的であれば活用されることが確認できた。

なお、本授業のような計測・制御のプログラミングではプログラムの制作と電気回路の製作を行うため、それぞれに対応したデバッグ手法が必要になる。

4.2 今後の課題

本授業では履修した学生が少なく、またアンケートの回答率も低いことから適切に分析が行えない場面があった。そのため、情報収集が十分に行える環境で授業を実施することが課題である。

また、本授業で発生した躓きの多くは構文エラーであった。構文エラーは AI による自動修正が可能⁹⁾ ため、AI によるデバッグの推進や、構文エラーが発生しないブロック型のプログラミング言語の採用も検討したい。

今後は技術科でデバッグ手法を活用したプログラミング教育を行う。その際には技術科の授業時数に合わせた授業設計を行い、利便性が高く、予算に見合ったデバッグ手法を選定して、効果を評価したい。

参考文献

- 1) 文部科学省, “【技術・家庭編】中学校学習指導要領（平成 29 年告示）解説,” 2017, pp.23,48-59, 152.
- 2) 特定非営利活動法人みんなのコード, “全国の学校教育における「2022 年度 プログラミング教育・高校「情報 I」実態調査」を公開,” 9 8 2023. オンライン). Available: <https://code.or.jp/news/20230809/>.
- 3) 小泉匡弘・藤川聡・渡壁誠, “自律型ロボット教材とした「プログラムによる計測・制御」に関する学習の教育的効果及び難しさの要因,” 日本教材学会, 第 28 巻, pp.59-68, 2017.
- 4) 岩城賢一・藤本光司・野口聡, “中学校技術科のプログラミング教育で登場するデバッグ手法の教科書調査,” 芦屋大学論叢, 第 83 巻, pp.73-80, 2025.
- 5) S. F. L. M. M. S. T. Zander, “Debugging : finding, fixing and flailing, a multi-institutional study of novice debuggers,” *Computer Science Education*, 第 18 巻, 第 2 号, pp.93-116, 2008.
- 6) 江木鶴子・竹内章, “プログラミング初心者にはトレースを指導するデバッグ支援システムの開発と評価,” 日本教育工学会論文誌, 第 32 巻, 第 4 号, pp.369-381, 2009.
- 7) 木下康仁, “修正版グラウンデッド・セオリー・アプローチ (M-GTA) の分析技法,” 富山大学看護学会誌, 第 6 巻, 第 2 号, pp. 1-10, 2007.
- 8) 安達一寿・中尾茂子, “プログラミング学習における学生のつまづき箇所の分析,” 日本教育情報学会, 第 10 巻, 第 4 号, pp.11-20, 1995.
- 9) 宇都宮魁斗・秋山楽登・近藤将成・亀井靖高・鶴林尚靖, “初学者のためのデバッグ作業支援に向けた大規模言語モデルの初期評価,” ソフトウェア工学の基礎ワークショップ論文集, pp.63-71, 2023.

— Abstract —

Selection of Effective Debugging Methods for Measurement and Control Programming in
Technology Teacher Training

IWAKI Kenichi (1) FUJIMOTO Koji (2) NOGUCHI Satoshi (3)

There are difficulties in teaching programming in technology and home economics (technology field). We hypothesized that these difficulties could be reduced by using debugging techniques. Therefore, in this study, we conducted a programming lesson on measurement and control using four types of debugging techniques with eight university students in a technology teacher training course. The results suggested that debugging techniques are effective in programming education. However, it was found that not all techniques were effective in all situations, and the most effective technique varied depending on the content of the project to be created in the lesson and the problems encountered.