

プログラミング教育における 剰余演算子活用能力を高める ドリル型記述演習問題の導入

三浦 元喜^{1,a)}

受付日 2019年5月31日, 採録日 2019年0月0日

概要: プログラミングにおいて、剰余演算は単に余りを求める演算というだけではなく、奇数・偶数判定や倍数判定、整数の桁を抽出するなど、さまざまな目的で利用されている。条件分岐を減らし、簡潔で効率的なプログラムを書くためには、剰余演算がどのような場面に適用できるかについての理解が求められる。しかし、プログラミングの初心者は、プログラミングにおいて剰余演算をどのように活用すればよいかの知識や経験に乏しい。本研究では、プログラミング初心者が剰余演算についての理解を深め、活用できるようにするためのドリル型演習教材を提案する。アンケートの結果より、条件分岐に頼らず演算子のみで解決可能な問題の範囲を示せることが確認できた。

キーワード: プログラミング教育, ブラックボックス問題, 汎化能力, 整数演算, Many Small Programs

Introduction of Drill Type Practice Problem which Raises the Ability to Use Remainder Operation for Programming Lectures

MOTOKI MIURA^{1,a)}

Received: May 31, 2019, Accepted: xx 0, 2019

Abstract: In programming, the modulo operation is used not only for calculating the remainder but also for various purposes such as odd / even judgment, multiple judgment, and extraction of integer digits. In order to write a concise and efficient program, it is necessary to understand what kind of situations where modular arithmetic can be applied. However, novice programmers have little knowledge of how to use modular arithmetic in programming. In this research, we propose a drill-type exercise material to enable programming beginners to understand and use residual arithmetic. From the results of the questionnaire, it was confirmed that the range of problems that can be solved only by the operators, without relying on conditional branching, can be shown.

Keywords: Programming Education, Black Box Problem, Generalization Ability, Integer Arithmetic, Many Small Programs

1. はじめに

大学生を対象とするプログラミング教育においては、条件分岐、繰り返しといったプログラミング特有の構文や、変数や配列、構造体といったデータ構造の理解と、それら

を活用した問題解決能力が重要視される。しかし、プログラミング初心者にとっては、条件分岐や繰り返しといった概念はもとより、変数や演算子がどのように機能し、活用できるかを理解することが難しい場合がある [1], [2]。また、プログラミング特有の表記・文法に習熟し、典型的な問題解決の経験を有する熟達者と、プログラミング初心者との違いとして、問題解決に資する系統的に整理された知

¹ 九州工業大学 基礎科学研究系
1-1 Sensui, Tobata, Kitakyushu, Fukuoka, 804-8550, Japan
^{a)} miuramo@mns.kyutech.ac.jp

識の量 [3] が指摘されている。熟達者は、プログラミング言語が標準的に提供する演算機能やその使い方を把握しているが、プログラミング初心者はそのような系統的な知識を持っておらず、また演算機能についてどのように活用できるのかの理解が不足していると考えられる。

そこで我々は、プログラミング初心者が、プログラミング特有の表記・文法と、それらの働きとの対応関係について早い段階で慣れることができるようにするため、整数演算と演算子に特化した初歩的なドリル型演習問題を作成した。特に、プログラミングにおいて頻繁に利用される剰余演算子の使い所や活用事例について、段階的・網羅的に習得できることと、学習者の帰納的推論や汎化思考を促す点に特徴がある。

2. 関連研究

土肥らは、認知心理学に基づいた学習理論によるシステムティックな情報教育メソッド SIEM (Systematical Information Education Method) を構築し、Java 言語入門教育に適用している [4]。SIEM では、系列位置効果に基づいた授業構成や、発見学習の要素を持った枠組み、スモールステップの導入、即時フィードバックなどが提唱されており、本研究実践における内容との親和性が高い。本研究実践では、剰余を含む演算子活用能力を高めることに着目し、分析を行っている点が異なる。

変数や剰余を含む演算子の理解に着目したプログラミング学習として、以下の研究がある。松本らは、C 言語読解学習のための、演算や代入を含むプログラム断片問題を自動生成するシステムを実講義で運用し、評価している [5]。学習者は、システムが生成したプログラム断片をトレースし、実行後の変数の値を記述や、多岐選択によって回答する。アンケート結果によると、上位群の学習者は特に剰余演算子や複合代入演算 ($+=$, $-=$ など) の知識と読解技能の向上効果を感じていた一方、下位群の学習者による読解学習への期待度は低かった。この原因として、自動生成問題が下位群の学習者にとって難易度が高く理解が困難であったことや、自動生成問題の妥当性について疑問を持たれていた可能性がある。本研究との主な相違点は問題の出題方式 (読解/類推と記述) および生成方式 (自動生成の有無) である。本研究では問題の難易度とその配列をあらかじめ設定したが、内容と難易度を考慮した問題の自動生成が可能になれば、多数の問題が提供できるため、学習過程における網羅性は高まると考えられる。

3. 剰余を含む演算子活用能力を高めるドリル型記述演習問題

我々が作成したドリル型演習問題は、中学 1 年生の数学における「ブラックボックスの働き」を予想しながら関数の意味を確認する学習実践 [6] に類似している。ブラック

ボックス (関数・函数) の入力値と出力値の関係を読み取り、ブラックボックスの処理を推測したうえで、そのような計算を行うプログラムを記述して回答するものである。例として、以下に最初の問題を示す。

【問題 1】

以下の例は、関数 $f(x)$ の入力と出力を、あらわしています。
整数 x を 1 つ入力したら、この関数 $f(x)$ の値を出力するプログラムを作成してください。
(具体例から一般化して、関数を推測してください)

$f(2) = 4$ $f(3) = 9$ $f(5) = 25$

学習者は、以下の制約

- 演算子 (加減乗除および、剰余) と、括弧のみを用いて記述する。
- 条件文 (if) や、三項演算子は使用しない。

のもとで、教授者が提供する雛形プログラムの一部を直接書き換えることによって回答する。図 1 に、Web インタフェースによる回答画面を示す。学習者は、プログラムの 5 行目の演算のみに着目して編集することが期待されている。学習者がプログラムを保存すると、正解かどうかをチェックすることができる。これにより、正解であれば次の問題にすすみ、間違っていたら再度検討することができる。

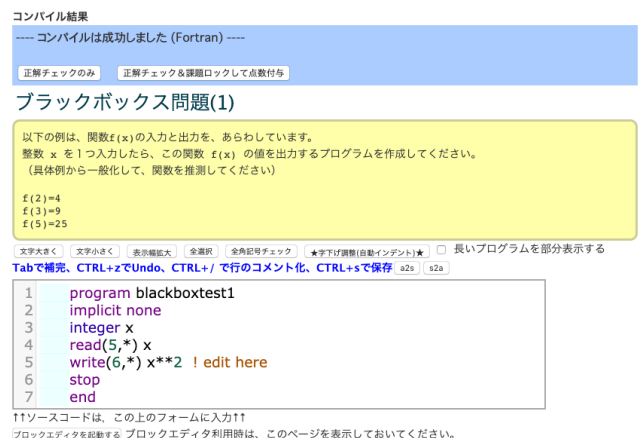


図 1 Web インタフェースによる回答画面

3.1 演習問題作成にあたっての基本方針

付録 A.1 に、今回作成したブラックボックス問題を示す。それぞれの問題は、以下に示す出題意図に基づいて作成している。

- (1) 【問題 1】 乗算またはべき乗の演算子による、2 乗の計算方法の確認。
- (2) 【問題 2】 減算の確認。
- (3) 【問題 3】 剰余演算子の基本的な働きの確認。

- (4) 【問題 4】 入力値が偶数のとき 0、奇数のとき入力値そのまま、となるように、剰余演算の結果を用いた計算方法を思いつけるかどうかの確認。
- (5) 【問題 5】 括弧と減算を用いて、0 / 1 を反転できることの確認。問題 4 では、剰余演算の結果をそのまま用いればよいが、問題 5 では 0/1 の反転が必要となる。
- (6) 【問題 6】 剰余演算子を用いて、整数の一の位の数を取り出せることの確認。
- (7) 【問題 7】 整数型同士の除算では、小数点以下が切り捨てられることの確認。
- (8) 【問題 8】 問題 6 における「整数の一の位の数」の取り出しと、除算による十の位の数を取り出しを個別に行えるかどうかの確認。
- (9) 【問題 9】 剰余演算結果を用いて、奇数のとき-1、偶数のとき 1 を作り出せるかどうかの確認。
- (10) 【問題 10】 問題 9 の奇偶判定を反転させる演算を作り出せるかどうかの確認。
- (11) 【問題 11】 問題 8 の発展（加算から乗算）。
- (12) 【問題 12】 3 で割ったときの剰余を用いて、0/1/-1 を作り出せるかどうかの確認。

これらの問題に順次取り組んでいくことによって、プログラミング学習における整数型変数や演算の特徴や注意点、加減乗除演算および剰余演算の具体的な適用例を学習の初期の段階で、経験的・網羅的に習得することをねらいとしている。

なお、この演習課題は、基本的な四則演算と剰余演算、括弧のみで構成されているため、多くは学習者の「数学」に関する既有知識に依拠している。変数と整数型演算の知識のみで取り組めるため、条件分岐や繰り返しの前段階において、基本的な理解の確認テストとして利用できる。

また、「ブラックボックスの働きを推測し、仮説をつくって、確認していく」という学習行為は、「不慣れなプログラムの記述と、その具体的な動きとを対応づけて、理解していく」というプログラミング学習における活動にも通じるものがある。さらに、想定されている多様な入力に対して共通の記述で対応できる処理を考えるという点で、汎化の概念も扱うことになる。そのため、初学者がプログラミング学習の初期の段階で取り組む際に望まれる意識や習慣付けという観点からも、意義があると考えている。

剰余演算の活用例については、通常のプログラミング演習のなかで、逐次導入していくことも可能ではある。しかし、プログラミング初学者の剰余演算に関する既有知識が乏しい場合、演習で求められる「発想」まで一気に着想することは難しい。Allen らは、一つの大きな問題 (One Large Program: OLP) よりも、細分化した多くの小さな問題 (Many Small Program: MSP) のほうが、学生のストレスを軽減し、自信と満足度を高めることを示している [7], [8]。我々も、プログラム学習におけるスモールス

テップの原則に則り、演算子のみの演習によって、段階的に学習をすすめていくほうが望ましいと考えている。また、演算子のみで解決可能な「問題の範囲」を先に例示することによって、通常のプログラミング演習において、学習者が「条件分岐が真に必要な場面かどうか」や「条件分岐の利用が妥当かどうか」、「繰り返し処理をどう記述すればよいか」を検討しやすくなると考えている。

4. 授業実践

ブラックボックス問題への学習者の取り組み状況や正答率と、講義における得点との関連を調査するため、授業実践を行った。実践 1 と実践 2 の違いは、(1) 前者は試験の点数を含めた分析をしている、(2) 後者はアンケートを実施し、アンケート回答を含めた分析を行っている、の 2 点である。

4.1 実践 1

2018 年度後期に 3 クラスで開講している、Fortran プログラミングと数値計算法に関する講義において、本研究において提案しているブラックボックス問題を適用した。初回の講義 (10 月 3~4 日) で、整数型の変数、四則演算、べき乗、剰余について説明したあと、ブラックボックス問題 (1)~(10) を 1 問 0.2 点の演習形式で出題した。締切日時は 10 月 12 日夜に設定した。学習者は図 1 で示した Web インタフェースで、締め切りまでいつでも回答できる状態であった。各学習者の、ブラックボックス問題の点数 (0~2)、ブラックボックス問題を保存した回数、小テストの点数、演習の点数、中間試験の点数、期末試験の点数を記録した。小テストは講義の最初の 10 分間で、相談しながら回答可能な選択問題である。演習は講義の後半や授業時間外に取り組むプログラミング課題である。

4.2 実践 1 の結果

図 2~図 4 に、ブラックボックス問題点数・保存回数と、その他の点数のヒストグラム (対角成分)、散布図 (左下)、および相関 (右上) をクラスごとに示す。なお、30 点未満の学生については、履修を途中で取りやめていたため分析からは除外した。その結果、クラス A,B,C の受講人数はそれぞれ 89, 84, 99 となった。相関に付与された赤字のドットおよび*,**,***は、それぞれ p 値が 10%, 5%, 1%, 0.1% 以下であることをあらわしている*¹。合計点数の等分散性の検定の結果、分散が異なっていたことと、中間試験/期末試験の難易度に差があったため、クラスごとに分析している。図 2 をみると、クラス A のブラックボックス問題 (BlackBox) と、保存回数 (BBSave) に、弱い正の相関 (0.28) が確認できた。クラス A に関しては、プロ

*¹ <https://github.com/R-Finance/PerformanceAnalytics/blob/master/R/chart.Correlation.R>

グラムを何度も修正し保存している学生のほうが、ブラックボックス問題の正答率が高かったといえる。そのほかのクラスに関しては、ブラックボックス問題の正答率が全体的に高いことから、成績との有意な相関はほとんどみられなかった。例外として、クラスBのブラックボックス問題(BlackBox)と、小テスト(Mini)との有意な相関(0.38)がみられた。ブラックボックス問題、小テスト、演習(Exe)は、周囲と相談しながら回答可能という点で共通していることから、回答時の学生の相談行動が影響していると考えられる。

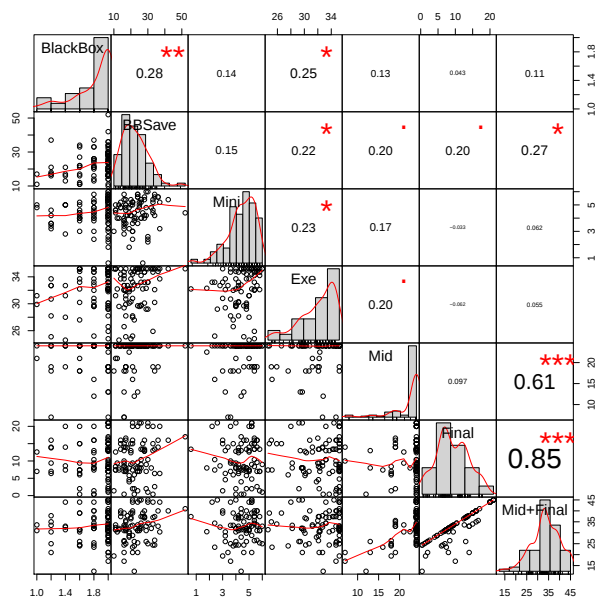


図2 ブラックボックス問題点数・保存回数と、その他点数との相関 (クラスA, N=89)

表1に、ブラックボックス問題の平均点を、クラスごとに示す。すべてのクラスにおいて、問題5がもっとも正答率が低かった。これは、剰余値0/1を、1から減ずることによって1/0に変換する技巧的な手法の難易度が高かったことが伺える。また、すべてのクラスにおいて、その次に正答率が低かったのは、問題4であった。問題3と問題4のあいだに、より単純な剰余による偶奇を0/1で返す問題を設定する必要があったと考えられる。

4.3 実践2

2019年度前期に1クラスで開講している、ProcessingとC言語によるプログラミング講義において、ブラックボックス問題(1)~(12)を1問0.1点として導入した。実践2では、初回講義(4月10日)で問題(1)~(5)、2回目講義(4月17日)で問題(6)~(10)、3回目講義(4月24日)で問題(11),(12)というように、分割して実施した。問題(1)~(10)は演習形式、(11),(12)は制限時間10分内で回答する試験形式をとった。プログラミング言語はC言語を用い

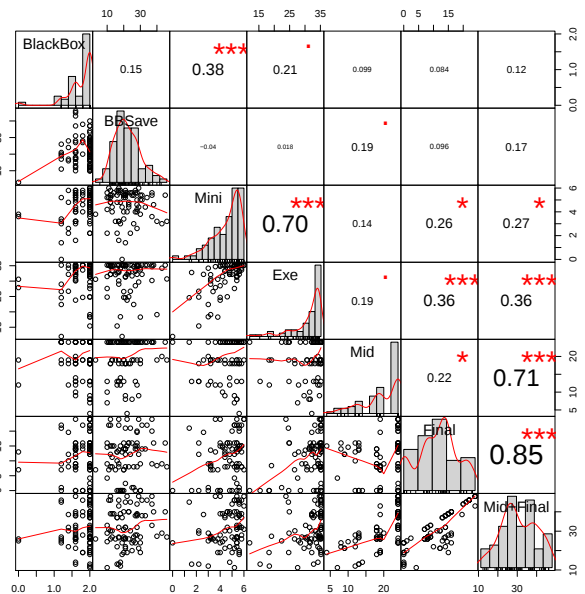


図3 ブラックボックス問題点数・保存回数と、その他点数との相関 (クラスB, N=84)

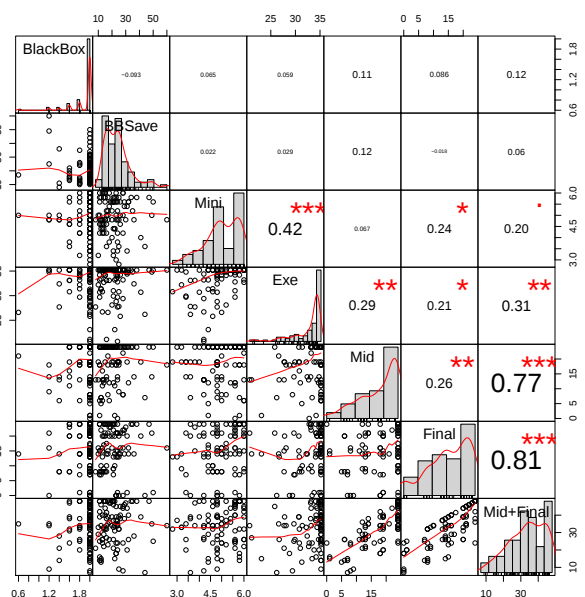


図4 ブラックボックス問題点数・保存回数と、その他点数との相関 (クラスC, N=99)

た。試験終了後に、アンケートを実施した。アンケート項目は付録A.3に示す。

4.4 実践2の結果

表2に、実践2における点数の平均と標準偏差を示す。問題1~10における傾向としては、実践1の結果(表1)と類似しているが、問題10の正答率が低かった。この理由として、学生が問題に難しさを感じて最後まで解けなかった可能性がある。また試験形式(制限時間内)での回答は、思考時間が不足したことで正答率は低かった。法則を推測

表 1 実践 1 におけるブラックボックス問題点数の平均と標準偏差 (0.2 点満点)

クラス	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
A	0.2	0.2	0.191	0.144	0.130	0.184	0.2	0.198	0.182	0.184
(標準偏差)	0.000	0.000	0.042	0.090	0.096	0.054	0.000	0.021	0.058	0.054
B	0.195	0.195	0.188	0.152	0.141	0.183	0.193	0.188	0.164	0.162
(標準偏差)	0.031	0.031	0.048	0.086	0.092	0.056	0.037	0.048	0.077	0.079
C	0.2	0.2	0.186	0.174	0.172	0.188	0.2	0.194	0.188	0.192
(標準偏差)	0.000	0.000	0.052	0.068	0.070	0.048	0.000	0.034	0.048	0.040

表 2 実践 2 におけるブラックボックス問題点数の平均と標準偏差 (0.1 点満点, N=68)

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
平均	0.0971	0.0971	0.0971	0.0943	0.0914	0.0929	0.0957	0.0943	0.0929	0.0900	0.0886	0.0757
標準偏差	0.0168	0.0168	0.0168	0.0234	0.0282	0.0259	0.0204	0.0234	0.0259	0.0302	0.0320	0.0432

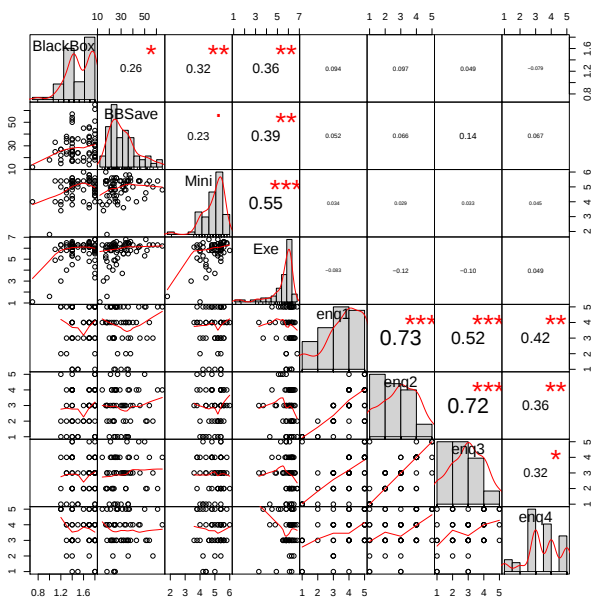


図 5 実践 2 におけるブラックボックス問題点数・保存回数と、アンケートとの相関 (N=68)

したうえで、プログラムで記述する、というステップを踏む必要があるため、時間制限は設けないか、十分に長く設定したほうが、学習者にとって回答しやすいと考えられる。

実践 2 におけるブラックボックス問題点数・保存回数と、小テスト点数、演習点数、およびアンケート回答の相関表を図 5 に示す。ブラックボックス問題点数 (Blackbox) と、保存回数 (BBSave)、小テスト点数 (Mini)、演習点数 (Exe) のあいだで、弱～中程度の相関 (0.26～0.36) が確認できた。また、アンケート ENQ1～ENQ4 のあいだに、相関 (0.32～0.73) の相関が確認できた。しかし、ブラックボックス問題点数・保存回数、小テスト点数、演習点数と、アンケート回答とのあいだには、有意な相関は確認されなかった。

アンケートの自由記述では、「面白くてすごくいいと思います」「頭を使うので楽しかった」「小学生の頃といていた問題を思い出して懐かしく感じた」といった肯定的な意見が 6 件、「難しかった」「法則を見つけるのに苦労してい

る」「具体例からある程度規則性を見いだせることができるが、そこからの推測がうまくいかない」といった否定的な意見が 8 件あった。また「関数のみでここまで活用できることが知れて、幅が広がったように思う」という意見が 1 件あった。このことから、条件文 (if や switch-case 文) に頼らずに、演算子のみで解決可能な問題の範囲を示せたと考えられる。

4.5 議論

アンケート結果から、ブラックボックス問題によって、演算子のみで解決可能な問題の範囲を示すことができたといえる。一方で、ブラックボックス問題に対する素質や特性と、他のプログラミング演習課題や試験パフォーマンスとの関連は明確にできたとはいえない。この理由としては、演習課題や試験で要求する知識や技能、問題解決力と、ブラックボックス問題が要求するそれらが完全に一致していないことが挙げられる。

保存回数と点数の関係が正の相関であることは、多くのブラックボックス出題にめげずに、最後まで諦めない姿勢や、何度もチャレンジする態度が影響していると考えられる。

一般に、プログラミングにおいては、「簡潔」かつ「読みやすく理解しやすい」ソースコードを書くことが要求されている。しかし、「読みやすく理解しやすい」の観点においては、ブラックボックス問題で求めているようなワンライナー記述より、条件分岐を用いたほうが、自然で適切な場合もある。学習者には、複数のアプローチや、解法が存在を理解してもらったうえで、そのアプローチを選択した理由の説明や、それぞれの利点、欠点を検討できるようになることが望ましい。たとえば、1～N までの和を計算する問題の場合、繰り返しを用いる解法と「和の公式」による解法があるが、どちらかで解ければよいということではなく、両解法の利点、欠点について学習者が理解し説明できることが望まれる。

本研究実践におけるブラックボックス問題演習は、形式

的には Skinner に代表される行動主義的な学習観（プログラム学習、スモールステップ等）に基づいており、教授者が配列した知識を、学習者が受動的に獲得していくというスタイルに近いものとなっている。しかし、学習理論における近年の潮流としては、構成主義的な学習観 [9], [10] に基づき、学習者自身が知識を主体的に構成していくことを重要視している。本研究実践においても、入口は行動主義的であっても、最終的には構成主義的な考え方やスタイルが学習者に定着するような方向性を探っていきたいと考えている。

5. おわりに

剰余を含む演算子のみでの問題解決思考を促すブラックボックス問題を作成し、プログラミング学習の初期の段階において、複数の科目に導入した。ブラックボックス問題で要求する知識や能力と、プログラミング演習や試験の点数との明確な関連は確認できなかったが、アンケート結果より、条件分岐を使わず、演算子のみで解決可能な問題の範囲を示せることが確認できた。ブラックボックス問題によって、学習者が演算子に対する理解を深め、複数の解法を考えたり、それらの是非を議論・検討したりできるようになる契機になればと考えている。

謝辞 本研究の一部は JSPS 科研費 (課題番号 19K03056) の支援によるものです。

参考文献

- [1] Iain Milne and Glenn Rowe. Difficulties in learning and teaching programming—views of students and tutors. *Education and Information technologies*, Vol. 7, No. 1, pp. 55–66, 2002.
- [2] Büşra Özmen and Arif Altun. Undergraduate students' experiences in programming: Difficulties and obstacles. *Turkish Online Journal of Qualitative Inquiry*, Vol. 5, No. 3, pp. 9–27, 2014.
- [3] Katherine B McKeithen, Judith S Reitman, Henry H Rueter, and Stephen C Hirtle. Knowledge organization and skill differences in computer programmers. *Cognitive Psychology*, Vol. 13, No. 3, pp. 307–325, 1981.
- [4] 土肥紳一, 宮川治, 今野紀子. SIEM を導入したプログラミング教育の実践効果. 情報教育シンポジウム 2003 論文集, Vol. 2003, No. 12, pp. 199–204, 2003.
- [5] 松本慎平, 山岸秀一, 加島智子, 林雄介, 平嶋宗. 書かせること以外のプログラミング指導から見てきたこと. 人工知能学会全国大会論文集 第 30 回全国大会, 1C5-OS-13b-2, 2016.
- [6] 徳島県中学校教育研究会数学部会. 平成 19 年 研究冊子「数学的活動を楽しむことのできる教材研究 (実践事例)」第 1 学年. April 2007. <http://chukyoken.tokushima-ed.jp/sugaku/> (2019 年 5 月 23 日確認).
- [7] Joe Michael Allen, Frank Vahid, Kelly Downey, and Alex Daniel Edgcomb. Weekly Programs in a CS1 Class: Experiences with Auto-graded Many-small Programs (MSP). In *2018 ASEE Annual Conference & Exposition*, 2018.
- [8] Joe Michael Allen, Frank Vahid, Alex Edgcomb, Kelly Downey, and Kris Miller. An Analysis of Using Many Small Programs in CS1. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pp. 585–591. ACM, 2019.
- [9] 生田孝至, 後藤康志. 構成主義的な学習観の教育への展開. 新潟大学教育人間科学部紀要 人文・社会科学編, Vol. 10, No. 1, pp. 1–12, 2007.
- [10] 西城卓也. 行動主義から構成主義. 医学教育, Vol. 43, No. 4, pp. 290–291, 2012.

付 録

A.1 ブラックボックス問題

【問題 1】

以下の例は、関数 $f(x)$ の入力と出力を、あらわしています。
整数 x を 1 つ入力したら、この関数 $f(x)$ の値を出力するプログラムを作成してください。
(具体例から一般化して、関数を推測してください)

$$f(2) = 4 \quad f(3) = 9 \quad f(5) = 25$$

【問題 2】

(問題 1 の問題文はすべての問題に共通であるため、以降省略)

$$f(3) = 9 \quad f(4) = 8 \quad f(5) = 7$$

【問題 3】

$$f(3) = 3 \quad f(4) = 4 \quad f(5) = 0 \quad f(7) = 2 \quad f(9) = 4$$

【問題 4】

$$f(2) = 0 \quad f(3) = 3 \quad f(4) = 0 \quad f(5) = 5$$

【問題 5】

$$f(2) = 2 \quad f(3) = 0 \quad f(4) = 4 \quad f(5) = 0$$

ヒント：問題 4 との違いに着目してください。mod の結果 (0 または 1) を反転させるために、括弧と引き算をつかいます。

【問題 6】

$$f(7) = 7 \quad f(10) = 0 \quad f(12) = 2 \quad f(27) = 7 \\ f(34) = 4$$

【問題 7】

$$\begin{array}{llll} f(1) = 0 & f(3) = 0 & f(4) = 1 & f(5) = 1 \\ f(7) = 1 & f(8) = 2 & f(16) = 4 & \end{array}$$

【問題 8】

この問題については、入力値の範囲は、99 以下の非負の整数とします。

$$\begin{array}{llll} f(6) = 6 & f(10) = 1 & f(13) = 4 & f(36) = 9 \\ f(46) = 10 & f(87) = 15 & f(91) = 10 & \end{array}$$

【問題 9】

$$f(3) = -3 \quad f(4) = 4 \quad f(5) = -5$$

【問題 10】

$$f(3) = 3 \quad f(4) = -4 \quad f(5) = 5$$

【問題 11】

この問題については、入力値の範囲は、99 以下の非負の整数とします。

$$\begin{array}{llll} f(99) = 81 & f(36) = 18 & f(77) = 49 & f(44) = 16 \\ f(75) = 35 & f(80) = 0 & f(16) = 6 & \end{array}$$

【問題 12】

$$\begin{array}{lll} f(1) = 0 & f(2) = 2 & f(3) = -3 \\ f(4) = 0 & f(5) = 5 & f(6) = -6 \\ f(99) = -99 & & \end{array}$$

A.2 ブラックボックス問題の解答例

【問題 1】

```
program blackboxtest1
implicit none
integer x
read(5,*) x
write(6,*) x**2 ! edit here
stop
end
```

【問題 2】 write(6,*) 12 - x

【問題 3】 write(6,*) mod(x,5)

【問題 4】 write(6,*) mod(x,2) * x

【問題 5】 write(6,*) (1 - mod(x,2)) * x

【問題 6】 write(6,*) mod(x,10)

【問題 7】 write(6,*) x / 4

【問題 8】 write(6,*) (x / 10) + mod(x,10)

【問題 9】 write(6,*) (1 - mod(x,2)*2) * x

【問題 10】 write(6,*) (1-(1-mod(x,2))*2) * x

または write(6,*) (2*mod(x,2) - 1) * x

【問題 11】 write(6,*) (x/10)*mod(x,10)

【問題 12】 write(6,*) (1-mod(x,3))*x

A.3 アンケート項目

- ENQ1: あなたは、ブラックボックス問題のような、具体的から類推・推測することは好きですか？
 - － 好き (5) / どちらかという好き (4) / どちらでもない (3) / どちらかという嫌い (2) / 嫌い (1)
- ENQ2: あなたは、ブラックボックス問題のような、具体例から類推・推測することは得意ですか？
 - － 得意 (5) / どちらかという得意 (4) / どちらでもない (3) / どちらかという苦手 (2) / 苦手 (1)
- ENQ3: あなたは、ブラックボックス問題における、事例の一般化（式をかながえたり、プログラムで表現すること）は得意ですか？
 - － 得意 (5) / どちらかという得意 (4) / どちらでもない (3) / どちらかという苦手 (2) / 苦手 (1)
- ENQ4: あなたは、ブラックボックス問題にどのように取り組みましたか？（なるべくあてはまるものを 1 つ選択してください）
 - － (戦略 A) 頭の中や、紙のうえでじっくり考えて、うまくいきそうだと確認してから、プログラム作成にとりかかった. (5)
 - － 戦略 A と戦略 B を、問題によって適宜切り替えた. (4)
 - － (戦略 B) なんとなくうまくいきそうだった段階で、プログラムとして入力し、実行結果をみながら修正していった. (3)
 - － 戦略 B と戦略 C を、問題によって適宜切り替えた. (2)
 - － (戦略 C) 深く考えずに、思いついた式を手当たり次第にプログラムとして入力し、何度も実行して、結果をみながら修正していった. (1)
- ENQ5: ブラックボックス問題に対する感想や意見、要望などがありましたら、ご自由にお書きください。（自由記述）