コンピュテーショナル・シンキングの育成を目指した小学校プログラミング実践の検討

山川 大輝

要約

本研究は，小学校プログラミング教育において育成が目指されるプログラミング的思考の基盤であるコンピュテーショナル・シンキング（CT），特に抽象化に着目し，その育成と評価の在り方を検討した。じゃんけんゲーム，迷路，セルフレジという 3 つの Scratch を活用した実践を通して，CT の抽象化を児童に育成することができた。実践の課題として，アンプラグドな課題については多くの児童が達成することができたが，Scratch 上での実装に苦戦する場面が見られた。その理由として，児童の Scratch に関する習熟度や Scratch 自体の複雑さが挙げられる。

キーワード：小学校プログラミング教育，コンピュテーショナル・シンキング，抽象化，評価，Scratch

１．はじめに

小学校プログラミング教育では，プログラミング的思考という考え方を育むことが目標とされている（文部科学省 2020，p.9）。この定義については，文部科学省の有識者会議にて，「いわゆる『コンピュテーショナル・シンキング』の考え方を踏まえつつ，プログラミングと論理的思考との関係を整理しながら提言された定義である」（文部科学省 2016）と補足されている。この記述から，プログラミング教育が獲得を目指すプログラミング的思考のベースとなった考え方がコンピュテーショナル・シンキング（以下 CT）と捉えることができる。

CT について，Wing（2008）は問題を解決し，システムを設計し，人間の行動を理解するために，コンピューティングの基本的な概念に基づいたアプローチをとることであると説明している（p.3717）。ただ，CT 自体の定義が曖昧であるという特質から，研究者間で CT の構成要素が異なっている。Shute et al.（2017）は，CT の構成要素として分解，抽象化，アルゴリズム，デバッグが文献の中で最も頻繁に出てくる構成要素であると示している（p.151）。その中でも，Wing（2008）は CT の構成要素について，抽象化が CT の本質であると指摘している（p.3717）。そして，Shute et al.（2017）はさまざまな文献の定義をふまえて CT における抽象化とは，問題や解決策の中にパターンを見つけることだと指摘している（p.151）。

CT の評価について，CT の定義や概念が多様であることから，広く受け入れられている評価がないという問題がある（Shute et al. 2017，p.149）。研究者は様々な研究において独自の CT 尺度を開発し適用する傾向があるが，CT の知識または CT に対する態度のために最も一般的に使用される評価方法はアンケート調査である（Shute et al. 2017，p.149）。だが，プログラムを用いて抽象概念を具現化することが CT 特有のものである（Mirolo et al. 2021，p.626）との指摘からも，抽象化能力をアンケート形式で評価することができるのかについては疑問が残る。

本研究の目的は，プログラミング教育が獲得を目指すプログラミング的思考のベースとなった考え方である CT，その本質である抽象化の育成と評価を目指した小学校プログラミング教育の実践を検討することである。提案する実践で

はいずれも Scratch を使用する。

２．CT における抽象化の評価：小学校におけるプログラミングの実践を通して

本実践では，各じゃんけんの手に対応するキーを同時に押すことで，じゃんけんを行うことのできるゲームをプログラミングする（図 1）。

じゃんけんをテーマとして設定した意図は，児童に馴染みがあるためである。Wing（2014）は CT の定義について，問題を定式化し，その解決策を，コンピュータが効果的に実行できるような方法で表現するための思考プロセスであると指摘している。この定義を踏まえると，対象とする問題はイメージしやすい題材が適切だと考えられる。よって，児童がじゃんけんのパターンを特定し，プログラムで表現することができるよう，実践を構想した。

抽象化の評価については，Scratch で作成したプログラムを収集し，個々のプログラムの内容を検討し，アンケートでの評価も行う。アンケートは「じゃんけんゲームでうまくじゃんけんすることができた」（項目 A），「じゃんけんゲームのむずかしさは自分に合っていた」（項目 B），「授業ではプログラミングをする時間が十分にあった」（項目 C）の 3 つの質問に対して，「5 そう思う」～「1 そう思わない」の 5 段階で評価を行う欄を設定した。この他に，分かったことや難しかったことなどを自由に記述する感想欄を設けた。

令和 6 年 12 月 16 日に，A 市立 B 小学校の 4 年生の児童 14 名を対象として，40 分の授業 2 時間で実践を行った。実践にあたって，大学生 3 名が授業中の補助者を務めた。

授業後に行ったアンケートの結果について，項目 A の平均値は 4.42，項目 B は 4.07，項目 C は 3.92 と全体的に高い数値であった。項目 B も高い数値であったが，自由記述の欄では課題の難しさに関わる内容を記述する児童が 14 名中 7 名と半数見られた。ただ，「しだいになれてきてた」，「ペアと力を合わせることでむずかしいプログラミングもできる」といった記述などもあり，プログラミングに児童が慣れることや手助けとしてペアを活用することで難しさが低減されていったと考えられる。また，項目 C が他の項目と比較して低い数値となった。これらの結果から，児童がプログラミングを行うことができるように活動時間をより長く取ることが必要であった。作成されたプログラムの分析については，正常に動作するプログラムを作成していた児童は 7 名，プログラムを完成させていたが不具合が見られた児童が 4 名，完成しなかった児童は 3 名であった。

実践の結果として，アンケートの項目 A が高い数値であったが，半数の児童が課題の解決には至らなかった。この結果から，児童の作成したプログラムと課題の達成に関わるアンケート項目の自己評価には乖離が見られたため，アンケート中心の評価で抽象化の能力を測るのは課

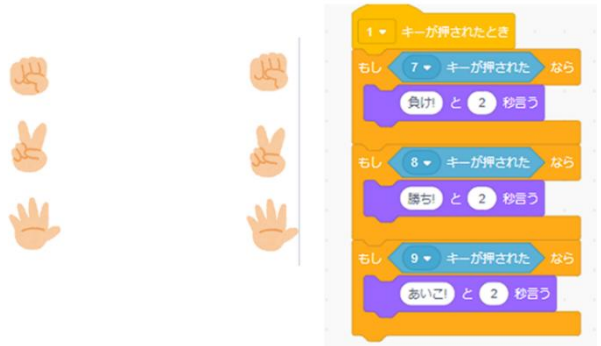

図１　じゃんけんゲームのイメージ（左）と児童が作成するプログラム（右）

題があると考えられる。

今回の評価における課題として，プログラミングが苦手であるが抽象化の思考はできていた児童を評価することが難しいこと，児童の間違い方を大きく分類したため，同じ分類の児童の中でどこまで抽象化できているのかという差異が見取れなかったことが挙げられる。

３．CT における抽象化を育成するための小学校プログラミングの実践と評価：条件分岐の考えを通して

本章では，縦 5 マス×横 7 マスの迷路の左下の位置からスタートし，右上のゴールを目指す経路を検討する実践を構想した（図 2）。本実践では，ゴールまでの経路のアルゴリズムをワークシート（アンプラグド）および Scratch のプログラムとして表現する活動とした。実践では以下の 3 つのステップに分けて課題を設定した。ステップ 1 では，障害物の無い状態で経路を検討する。ステップ 2 では，道中に障害物としてウェイターを配置し，それを避けるような経路を考察する。ステップ 3 では，もしウェイターが進路上にいた場合には，進行方向を変更する条件分岐のブロックを追加する。ステップ 1 の何もない状態から，ステップ 2 にて障害物であるウェイターを追加してもゴールへ到着できる経路を検討することで，複数の問題の中から共通するパターンを見出す。そして，最終的にはステップ 3 でどのような状況であっても適用できるようなアルゴリズムに昇華させることを期待した。これらは，問題の本質的なパターンを特定することになり，CT における抽象化として捉えることができる。

本実践は，令和 7 年 6 月 18 日および 20 日に，総合的な学習の時間の一部として，C 市立 D 小学校第 4 学年の児童を対象に，45 分の授業を 2 時間連続として実施した。出席者はそれぞれ，28 名と 32 名であった。実践にあたって，それぞれの授業で大学生 1 名が補助者を務めた。

抽象化については，実践前後のテストの達成状況（分析 1），各ステップにおけるワークシートに記述するアルゴリズムの達成度状況（分析 2），作成されたプログラムの解析（分析 3）の 3 点で評価する。テストについては，Bubica and Boljat（2022）の抽象化を評価するための問題を参考にして作成した（図 3）。なお，分析 1 では，事前テストと事後テストの両方が揃っている児童のみを対象とした。分析 1 の対象となった児童数は 53 名であった。結果について，事前テストは 24 名が正答であったが，事後テストでは 46 名が正答と，事後テストの正答者が増加した。これは，今回の実践の中で条件分岐の考え方を取り扱ったことによって，蜂がどのように動くのかイメージしやすくなったためだと考えられる。振り返りで設定したアンケートの自由記述欄には，「最初の紙に書くやつは意見がわからなかった」（原文ママ）といった記述もみられた。これまでに見たことのない問題に対する難しさがあったことを踏まえると，条件分岐の考えを十分に学んだ後の事後テストでは児童が問題文

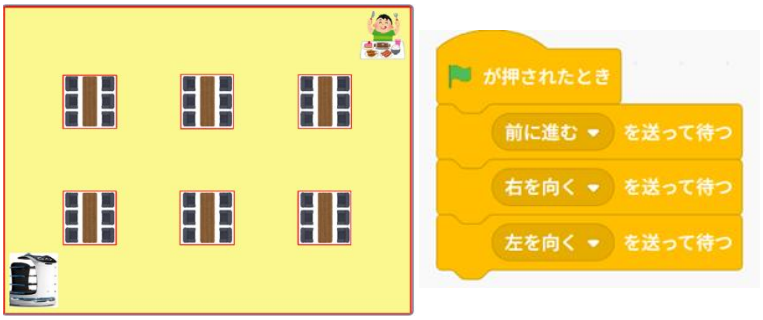

図２　めいろのイメージ（左）と児童が作成するプログラム例（右）

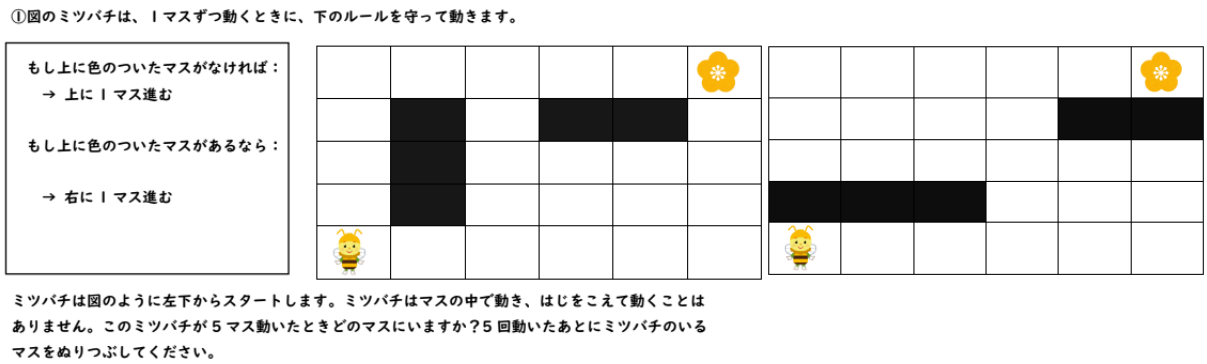

図３　事前テスト（左）と事後テストの図（右）

の意味を理解しやすかったと考えられる。

分析 2 では，60 名の児童が対象となった。ステップ 1 の正答者数は 54 名，ステップ 2 では 55 名，ステップ 3 では 52 名と多くの児童が正しい順路でゴールまでの経路を記述していた。

分析 3 のプログラムの解析では，ステップ 2 で正常に動作するプログラムを作成していた児童は 30 名中 18 名，ステップ 3 では 35 名中 19 名であった。ステップ 1，2，3 のいずれもワークシートでは多くの児童が課題を解決できたが，Scratch 上での実装に課題が見られた。

本実践の成果としては，実践前後のテストを行ったことで，児童の抽象化の変化を 2 章の実践と比較して詳細に評価できたこと，アンプラグドな評価を導入したことで，プログラミングは不得手であるが抽象化の思考はできていた児童を評価することができたことが挙げられる。

本実践でも Scratch の実装については課題が認められた。その理由として，本実践中では使用するブロックが記載されたワークシートを使用せずにプログラミングを行う児童も一定数見られた。その他の課題として，事前・事後のテストに時間がかかりすぎたこと，児童が作成した Scratch のプログラムを完全に回収することができなかったことが挙げられる。

4．CT における抽象化を育成するための小学校プログラミングの実践と評価：定義ブロック・バックパックを通して

現実世界をテーマとすること，定義ブロックを活用することといった視点を踏まえ，実践のテーマとして，セルフレジを設定した（図 4）。この課題は「魚屋 魚スク POS レジ POS システム」（https://scratch.mit.edu/projects/786326095）をリミックスしたものである。本実践で用意したプロジェクトは，リミックス元を踏まえ，紫色の部分で魚をスキャンし，お会計ボタンを押すことでスキャンした魚の合計金額を出力するものである。基礎課題では，3 匹の魚を対象にプログラミングを行うが，発展課題では魚の数を 10 匹に増やした。多くの魚それぞれにプログラミングを行うことは大変であることを共有

図４　セルフレジ課題のイメージ（左）と児童が作成する定義ブロックを用いたプログラム（右）

し，定義ブロックとバックパックの考え方を示す。Scratch では，定義ブロックに特定のプログラムをまとめることで，プログラム全体を簡略化することができる。また，バックパックはプログラムを保存し，保存したプログラムを引き出すことができる機能であり，この機能を用いることで，定義ブロックを異なるスプライト間で使いまわすことができる。

抽象化については，振り返りにおける定義ブロックとバックパックの説明課題（分析 1），作成されたプログラムの解析（分析 2）の 2 点で評価する。分析 1 では，児童の記述を，「抽出」，「本質的でない情報の無視」，「一般化」の 3 つの観点で分類する。分析 2 の基礎課題に関して，プログラムが問題なく動作する児童を正答，プログラム自体は正しいものであるが設定した変数の値が異なっているなど，概ね完成しているが不備が見られる場合には部分正答，プログラムが完成していない場合やそもそもアルゴリズムが異なる場合には誤答，未回答の 4 つで評価した。発展課題についても，評価基準は同じであるが，この課題では定義ブロック内の魚の名称に関する部分が異なる場合にも誤答とした。その理由は，発展課題では魚の名称が違う場合にプログラムが正常に動作することはないこと，抽出の観点から，各プログラムの共通する部分とそうでない部分を理解していないからである。ただし，定義ブロックのみに焦点化した場合，魚の名称が異なる場合でも定義ブロックのアルゴリズムとしては適切である。そこで，定義ブロックの評価も行う。使用されている定義ブロックのアルゴリズムと魚の名称が正しいものである場合には正答，アルゴリズムは正しいが，魚の名称が異なっている部分があるものは部分正答，定義ブロック内のアルゴリズムや使用するブロックが異なっている場合には誤答，未回答の 4 つに分類した。

本実践は，令和 7 年 10 月 30 日および 11 月 5 日に，総合的な学習の時間の一部として，C 市立 D 小学校第 4 学年の児童を対象に 45 分の授業を 2 時間連続として実施した。出席者はそれぞれ，26 名と 31 名であった。実践にあたって，10 月 30 日は大学生 3 名，11 月 5 日は大学生 2 名がそれぞれの授業で補助者を務めた。

分析 1 について，回収できたワークシートは 50 名分であり，そのうち未回答は 4 名であった。プログラムを使いまわすことができるという「抽出」は定義ブロックで 0 名，バックパックで 14 名だった。ブロック数を削減するという「本質的でない情報の無視」は定義ブロックで 9 名，バックパックで 0 名だった。しかし，「一般化」は定義ブロック，バックパックともに 1 名のみだった。この結果のみでは，抽象化の思考を働かせた児童数は少ないと考えられる。だが，「楽になる」などの効率化に関する記述が定義ブロックで 12 名，バックパックで 9 名と多く見られた。定義ブロックとバックパックを使用してプログラミングを効率的にするためには，抽出や本質的でない情報の無視といった思考を働かせる必要がある。これらの記述を含めると，半数以上の児童が抽象化を働かせた可能性があることを示唆している。

分析 2 について，結果を表 1 に記載する。

基礎課題では 41 名が正答と，3 章実践と比較して正答者数が増加した。これに対して，発展課題については誤答が 38 名という結果となった。発展課題においては，定義ブロック内で魚の名称が適切に変更できていないことから，誤答に分類されるものが多かった。その理由として，定義ブロックをすべての魚へコピーし，その後修正しようとしたが間に合わない児童も見

表 1　分析 2 の結果（N＝57）

	正答	部分正答	誤答	未回答
基礎課題	41	8	6	2
発展課題	4	13	38	2
定義ブロック	19	33	3	2

られたことが挙げられる。一方で，定義ブロックのプログラムについては，正答 19 名，部分正答が 33 名であった。発展課題については誤答に分類されるプログラムが多かったが，定義ブロックを見るとアルゴリズムは適切に表現されているものが多かった。また，バックパックについては，回収できた児童のプログラムすべてが定義ブロックを正しく取り込めていた。

本実践では，抽象化の思考の評価について，説明課題での評価を行った。その結果として，児童の定義ブロックとバックパックに対する思考から，「抽出」と「本質的でない情報の無視」といった抽象化の思考について評価することができた。また，プログラミング課題については，3 章実践の課題であったプログラムの回収率を改善することができ，プログラミングについても正答に近づく児童数が増加した。

５．おわりに

本研究では，CT における抽象化を育成・評価する小学校プログラミング教育の実践を提案することを目的として論を進めてきた。本研究の成果として，現状の CT 評価の課題を踏まえてアンケートに留まらない評価方法を提案することができた。また，児童のプログラミング課題の達成度についても，実践ごとに正答者が増加していた。本研究の課題として，児童が Scratch での実装に苦戦する場面が見られた。その原因として，児童が Scratch に習熟していなかったこと，関連して Scratch のブロック数の多さやプログラムの複雑さに混乱してしまったことが考えられる。

謝辞

実践に協力して頂いた小学校の児童の皆さん，校長や学級担任を始めとする先生方，実践の準備とサポートをして頂いた大学生の皆さんに謹んで謝意を表します。

本研究に関わる学会発表等

山川大輝・深見俊崇（2024）コンピュテーショナル・シンキングに基づく小学校低学年を対象としたプログラミング教育の実践，島根大学教育臨床総合研究，23，31-46.

山川大輝・深見俊崇（2024）コンピュテーショナル・シンキングにおける抽象化能力の評価の課題，日本教育工学会 2024 年秋季全国大会講演論文集，451–452，東北学院大学（ポスター発表）.

山川大輝・深見俊崇（2025）コンピュテーショナル・シンキングにおける抽象化の評価：小学校におけるプログラミングの実践を通して，日本教育工学会 2025 年春季全国大会講演論文集，461‐462，成城大学（口頭発表）.

山川大輝・深見俊崇（2025）コンピュテーショナル・シンキングにおける抽象化能力の評価方法，島根大学教育臨床総合研究，24，131–141.

山川大輝・御園真史・深見俊崇（2025）コンピュテーショナル・シンキングにおける抽象化を育成するための小学校プログラミングの実践と評価，日本教育工学会 2025 年秋季全国大会講演論文集，605–606，ウインクあいち（ポスター発表）.

引用文献

Bubica, N. & Boljat, I. (2021). Assessment of Computational Thinking: A Croatian Evidence-Centered Design Mode, Informatics in Education, 21(3), 425-463.

Mirolo, C., Izu, C., Lonati, V., & Scapin, E. (2021). Abstraction in Computer Science Education: An Overview, Informatics in Education, 20(4), 615-639.

文部科学省 (2016). 小学校段階における論理的思考力や創造性，問題解決能力等の育成とプログラミング教育に関する有識者会議（第 3 回）議事録，https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/gijiroku/1382219.htm（2026 年 1 月 4 日最終閲覧）.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying Computational Thinking, Educational Research Review, 22, 142-158.

Wing, J. M. （著），中島秀之（訳）(2006/2015). Computational Thinking. Communications of the ACM, 49(3), 33-35（計算論的思考，情報処理，56(6)，584-587）.

Wing, J. M. (2008). Computational Thinking and Thinking about Computing, Philosophical Transactions, Series A, Mathematical, Physical, and Engineering Sciences, 366, 3717-3725.

Wing, J. M. (2014). Computational Thinking Benefits Society, http://socialissues.cs.toronto.edu/2014/01/computational-thinking/（2026 年 1 月 4 日最終閲覧）.