プログラミング的思考とプログラミング力の関係

米 川 雅 士

目次

1．背景・目的

日本の教育カリキュラムは1947年に施行された「教育基本法」と「学校教育法」を基に全国どの地域に住んでいても同じ水準の教育を受けられるようにするため，文部科学省は1951年にアメリカ教育使節団の報告書の影響を強く受けて学習指導要領を規定し，この学習指導要領をベースに教科書の作成や，教員の指導方法についての統一が図られている。最近では学習指導要領は約10年に1度改定されるが，「詰め込み教育」「ゆとり教育」など毎回改定があるたびに大きな話題となっている。最新の改訂は，初等教育が2020年度から改定され，中等教育では2021年度に中学校が，2022年度には高等学校で改訂されるが，今回も大きな改革が何点かあり改定前から話題となっている。なお，変更点の詳細については2章に記述する。その改革点の中で本論文が着目したのは，全教育期間で実施されるプログラミング教育の必修化である。

このプログラミング教育は小学校で3年生からの4年間，中学校で3年間，高等学校で3年間の合計10年間の教育課程が用意されている。それだけで将来の日本において重要な知識であることは容易に想像できる。では，なぜプログラミング教育が必要かというと日本では概ね10年毎に未来社会の姿として内閣府から科学技術政策が提唱される。2020年から提唱されているのがSociety5.0である。これは仮想現実空間を指すサイバー空間と現実世界を指すフィジカル空間を高度に融合させることで，経済発展と社会的課題を解決し，人間がより快適な生活を送れるようにするためデジタル化を進める事が求められている。このデジタル化を担う人材としてITに精通した人材が質と量の両面で必要となる時代が来ている。現在の予想では独立行政法人情報処理推進機構の報告である「IT人材白書2020」[1]によると，日本のIT人材は企業の大小や業種に関わらず2030年には約79万人が不足すると報告がされている。2020年から小学生のプログラミング教育が始まり，大学を卒業するまでに約14年かかると考えた場合に全てのプログラミング教育を受けて社会に

出るのが2034年となる。日本の教育を受けた子ども全員がIT関連の職種を希望する事はありえないので，この点を考えても将来にわたって日本のIT人材不足は深刻であることが伺え，このような状況で実施されるプログラミング教育の重要度はとても高いと言える。

今回の学術指導要領改定のポイントの１つとして「主体的・対話的で深い学び」が挙げられている。これはアクティブ・ラーニングと言った方がイメージしやすいと思われるが，このアクティブ・ラーニングの学習自体は正しく実施されればプログラミング教育に適した学習方法だと考えている。なぜならば，本来プログラミングとはコンピュータに命令を出すための道具であって，このプログラミングを使って何をコンピュータに実行させるかという方が重要である。この点について今回の学習指導要領には，小学校ではプログラミング学習の時間にはコンピュータに向かって一人で作業するような指導はなく，プログラミングを体験しながら，コンピュータに意図した処理を行わせるために必要な論理的な思考力を身に付けるための学習活動を実施すると定義されており，これは実際にプログラミングを作成するわけではなく，プログラミング的思考の育成を目指した教育を実施するものである。ちなみにプログラミング的思考とは学習指導要領には「自分が意図する一連の活動を実現するために，どのような動きの組み合わせが必要か，どのように改善していけばより意図した活動に近づくのかという事を論理的に考えていく力」と説明されているが，どのような力なのかイメージがしにくいと思われる。そこで，本論文では前記したようにプログラミング教育が導入される原因はIT人材の不足からきていると説明したが，それならばプログラミング教育の最終目的はIT人材の育成であり，そのスタートにある小学校での教育で最も重要視されているのがプログラミング的思考の育成をすることが重要だと明記されている。そこで実際にプログラミング的思考とプログラミングを作成する能力に関連性があるのかについては明確な研究報告はされていないため，逆説的であるがプログラミングを作成する能力を持っている学生はプログラミング的思考も持っているのか確認し，プログラミング的思考とはどのような能力で構成されているのかについて研究し，最終的にはどのような力を付け，どのような教育がプログラミング教育として最初の学習を実施する小学校では適しているのかについて提案することを目的とする。

２．2020年改訂の学習指導要領

学習指導要領とは教育基本法と学校教育法等に従い作成され，児童・生徒の人間としての調和の取れた育成と心身の発達を考慮し適切な教育過程が日本全国どこにいても一定水準で受けられるように編成されている。学習指導要領は小学校，中学校，高等学校等ごとに，それぞれの教科等の目標や大まかな教育内容が定められており，その他に学習指導要領とは違い法的拘束力はないが２種類の補足文書が作成されている。１つ目は学校教育法施行規則であり，年間の標準授業時間が定められている。２つ目は学習指導要領解説であり，科目毎に発行され学習に関する詳細について様々な解説事項が記述されている。1951年の実施から学習指導要領は現在までに一部改訂も加えると今回の改定で９回目となる。全改定の説明は本論文の趣旨とは異なるので割愛するが，2020年度から施行される最新の

改定概略については本章で説明する。

2020年からの施行される新改訂では，育成すべき資質・能力の 3 つの柱を総合的にとらえて，それぞれを構造的に組み上げる事で将来に社会で活躍できる人材の育成を目指して改定が進められている。3 つの柱には，どのように社会・世界と関わり，よりよい人生を送るかを学ぶ「学びに向かう力・人間性等」，何を理解しているか，何ができるのか学習者が客観的に理解できる「知識・技能」，理解している事とできる事をどのように使って探究するための能力「思考力・判断力・表現力」である。これらを柱に据え，情報化やグローバル化といった社会的変化が予想を超え，予測不可能な社会となったとしても主体的に向き合い，広い視野を持って自分の人生を切り開いていけるような力を身に付けていく事を重視した改定が行われている。また，これらを身に付ける学び方についてもアクティブラーニングの視点から，学ぶ事に興味や関心を持ち，自分のキャリア形成の方向と関連付けながら将来の見通しをもって粘り強く取り組み，自己の学習活動を振り返って次につなげる事ができる「主体的な学び」，子ども同士の協働，教職員や地域の人との対話，先哲の考え方を手掛かりに考える事を通じ自己の考えを広げつかめる「対話的な学び」，習得・活用・探究という学びの過程の中で，各教科の特質に応じた見方や考え方を働かせながら，知識と相互に関連付けてより深く理解したり，情報を精査して考えを形成したり，問題を見出して解決策を考えたり，思いや考えを基に想像したりする「深い学び」を実施するような授業改善についても初等教育・中等教育全課程で明記されている。

2.1 小学校の学習指導要領変更点［2］［3］

各教育内容において様々な改善事項があるが，本節では重要事項のみ説明する。大きく変更された点は 3 点ある。1 点目が外国語の教科化，2 点目が道徳の教科化，3 点目がプログラミング教育の導入である。

1 つ目の外国語の教科化は英語を指しているが，3 年生から授業が始まり，3・4 年生では英語に慣れ親しむことを目的とした「聞く」「話す」といったコミュニケーションを中心に学び，年間35コマの授業が行われる。これは 1 コマ/週の実施となる。5・6 年生では年間70コマに授業時間は増え，「聞く」「話す」に加えて文字指導や文脈からの気づきなどが指導され，評価が伴い成績が付く教科となる。

2 つ目の道徳は社会問題になっている「いじめ」「自殺」に対して，状況やその時の心の葛藤など，自分とは異なる意見を持つ他者と意見を交わすことで多角的な考えを学び，社会における道徳的価値を理解することで，自分自身と他者とのかかわり方を深めていく事に意義を見出す授業となる。しかし，このような授業では答えがあるわけではないため教科化されてはいるが評価が伴う成績を付ける事はない。

3 つ目のプログラミング教育は第 1 章にも記述したが，将来の日本において多くの IT 人材が必要となるが，圧倒的に人材が不足している。それらを少しでも解消するために小学校の 3 年生から情報やコンピュータについて学習することで，実際にプログラミングを作成する中等教育への準備を実施する教科となる。そのため「総合的な学習の時間」内で

学ぶ内容はいくつかあるが，その内の１つとして実施される。そのためプログラミング教育については教科化されていないため数値的な成績評価をすることはない。

2.2　中学校の学習指導要領変更点［4］［5］

中学校の新学習指導要領は2021年度から改定されているが，学習への方向性として，すでに持っている知識や新たな知識を組み合わせる事ができるような教育を実施することを目標としているため，問題点を導き出したり，解決方法に向けた行動に対して何が足りないのか導き出すような思考力や，様々な考えを尊重してその中で自分の意志を持つという事で集団の中で生活するために必要な感性と人間性を高める内容となっている。

教科の変更点は全教科にあるが，最も大きな変更点は２点あり，１点目がプログラミング教育の充実にあり，これは2012年に改定された学習指導要領から技術・家庭科の技術分野で学ぶべき内容として盛り込まれていたが，今回の改定でより高度且つ実践的な目標が設定されている。大きな改定は「ネットワークを利用した双方向のあるコンテンツのプログラミングによる解決」「計測・制御のプログラミングによる問題の解決」があげられる。これは自分自身で実施したい行動に対して必要なソフトウェアは何を使うのか判断し実際に最適な方法を考え，アプリケーションを作成する授業となっている。２点目は英語で求められる能力が変わる点である。今までは書く・読む・聞く・話すが重点項目となっていたが，単語数や高等学校で習っていた現在完了進行形，仮定法などの文法も増えりことにより，自分の表現したい内容や自国の文化などを英語できちんと伝える事や発表することを目指した教育が実施される。そのため，英語の授業でもグループワークが増え日常会話や様々な知識を駆使して新たな創設が求められる。

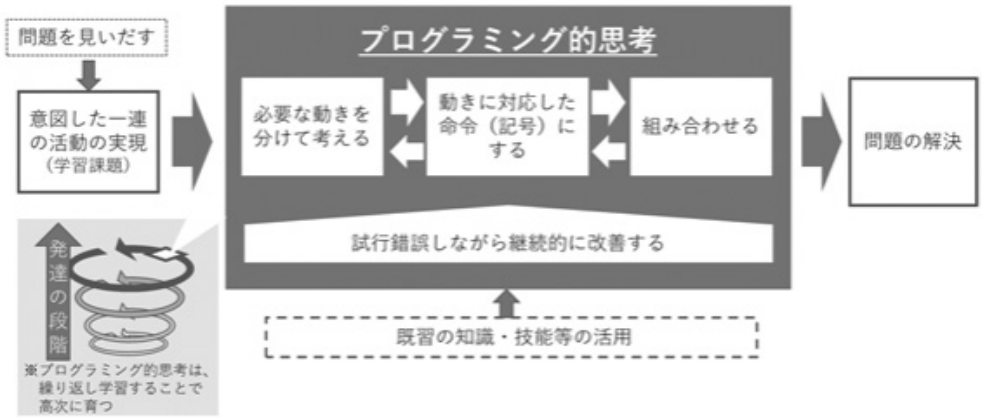
2.3　高等学校の学習指導要領変更点［6］［7］

高等学校においても各教科の枠組みや教科内容の変更はあるが，大きな変更点のみ本節では説明する。１点目は地理歴史・公民である。グローバル化や科学技術の発展が高速で進むことから，社会の激しい変化に対応する力と課題解決力が必要と考え，実際の社会が持つ課題に対して解決方法を見出す考え方を身に付けるため，思考力と判断力だけではなく，表現力や人間性を高めるための社会的な見方や考え方の育成や，知識を身に付けるだけではなく，どのように利用するのかまで思考できるように基本的な知識と技能の習得などが変更点として挙げられている。２点目としては情報Ⅰという教科を新設することにより，プログラミング，ネットワーク，情報セキュリティ，データベースの基礎的な内容が必修化となる。中学校でプログラミングの基礎を学んでいるので，高等学校ではコンピュータの仕組み，モデル化とシミュレーション，最適なアルゴリズムの構築など，より実践的なプログラミング教育が実施される。なお，情報Ⅰは必修化される。

３．プログラミング的思考

学習指導要領にも記述があったがプログラミング教育の最初の学びとして，小学校では

プログラミング的思考を身に付ける事を重要視しているが，このプログラミング的思考を身に付けているとプログラミングを実際に作成する際にスムーズに理解することが可能なのか不明である。このプログラミング的思考は計算機的思考と呼ばれ，1950年代頃にあった数学的な手法で，抽象的なデータを論理的に整理し表現する方法が基であったと言われており教育での利用とは関係のない考え方だった。コンピュータサイエンスの分野における考え方を教育業界で注目を集めるようになったのが2006年にコロンビア大学の Jeannette Wing によって発表された “Computational Thinking”［8］であり，コンピュータ的思考を学び様々な授業で横断的に利用することで，子ども達は日常生活においても効率的な思考の基で生活を送ることが可能であると発表した。これによりコンピュータサイエンスとして構築された考え方が教育の考え方に取り入れられ，この様な思考力がコンピュータサイエンス分野のみにとらわれることなく将来的にどのような職業にも必要な論理的な思考として定義された。その結果，このような考え方を新学習指導要領ではプログラミング的思考として用いられている。しかし，小学校で学ぶ内容としてこれらの説明でイメージがしにくいと思われる。そのため文部科学省はプログラミングの手引き［9］を学習指導要領とは別に配布している。このプログラミングの手引内には，具体的なプログラミング的思考のイメージとして図 3.1 があり，このような学習を実施するためにはどのように学習を行うのか，コンピュータを用いない場合の指導の考え方，教材選定の観点など細かく記述がある。



※文部科学省“プログラミング教育の手引（第三版）”抜粋

図 3.1　プログラミング的思考を働かせるイメージ

このプログラミングの手引にある記述に従い，多くの小学校ではプログラミング的思考を学ぶためにアンプラグドプログラミング，ロボット・ブロックプログラミング，ビジュアルプログラミングの 3 つの手法の中から 1 つのツールを使い学習を実施している。以下の各節でこれらツールの説明を記述する。

3.1　アンプラグドプログラミングとは

小学３年生は大人がイメージする以上に細かな作業が苦手な児童が多くいる。そのような児童に対し，初めて触れるコンピュータを思い通りに使えない状態でプログラミング教育を実施した場合，コンピュータとプログラムの両方に苦手意識を持つ可能性がある。これが小学生に対してなら経験を積んだ大学生以上に強い苦手意識となる恐れがある。そのようなことにならないようにするため，コンピュータを使わずにプログラミング的思考を学ぶ手法がアンプラグドプログラミングである。このアンプラグドプログラミングが有名になった切っ掛けがフィンランドのプログラマーであるリンダリカウスが書いた書籍「ルビィのぼうけん」[10] である。アンプラグドプログラミングでは自分が経験したことを漫然と実施してきた行動に対して論理的思考を身に付けるために細かく手順を考え，並べ，コンピュータを使わないことでグループワークとして実施することに向いており，選択する教材によっては学習指導要領で「総合的な学習の時間」に求められている教育用の枠を超えた積極的・総合的な学習と同時に探究的な学習や共同的な学習を実施することも可能である。アンプラグドプログラミングについてはあまり難しく考える必要はなく，「朝起きてから小学校に行くまでを考えよう」として工程を順番化し，その一つの工程をさらに細分化していくなどでプログラミング的思考を身に付ける。

3.2　ロボット・ブロックプログラミングとは

世界中の様々な企業がプログラミング的思考を学ぶためにロボットを使った学習ツールが販売されている。販売されているロボットによって様々な特徴はあるものの基本的には事前に何か目的を設定し，その目的を達成させるために用意されているパーツでロボットを自分で設計・作成し，ロボットに対して命令を与える事で目的を達成さることができる。対象年齢が低いロボット・ブロックプログラミングではロボットを組み立てる仕組みが無かったり，ロボットへの命令がカードなどで簡略化されている。ロボット・ブロックプログラミングでは制作者が目的達成を目指して試行錯誤することが可能である。これは学習指導要領にある探究的な学習を体現しながらプログラミング的思考を学ぶことを可能としている。設定された目的に対し，情報を収集し目的を達成するための分析ののち自分が設計した最適なロボットを作成するという体験ができ，目的達成までトライ＆エラーを繰り返すことで最終的に目的を達成させることでプログラミング的思考を身に付ける。

3.3　ビジュアルプログラミングとは

Web アプリケーションに近いイメージでブラウザを使って視覚的にプログラミングを実行する。実行できる種類はプログラミング的思考を必要とするアプリケーションが用意されており実行していくタイプか，ビジュアルブロックを組み合わせることで自分が思考したアプリケーションを作成するタイプに大別できる。両タイプとも基本的にはドラック＆ドロップのような簡単な操作のみで実行が可能でキャラクターを動かしたり，ゲームを作ったりすることが可能である。また，最大の特徴として PC やスマートフォンなどイン

ターネットを利用できる環境下にあれば無料で利用することが可能である。ビジュアルプログラミングは本章で説明した 3 種類の中で最もテキストで実施するプログラミングに近く，殆どのビジュアルプログラミングでは日本語での利用が可能なため関数の知識が無くてもプログラミングのように思考して操作することで自然とプログラミング的思考を身に付けることが可能である。また，3.2 節で説明したロボット・ブロックプログラミングのうちロボット作成はできないが，その他のプログラミング的思考を学ぶ手法についてはバーチャル世界で再現されている。

4．プログラミング的思考とプログラミング力の関係

現在の小学校でプログラミング的思考を学ぶために用いられている手法については 3 章で説明したが，プログラミング的思考と実際にプログラミングを組むプログラミング力は関係しているのかについては先行研究として明確な報告はなく，実際にはプログラミング的思考についての定義や説明も，学習指導要領で定義した以外にも各研究者により異なった状態となっている。[11] [12] [13] [14] そこで本章では逆説的にプログラミング力の高い場合，プログラミング的思考も高いのか実際に実験を実施し，その関係性について確認してみた。

4.1　プログラミング力の定義

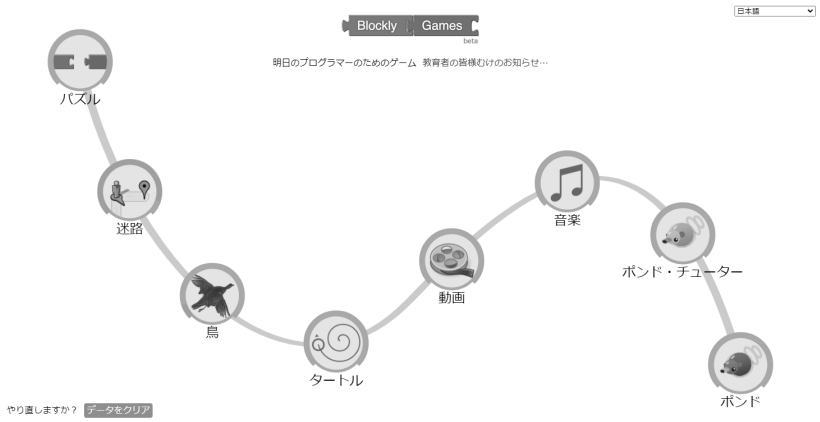
学生へのアンケートでプログラミングの経験などを確認すると，多くの学生は「できません」「授業やったことはあるが忘れました」という回答が多くあるので，実際に33人の学生に対して表 4.1 に示した 3 種類のプログラミングを作成させることでプログラミングの作成レベルがどの程度あるのかレベルを分ける事とした。実験開始時のプログラミングに対する知識量・技術料の差がないようにするため，全員に 9 回にわたり同じプログラミング教育を事前に実施した。

表 4.1　プログラミング力レベル分け

レベル	人数	プログラミング内容
なし	3 人	支持された 3 種類のプログラミングは何も作成できなかった。
初級	8 人	キーボードから入力した文字を画面に表示させるプログラミングのみ作成できた。
中級	18人	初級のプログラミングと 1 からキーボードで入力した数字までの合計値を計算して画面に表示させるプログラミングを作成できた。
上級	4 人	初級，中級のプログラミングと 1 から100までの数字を使いランダムに足し算計算を学ぶためのアプリケーションを作成できた。

4.2　プログラミング的思考の測定

プログラミング的思考を身に付けるための学習方法として，3 章で説明したように 3 種類のツールを使った学習方法があるが，その中で Google LLC が無料で提供している図 4.1 に示したような Blockly Games というビジュアルプログラミングを体験できるサイトがある。



※https://blockly.games/?lang=ja 参照

図 4.1　Blockly Games の TOP 画面

この Blockly Games には図 4.2 に示したようにブロックを組み合わせて 8 種類のゲームをクリアすることでプログラミング的思考を身に付ける事を目的としたサイトで，各ゲームはクリアすると同じ種類のゲームで難易度が上がったゲームが開始される。ゲームの種類によって用意されている難易度数は異なるが，全てのゲームをクリアするのにはそれなりの苦労が必要となる。

図 4.2　Google Blockly のゲーム画面抜粋

この 8 種類のゲームを 3.1 節でレベル分けをした全学生に実際に実施してもらい，ゲームのクリア率とゲームプレイ時間を計測してもらった。結果の平均値を表 4.2 に記述する。

表 4.2　プログラミング的思考の測定結果

レベル	トータルゲームプレイ時間	クリア率
なし	4 時間40分	88%
初級	3 時間54分	92%
中級	3 時間30分	94%
上級	3 時間28分	94%

実際にプレイした学生に対しては，実施する前に以下の 3 点について説明をした。

- 8 種類のゲームについて全てのレベルのゲームをクリアする必要はない。
- Blockly Games には保存機能があり，ゲームプレイ期間は 1 週間と設定しているのでいっぺんに全ゲームをクリアする必要はない。
- ゲームをプレイする順番は決まっていないので好きな順番でプレイをしても良いが，遊んだ回数は記録しておくこと。

4.3　プログラミング的思考の定義

今回はプログラミング的思考を学ぶ教材の中でビジュアルプログラミングを選択した。その結果，プログラミング力が高い方が若干クリア率が高い結果になったが，基本的には殆どの学生が高いクリア率を維持していることが分かった。これは大学生まで教育を受けてきている過程でプログラミング的思考が身についているのではないかという事が言える。また，トータルゲームプレイ時間を確認すると，ここではプログラミング力の差によって最大 1 時間以上の開きがあることが分かった。この結果からプログラミング力が高い学生はプログラミング的思考において必要な状況を精査・処理する速度が早い傾向にあることが分かった。また，実験終了後に Blockly Games のクリア状況の報告と簡単なコンピュータ利用に関するアンケートを取ったが，コンピュータの利用頻度について違いが顕著に出た。レベルが「なし・初級」の学生は PC 利用について，自宅にパソコンはあるが全く使っていない，若しくは自宅に PC がないと回答した学生が多かった。また，「中級・上級」の殆どの学生は自宅に PC があり，多くの作業や動画を見るなど趣味に対しても PC を使っていることが多いことが分かった。これらの結果から，大学生ではプログラミング的思考は日々の生活である程度身についており，プログラミング的思考の処理の速さがプログラミング力の高さとつながっていることが分かった。また，アンケート結果から学習成果の指標として，PC などに触れている時間やプログラミングに触れている時間が長いほど中級・上級のレベルにいる事が分かったので，プログラミング的思考の学習項目にコンピュータに長く振れている経験を指標に取り入れることが重要だと考える。

５．考察

本論文では実験を通してプログラミング的思考とプログラミング力についての関連性について確認した。プログラミング的思考が高い場合はプログラミング力も高いことが分かったが，プログラミング的思考は大学生になるまでに特殊な訓練をしなくても，ある程度は身につくことが分かった。そのため重要視する点はプログラミング的思考で自分の考えを構築する処理速度が大きくかかわっていることが分かった。そのため小学校で実施するプログラミング教育については，グループワークで学習を実施する際に図 3.1 にあるように問題点の洗い出しなどの繰り返し学習に時間を掛けるよりは，チームで競わせることや異なる教材によって同じ内容の学習を実施することで多くの体験をさせるなど，作業方法の経験を多く積ませる授業計画にしたほうが，最終的な IT 人材の育成につながるプログラミング力の強化には有用である。また，PC に触っている時間が長い方がプログラミング力の向上に繋がる事もわかったので，３章で説明したプログラミング的思考を学ぶためのツールとしてはアンプラグドプログラミングはあまり適さないと言える。その点を考えると３章で説明したプログラミング的思考を学ぶツールは以下のような不等式が成り立つ

ビジュアルプログラミング≧
ロボット・ブロックプログラミング＞
アンプラグドプログラミング

現在，教育環境の ICT 化が進められているが，授業で児童・生徒が利用するツールは教室の広さなどの環境や価格の面からタブレットを選択する傾向にあるが，アンケート結果からスマートフォンは全対象者が毎日多くの時間利用していることが分かっている点から，IT 人材の育成という面からは適していないことが分かる。ただし，タブレットの教育利用については ICT 教材の教育効果や，学習方法の多様化など多面的な評価が必要だと思われるが本論文においては，これらの点については論じていないので検討外とする。

［参考文献］

［１］ 独立行政法人情報処理推進機構社会基盤センター，“IT 人材白書2020”，2020年８月31日
［２］ 文部科学省，“小学校学習指導要領（平成29年告示)”，2017年３月
［３］ 文部科学省，“小学校学習指導要領（平成29年告示）解説 総合的な学習の時間編”，2017年３月
［４］ 文部科学省，“中学校学習指導要領（平成29年告示)”，2017年３月
［５］ 文部科学省，“中学校学習指導要領（平成29年告示）解説 技術・家庭変”，2017年３月
［６］ 文部科学省，“高等学校学習指導要領（平成30年告示)”，2018年３月
［７］ 文部科学省，“高等学校学習指導要領（平成30年告示）解説 情報辺”，2018年７月

［8］ Wing, Jeanetta M, “Computational thinking”, Communications od the ACM, Vol 4 no 3 pp. 33-35, Mar. 2006
［9］ 文部科学省，“小学校プログラミング教育の手引（第三版)”，2020年 2 月
［10］ リンダ リカウス 鳥井由紀，“ルビィの冒険 こんにちは！プログラミング”，翔泳社，2016年 5 月19日
［11］ 小林祐紀 兼宗進，“コンピュータを使わない小学校プログラミング教育ルビィの冒険で育む論理的思考”，翔泳社，2017年 4 月 1 日
［12］ 尾崎光 伊藤陽介，“小学校におけるプログラミング教育実施上の課題”，鳴門教育大学情報教育ジャーナル No 15(1)，pp. 31-35，2017年12月
［13］ 藤原伸彦 阪東哲也 曽根直人 長野仁志 山田哲也 伊藤陽介，“ティン化リングとしてのプログラミング”，鳴門教育大学情報教育ジャーナル No 16，pp. 21-26，2019年 3 月
［14］ 遠藤諭，“プログ民具的思考が分からない”，情報処理62巻 3 号，pp. 130-133，2021年 2 月15日