初等中等教育におけるプログラミング教育の教育的効果についての考察

The Effects of Programming Education in Elementary and Secondary Education

中　植　正　剛

要　旨

初等教育でプログラミング教育が必修化されることとなった。しかしながら、児童・生徒に育む資質や能力、つまりその教育的効果については充分に明確化されておらず、広い合意が得られているとは言いがたい。本稿ではこの点を課題として捉え、今後のプログラミング教育の推進および教員養成における指導力の育成に資するための基礎的な知見を得ることを目的として、先行研究や先行する見解を整理・検討しつつ、プログラミング教育の教育的効果を明確化するための考察をすすめた。

その結果、教育的効果として、「コンピュータサイエンスの力が身につく」、「問題解決のプロセスをコントロールする力が身につく」、「コンピュータでなければできない（あるいは実現が難しい）表現力が身につく」、「他者と創造的に関わりながら、アイデアを整理したり、表現したり、共有したりする力が身につく」、「コンピュータやテクノロジーに対する新たな視点が身につく」、「様々な学問分野の原理を理解したり活用したりする力が身につく」、「社会におけるコンピュータの役割についての理解が深まる」、「自信と責任を持って創造的に情報技術を活用する情意や態度が育つ」ことが示された。

これらの定義を用いると、議論のとりまとめと次期学習指導要領の答申で述べられている資質・能力が充分に説明できるとともに、プログラミング教育の教育的効果として頻繁に提案される問題解決能力や論理的思考力について、より具体的な説明ができることがわかった。

キーワード：プログラミング教育　教育的効果　初等中等教育　教員養成　資質・能力

1．背景

2016年、初等教育におけるプログラミング教育の必修化が決定した。その３年前の2013年６月、日本経済再生本部による「日本再興戦略－ Japan is Back－」が閣議決定され、このなかで、産業競争力の源泉となるハイレベルなIT人材の育成と確保を重点施策として推進することが表明されるとともに、その一環として義務教育段階でのプログラミング教育の推進が示された。これと同日、高度情報通信ネットワーク社会推進戦略本部（IT戦略本部）による「世界

神戸親和女子大学　発達教育学部　児童教育学科　準教授

最先端IT国家創造宣言」の初版が閣議決定された。同宣言には一年ごとに変更が加えられているが、初版では、高度なIT人材の育成の一環として、初等・中等教育段階におけるプログラミング教育の推進が表明されている。2014年版からは、高度なIT人材の育成だけでなく、「国民全体の情報の利活用力向上を実現」する際に重要なこととして、「初等・中等教育段階におけるプログラミングに関する教育の充実に努め」るとともに、「ITに対する興味を育む」こと、「IT を活用して多様化する課題に創造的に取り組む力を育成すること」を強化することが記載された。このような流れを引き継ぐ形で、2016年 4 月の経済産業力会議において、「第 4 次産業革命に向けた人材育成総合イニシアチブ」が文部科学大臣によって示された。これは、初等中等教育や高等教育から研究者レベルまでの包括的な人材育成総合プログラムを体系的に実施するという計画であり、初等中等教育においては、次代に求められる情報活用能力の育成の具体的な在り方として、各教科の特性に応じたICTの活用とともに、発達段階に即したプログラミング教育を必修化することが挙げられている（経済産業力会議　2016）。2016年 6 月には、2020年度から施行される次期学習指導要領に向けて、「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」（以下、有識者会議）から「小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）」（以下、議論の取りまとめ）が発表され、学校におけるプログラミング教育の在り方について、どのような資質や能力をどのように育成するのかについての提言がされている。議論のとりまとめでは、プログラミング教育を通して培う資質や能力についての一定の提案はあるものの、同資料のなかで「これらの資質・能力の在り方を踏まえつつ、子供たちに求められる普遍的な力とは何かを明確にし、認識の共有を図っていく必要がある」と記述されているように、資質や能力のさらなる明確化が課題として残っている。

政府によるプログラミング教育に関する議論の一方で、教育関係団体（NPO法人や民間教育事業者）によるプログラミング教育が先行して全国で実施されてきた。「プログラミング人材育成の在り方に関する調査研究　報告書」（総務省 2015）では、全国43の教育関係団体が確認されている。このうち25団体については、古くは1999年からプログラミング教育に取り組んでいる一方で、約 6 割の団体が、初等中等教育段階でのプログラミング教育の推進が閣議決定された2013年以降にプログラミング教育を開始している。受講生の年齢については、小学校 4 年生～ 6 年生が最も多く、ついで小学校 1 年生～ 3 年生と中学生になっており、この点においても、教育関係団体によるプログラミング教育が初等中等教育におけるプログラミング教育の必修化に向けた政府の議論と同調するものであることが確認できる。教育関係団体による取り組みで利用されているプログラミング言語は、Scratchが最も多く、JavaScript、Java、レゴマインドストーム、Python、Viscuitなどが挙げられている。本稿の考察対象であるプログラミング教育の目的や育成すべき資質・能力の明確化については、同報告書でも、「定義を明確にする必要があるという有識者の意見が強かった」ことが示されており、これが課題として残

されていることが確認できる。

以上の背景と課題を踏まえ、本稿では、先行研究や先行する見解を整理・検討しつつ、どのような教育的効果をプログラミング教育のねらいとするのかについて考察をすすめることで、今後のプログラミング教育の推進および教員養成における指導力育成に資するための基礎的な知見を得たい。なお、プログラミング教育による教育的効果については、汎用的な問題解決能力や論理的思考力やコミュニケーション力が身につくという言説が頻繁にみられる。様々な教育の分野において、適切な学習方法を選択すればそれらの汎用的な力が身につくものであるが、本稿では、特にプログラミング教育という領域に固有の教育的効果にできるかぎり焦点をあてて論をすすめたい。

2．先行研究の調査

2.1.「コンピュテーショナル・シンキング」の検討

有識者会議による議論のとりまとめ(2016)においては、「プログラミング的思考」について、「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」と述べられている。コンピュテーショナル・シンキングは、ジャネット・ウィングによって2006年に提唱されて広まった概念であり、次のように定義されている（Wing 2010）。

> Computational Thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.

つまり、「問題やその解法を、コンピュータが効果的に処理できる形式で表現するための思考のプロセス」のことである。これは論理的思考やシステム思考と重なるものであるとされている。ここでいう「問題」とは、数学の問題のように既に明瞭に定義された問題から、実世界における複雑で明瞭に定義されていない問題まで、多様な問題を含んでいる。ウィングによれば、最も高い次元のコンピュテーショナル・シンキングは現実の問題を抽象化する力であり、これは、問題の解決とは関係のない要素を排除して本質的な要素だけを抽出したり、その問題にとって何がインプットで何がアウトプットなのかを分析したりする思考などのことを指す。さらに、プログラミング言語の特性を考慮しながら問題を抽象化するというようなことも含まれている。

コンピュテーショナル・シンキングは、オーストラリアや米国、英国において取り入れられているものの、その定義については、各国がそれぞれの解釈をしている。プログラミング教育が教科コンピューティングとしてナショナルカリキュラムに取り入れられている英国（イングランド）では、コンピュテーショナル・シンキングと創造性を育むということが目的として掲げられており、それについて次の４つのねらい（Aim）が定められている（National

curriculum in England: computing programmes of study 2013　筆者訳。E-１～E-４のナンバリングは筆者付加)。

E-１．コンピュータサイエンスの基本的な原理や概念を理解して使いこなすことができるようになる（抽象化、論理、アルゴリズム、データ表現を含む)。

E-２．コンピューティングの専門用語を使って問題を分析することができるようになるとともに、問題解決のためにプログラミングの経験を積む。

E-３．新しい情報技術や、あまりよく理解されていない情報技術を含む、さまざまな情報技術を評価したり活用したりすることができるようになる。

E-４．情報技術とコミュニケーション技術について､責任感があり､競争力があり、自信があり、創造的に活用できるような利用者になる。

一方、ブレナン&レズニックは、コンピュテーショナル・シンキングを次の三つに分類して定義している（Brennan & Resnick 2012, BR-１～BR-３のナンバリングは筆者付加)。

BR-１．コンピュテーショナルコンセプト（Computational Concepts）

BR-２．コンピュテーショナルプラクティス（Computational Practice）

BR-３．コンピュテーショナルパースペクティブ（Computational Perspective）

BR-１は、プログラマー（ブレナン&レズニックはデザイナーと表現している）が、意図する動作をプログラムに置き換える際に活用される概念であり、手続き型言語では、順次・繰り返し・並列処理・イベント・条件分岐・比較演算・データ構造などの概念がそれにあたる。BR-２は、プログラマーが実践のなかで発達させてゆく力のことである。プログラミングによる問題解決は一方向の単純なプロセスではなく、スモールステップで少しずつ積み上げながら(incremental)、問題解決の手順を行きつ戻りつするような反復性のある（iterative）プロセスであるが、そのようなインクレメンタルで反復的な問題解決の実践力がコンピュテーショナルプラクティスに含まれる。さらに、テストやデバッグをして問題を解決する力や、他者のつくったプログラムを再利用したり（reuse）改良を加えたり（remix）する力、問題を抽象化したりモジュール化したりする力がここに含まれている。BR-３は、プログラマーがこの世界や自分自身に対して持つ世界観や視点のことで、例えば、メディアを表現の道具であるとみなしたり、問題解決をするにあたって他者とのコネクションを作ったり、他者と関わったりすることである。ブレナン&レズニックは、コンピュテーショナルパースペクティブを、「表現すること」(Expressing)、「つなぐこと」(Connecting)、「問うこと」(Questioning）という３つの次元で説明している。「表現すること」とは、コンピュータのような双方向のメディアを消費の対象として捉えるのではなく、何かをデザインしたり自己表現のために活用したりできるものとして捉える視点のことである。「つなぐこと」とは、プログラミングによる創造活動を社会的な実践であると捉えて、他者とともに、あるいは他者のためにプログラムをつくるもの

だと捉える視点である。「問うこと」とは、テクノロジーを所与のものとして捉えるのではなく、自らが作り変えたり影響を与えたりできるものだと捉える視点のことである。

2.2. ビジュアルプログラミング言語開発者らによる言説

発達段階に応じたプログラミング教育を実施するにあたり、初等教育ではScratch, Scratch Jr., Viscuitなどのビジュアルプログラミング言語を多用することが予想される（例えば、唐沢（2016b）、山本ら（2016）、総務省（2015）、丸山（2014））。そこで、ビジュアルプログラミング言語の開発者であり、プログラミング教育の実践者でもあるミッチェル・レズニック、阿部和広、原田康徳による教育的効果に関する言説を検討する。

Scratchの提案者であり開発者でもあるミッチェル・レズニックは、プログラミングを技術的なスキルセット（“a set of technical skills”）ではなく、新しいタイプのリテラシーや表現手段（“a new type of literacy and personal expression”）であると捉えている。レズニックらによって開発されたScratchは、ブロックを組んでコーディングをするタイプのビジュアルプログラミング言語で、キャラクターなどを動かしてアニメーションやゲームをつくることができる。Scratchの大きな特徴は、作成したプログラムをScratchのサイトで共有したり、他者によって共有されたプログラムを実際に試してコメントを投稿したり、他者のプログラムの中身を確かめたり、それに対して修正や拡張を加えたりすることができるソーシャルな機能が備わっていることである。この機能を活用することで、コーディングによる表現を通じて、他者とともにアイデアを整理したり、表現したり、共有したりすることができる（“We see coding as a new way for people to organize, express, and share their ideas.”）（Resnick 2015）。つまり、レズニックの言うリテラシーは、プログラムを読んだり書いたりすることだけを指しているのではなく、プログラミングを通して自己や他者や世界と関わるための力として位置づけられている。また、レズニックは、プログラミングを通して、コーディングを学ぶ以上の「深くて広い学び」ができるとしているが、それは算数・数学的な知識やコンピュータに関する知識を学ぶことにとどまらず、問題解決の方略や計画を立てることやコミュニケーションのアイデアが学べることであり、そこにプログラミング教育を行う理由があるのだと述べている（Resnick 2013, Resnick 2015）。

Scratch日本語版の開発者である阿部和広は、レズニックと同様、プログラミングを「類まれな表現手段」であると捉えている（石坂 2016, 阿部・豊福 2016）。用途が固定されたピアノや絵の具などと違い、コンピュータはプログラムによって多様な機能をもたせることができ、多様な用途に用いることができるという点で、「他の表現手段やメディアにない特徴」を持っているという。阿部は、プログラミングそのものの持つおもしろさを動機として、学習者が自ら探求的にプログラムをつくっていくなかでプログラムに必要とされる様々な知識を学ぶきっかけになるのではないかと考えており、この説明として、シューティングゲームの弾の到達時

間を予測するのに「速さ・時間・距離」の問題について考えることで、単に公式から答を導き出すだけではない深い理解が得られるのではないかという例を挙げている（石坂 2016）。 また、プログラミング教育について、高度IT人材の育成は重要な課題であるものの、目的への過度な最適化についての懸念を示しており、プログラミングの目的としてミッチェル・レズニックが挙げるCode to Learn（学ぶためにコーディングをすること）に共感を示している。

Viscuitの開発者である原田康徳は、プログラミング教育を「実用的なプログラミングスキルを書くためのスキル教育だけではなく、情報化社会を生き抜く上で広く全員が知るべき教養的な位置づけ」として捉えており、コンピュータと無関係に生活することが不可能である現在、すべての人がコンピュータとは何かという基本原理を理解することが必要であり、これを義務教育でプログラミング教育を実施する理由として挙げている（原田 2015）。原田はプログラミングを体験させる理由として、コンピュータのわかりにくさを挙げている。それは、コンピュータは動いているところが見えず、さらに、先行する他の発明品で例えることが難しいというわかりにくさである（原田 「コンピュータってなに？」）。さらに原田は、コンピュータとは何かという基本原理の理解ということを、特に「プログラムを書くことによってしか得られないコンピュータ観」に焦点化しており（原田 2016）、ソフトウェアやアプリケーションの利用者とは違った視点でコンピュータの理解を深める必要があるのだと説明している（唐沢2016a）。例えば、「プログラムは何度でも使われ、プログラムは簡単に修正でき、一斉に壊れる」ことの理解、「単純な命令の組合せで、複雑な動きができている」ことの理解、「情報はすごい勢いで拡散する」という情報の原理の理解、「プログラムを作るということは、人間が理解している意味を、コンピュータが分かる単純な意味へ分解する作業」であることの理解、「シミュレーションは簡単に実験できる場」であることの理解、「脳の拡張としてのコンピュータ」の理解（原田 2016）などが挙げられている。

原田はまた、プログラミング教育を逐次処理、繰り返し、条件分岐を教えることに矮小化されつつある風潮に懸念を示しつつ、「作りたいものがあってそれを作る」その行為全体の力をつけることがプログラミング教育の目的であると述べている（原田 「繰り返しとか条件分岐とか」）。彼は特に、「作りたいものがあって」という部分に着目し、多くの場合は「作りたいものがその人の中に全く無いか、ありすぎるか」のどちらかであり、それに対して「コンピュータで作れそうな」作りたいものを適切なレベルで思い浮かべる能力を鍛えることを提唱している。これは、コンピュータやプログラミングについての理解に加えて、「自分に足りない部分を知る」という、自己の能力への評価と理解が伴うものである。

原田の開発したViscuitは、JavaやScratchなどの手続き型の言語ではなく、宣言型の書き換え型プログラミング言語で、画面上に配置したオブジェクトに対してメガネという仕組みをつかって単純な動きを実現し、複雑な動きを単純な動きの組み合わせで実現するというプログラミング環境である。Viscuitは、ビジュアルプログラミング言語の多くが手続き型の言語であ

るという現在の状況において独特の位置づけをもつ存在である。Viscuitのような手続き型言語とは異なるパラダイムのプログラミング言語に触れることによって、コンピュータを違った角度から理解できるとともに、コンピュータという存在の奥深さや幅広さが経験できる。この点に関連して、阿部は、有識者会議での議論が手続き型言語のみであることを指摘し、「複数のパラダイムのうち、ひとつだけしか明示していないことは大きな問題」だと述べている（石橋 2016）。

2.3. 議論のとりまとめから

有識者会議の議論のとりまとめにおけるプログラミング教育の目的に関する記述は以下のように要約できる（Ｍ１～Ｍ５のナンバリングは筆者による）。

Ｍ１．コンピュータの働きを理解しながら、自らの問題解決にどのように活用できるかをイメージし、意図した処理がどのようにすればコンピュータに伝えられるか、コンピュータを介してどのように現実世界に働きかけることができるかを考えること。

Ｍ２．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合せが必要であり、一つ一つの動きに対応した記号を、どのように組み合わせたらいいのか、記号の組合せを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、といったことを論理的に考えていく力が必要となる。この力を「プログラミング的思考」と呼ぶ。

Ｍ３．時代の変化によって将来的にプログラム言語や技術が変化をしても、社会で果たすコンピュータの役割を理解しながら「プログラミング的思考」を発揮し、その時代の情報技術を効果的に活用して問題を発見・解決していけるような、時代を超えて必要となる資質・能力を身につけること。

Ｍ４．コンピュータはプログラミングを通して人間の意図した処理を行わせることができる機械であると理解すること。

Ｍ５．情報技術によるサービスを受け身で享受するだけでなく、その働きを理解して、自分が設定した目的のために使いこなし、よりよい人生や社会づくりに生かしていくこと。

また、これらの資質・能力を育むにあたっては、次期学習指導要領において強調される「主体的・対話的で深い学び」に資するプログラミング教育にすることが重要であるとされている。さらに、小学校段階でのプログラミング教育の目的について、上記は次のように整理されている（ＭＥ１～ＭＥ３のナンバリングは筆者による）。

「子供たちに、コンピュータに意図した処理を行うよう指示することができるということを体験させながら」

ＭＥ１．身近な生活でコンピュータが活用されていることや、問題の解決には必要な手順があることに気付くこと。

ＭＥ２．各教科等で育まれる思考力を基盤としながら基礎的な「プログラミング的思考」を

身に付けること。

ME３．コンピュータの働きを、よりよい人生や社会づくりに生かそうとする態度を涵養すること

これらを、次期学習指導要領に向けて「三つの柱（（１）何を理解しているか、何ができるか（知識・技能)、（２）理解していること、できることをどう使うか（思考力・判断力・表現力)、（３）どのように社会・世界と関わり、よりよい人生を送るか（学びに向かう力、人間性等)）で整理していくとされていること等に留意することが必要」であると示されている。

３．プログラミング教育の教育的効果

以上を踏まえて、プログラミング教育による教育的効果を整理してまとめた。

表１　「プログラミング教育の教育的効果」

プログラミングによって実現可能（あるいは実現が容易）となる問題解決や表現活動を体験することによって

Ｎ１：コンピュータサイエンスの力が身につく Ｎ１-a：コンピュータの仕組みや原理の理解 Ｎ１-b：問題の抽象化やモジュール化の力 Ｎ１-c：意図した動作や表現をプログラミングによって実現できる力（アルゴリズムやデータ表現を含む）
Ｎ２：問題解決のプロセスをコントロールする力が身につく Ｎ２-a：自らの力量を省みながら、コンピュータによって実現可能な問題解決を想起する力 Ｎ２-b：スモールステップや反復的な問題解決の実践力 Ｎ２-c：テストとデバッグの実践力 Ｎ２-d：他者による問題解決の成果を再利用したり改良したりする実践力 Ｎ２-e：問題解決に必要な情報技術を新たに学んで活用する実践力
Ｎ３：コンピュータでなければできない（あるいは実現が難しい）表現力が身につく
Ｎ４：他者と創造的に関わりながら、アイデアを整理したり、表現したり、共有したりする力が身につく
Ｎ５：コンピュータやテクノロジーに対する新たな視点が身につく Ｎ５-a：コンピュータをデザインや自己表現のために活用できるものと捉える視点の獲得 Ｎ５-b：プログラミングを社会的な実践であると捉える視点の獲得 Ｎ５-c：テクノロジーを、自らが作り変えたり影響を与えたりすることができるものだと捉える視点の獲得
Ｎ６：様々な学問分野の原理を理解したり活用したりする力が身につく
Ｎ７：社会におけるコンピュータの役割についての理解が深まる
Ｎ８：自信と責任を持って創造的に情報技術を活用する情意や態度が育つ

まず、コンピュータサイエンスの力が身につくという分類（Ｎ１）であるが、これは、E-１、E-２、BR-１、および、原田の見解を整理したものである。「Ｎ１-b 問題の抽象化やモジュール化」については、ブレナン＆レズニックの定義では「BR-２コンピュテーショナルプラクティス」に分類されているが、この分類を詳細に検討すると、「問題の抽象化やモジュール化」以外の３つの項目については問題解決のプロセスをメタにコントロールする力に焦点があてられ

ているのに対して、「問題の抽象化やモジュール化」はそれらとは異なる。そこで、「問題の抽象化やモジュール化」をＮ１に含めることとし、「Ｎ２問題解決のプロセスをコントロールする力が身につく」という分類をＮ１とは別に作って残りの３つの項目を含めることとした（Ｎ２-b, Ｎ２-c, Ｎ２-d)。また、Ｎ２には、原田の述べる「作りたいものがあってそれを作る」という行為全体のうち、「コンピュータで作れそうな」作りたいものを適切なレベルで思い浮かべる力をＮ２-aとして加えるとともに、E-３の新たな情報技術を必要に応じて評価したり活用したりする力をＮ２-eとして加えた。

原田や阿部の指摘を踏まえて留意したいのが、Ｎ１は手続き型言語だけを考慮したものではなく、宣言型言語などの多様なパラダイムを考慮しているということである。特に、「Ｎ１-c意図した動作をプログラミングによって実現できる力」については、手続き型言語における構造化プログラミング（順次・繰り返し（反復）・条件分岐（分岐））の理解がその具体例として挙げられることが頻繁にあるが、他のパラダイムにおけるＮ１-cの具体的な在り方についての明示が望まれる。これについては今後の研究に委ねる。

レズニックや阿部が述べているように、コンピュータを新しい表現手段として捉えるならば、プログラミングをすることによってしか実現できない（あるいは実現が飛躍的に容易になる）表現活動や問題解決活動を体験することで、コンピュータでなければできない（あるいは実現が難しい）表現力の育成を図ることができる。これをＮ３とした。Ｎ４では、レズニックの言うリテラシーのうち、プログラミングを通して世界や他者と関わることを反映した。ブレナン&レズニックによるコンピュテーショナルパースペクティブの「表現すること」「つなぐこと」「問うこと」については「Ｎ５コンピュータやテクノロジーに対する新たな視点が身につく」としてまとめた。レズニックや阿部の「学ぶためにプログラムをする（code to learn)」については、人文科学や社会科学や自然科学や芸術などといった様々な学問の原理に関わるきっかけとなったり、理解を深めたり、それらを活用する力を身につけることとして、Ｎ６とした。Ｎ７には、Ｍ３のうち「社会で果たすコンピュータの役割を理解する」を反映している。Ｎ８にはE-４を反映している。

表１に示した教育的効果は、それぞれの項目に挙げた力が個別に身につくものではなく、プログラミングを経験することによって複合的に身につく力である。また、当然のことながら、これらの教育的効果は学習方法に依存するものであり、どのような学習方法をとってもこれらの教育的効果が身につくというものではない。例えば、Ｎ４やＮ５-bについて考えると、「他者と創造的に関わる」という要素が学びのプロセスに存在しなければ、このような教育的効果をもたらすことはできない。

４．考察

表１で示されたプログラミング教育の教育的効果についての考察を加えたい。まず、表１と

議論のとりまとめの関連性について考察をすることで、表１が議論の取りまとめを充分に説明できるかどうかを検討する。議論の取りまとめのＭ１には、表１のＮ１とＮ２-aが対応している。Ｍ２の「プログラミング的思考」については、本稿執筆時に広く合意された定義が存在しないが、表１においては、Ｎ１-c、Ｎ２-b、Ｎ２-cが対応していると思われる。Ｍ３にはＮ７やＮ８が、Ｍ４にはＮ５-aが対応し、Ｍ５にはＮ５-aとＮ８が対応していると考えられる。小学校段階でのプログラミング教育の目的としてまとめられた記述については、ＭＥ１はＮ７、Ｎ２が、ＭＥ２にはＮ１-c、Ｎ２-b、Ｎ２-c、Ｎ６が、ＭＥ３にはＮ８がそれぞれ対応している。これらの考察から、表１は「議論のとりまとめ」で提示されている資質・能力を全て包含しており、表１を用いて議論の取りまとめで示された資質・能力が説明できることが確認できる。

また、背景でも述べたが、プログラミング教育の教育的効果については、論理的思考力や問題解決能力が身につくということが頻繁に挙げられる（例えば、山本ら（2016）、久野（2016）、総務省（2015）、松林（2015）、神谷（2015））。しかしながら、多くの記述では、論理的思考力や問題解決能力を汎用的能力として捉えているのか、プログラミングという領域固有の能力として捉えているのかが曖昧な場合が多い。領域固有の論理的思考力・問題解決能力と汎用的能力としての論理的思考力・問題解決能力との関係をどのように捉えるのか、領域固有の論理的思考力・問題解決能力が汎用的能力として転移するのか、あるいはそれはどのように転移するのかなどについて詳細に検討することを本稿の枠組みで行うことは難しいので、これらについては割愛するが、本稿の立場では領域固有の論理的思考力・問題解決能力に焦点をあてて教育的効果を提案しているので、この視点からプログラミング教育で育む問題解決能力と論理的思考力についての考察をしておきたい。

論理的思考力については、表１では、Ｎ１-b, Ｎ１-c, Ｎ２に関係が深いと思われるが、これらをみると、プログラミングの体験を通して培われる論理的思考力は、①解決しようとしている問題をプログラムという表現方法で表すために論理的に考えるという認知的能力（Ｎ１-b, Ｎ１-c）と、②問題解決のプロセス全体を論理的にコントロールするための認知的能力（Ｎ２）の二つに分類できることがわかる。

問題解決能力については、文部科学省が学習指導要領の策定やそのねらいの実現に向けて参照することの多いPISA2012の定義 が示すように、広範な能力を指しているため、表１で示された力のうち、問題解決とは直接関係がないと思われるＮ７を除くすべてが該当していると考えられる。

問題解決能力については、中央教育審議会（2016b）の次期学習指導要領の答申で示されている資質・能力の三つの柱である「①何を理解しているか、何ができるか（生きて働く「知識・技能」の習得）」「②理解していること・できることをどう使うか（未知の状況にも対応できる「思考力・判断力・表現力等」の育成）」「③どのように社会・世界と関わり、よりよい人生を送るか（学びを人生や社会に生かそうとする「学びに向かう力・人間性等」の涵養）」に基づ

いて整理する必要があろう。三つの柱のうち、①生きて働く「知識・技能」については、Ｎ１やN6, Ｎ７が対応している。②未知の状況にも対応できる「思考力・判断力・表現力等」の育成について、次期学習指導要領答申では次の３点にまとめられている。

１．物事の中から問題を見いだし、その問題を定義し解決の方向性を決定し、解決方法を探して計画を立て、結果を予測しながら実行し、振り返って次の問題発見・解決につなげていく過程

２．精査した情報を基に自分の考えを形成し、文章や発話によって表現したり、目的や場面、状況等に応じて互いの考えを適切に伝え合い、多様な考えを理解したり、集団としての考えを形成したりしていく過程

３．思いや考えを基に構想し、意味や価値を創造していく過程

この記述を検討すると、表１ではＮ２、Ｎ４が関係し、３についてはＮ３も関係するのではないかと考えられる。③学びを人生や社会に生かそうとする「学びに向かう力・人間性等」の涵養）については、「①及び②の資質・能力を、どのような方向性で働かせていくかを決定付ける重要な要素」とされており、次の２点が挙げられている。

１．主体的に学習に取り組む態度も含めた学びに向かう力や、自己の感情や行動を統制する能力、自らの思考の過程等を客観的に捉える力など、いわゆる「メタ認知」に関するもの。

２．多様性を尊重する態度と互いのよさを生かして協働する力、持続可能な社会づくりに向けた態度、リーダーシップやチームワーク、感性、優しさや思いやりなど、人間性等に関するもの。

これらについては、Ｎ２-aで示される「自らの力量を省みる」ことや、Ｎ４、Ｎ８に関連があると考えられる。さらに、指導要領の答申では、各教科等の特質に応じた「見方・考え方」についての言及がある。我が国のプログラミング教育は英国のコンピューティングのような教科化がされているものではないので、「教科の特質に応じた」という記述には当てはまらないかもしれないが、Ｎ５の「プログラミングを通じてコンピュータやテクノロジーに対する新たな視点を持つ」という効果を「見方・考え方」の育成として捉えることができる。

おわりに

本稿では、プログラミング教育で育むべき資質と能力についての定義が明確化されていないことを課題として、先行研究や先行する見解の整理・検討を踏まえながら、プログラミング教育においてどのような教育的効果が得られるのかを明確に提示することに取り組んだ。また、本稿で提示した教育的効果と「議論の取りまとめ」で提示されている資質・能力との関連や、次期学習指導要領の答申で提示されている資質・能力との関連を考察した。さらに、提示した教育的効果のなかで、論理的思考力や問題解決能力がどのように表現されているのかについて

の考察をすすめた。その結果、本稿で提示した教育的効果によって議論のとりまとめや次期学習指導要領で示されている資質・能力を充分に説明できることや、論理的思考力や問題解決能力をより具体的に説明できることがわかった。

初等教育のプログラミング教育については、実施する教科や学習内容が定まっておらず、カリキュラムマネジメントは各学校の実情に委ねられている。また、発達段階に応じてどのような力をどのような学習活動や学習場面で身につけるのかについての議論も充分ではなく、今後の系統的なカリキュラムの開発が待たれるところである。さらに、プログラミング教育を効果的に指導できる教員の養成に際して、どのような力を身に付けさせるべきなのかについての議論もこれからである。本稿は、プログラミング教育を推進するためにそれらを検討する際の基礎的資料となるものである。

参考文献

阿部和広，豊福晋平（2016）「子どもとプログラミング（特集：子どもの未来と情報社会の教育）」『智場』#120，53-65，国際大学グローバル・コミュニケーション・センター

阿部和広（2016）「子供の創造的活動とプログラミング学習」『情報処理学会誌』57（４），349-353

Brennan, K., Resnick, M.（2012）New frameworks for studying and assessing the development of computational thinking <http://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf>（2016.12.25最終閲覧）

中央教育審議会（2016a）「情報ワーキンググループにおける審議のとりまとめ」<http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo3/059/sonota/__icsFiles/afieldfile/2016/09/12/1377017_1.pdf>（2016.12.25最終閲覧）

中央教育審議会（2016b）「幼稚園、小学校、中学校、高等学校及び特別支援学校の学習指導要領等の改善及び必要な方策等について（答申）」<http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo0/toushin/__icsFiles/afieldfile/2016/12/27/1380731_00.pdf>（2016.12.25最終閲覧）

Department of Education（2013）National curriculum in England: computing programmes of study <https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study>（2016.12.25最終閲覧）

神谷加代&できるシリーズ編集部（2015）「子どもにプログラミングを学ばせるべき６つの理由」インプレス

唐沢正和（2016a）「子どもだけではなく全ての日本国民にとってプログラミングが重要である、たった１つの理由」『特集：小学生の「プログラミング教育」その前に（３）』atmarkIT <http://www.atmarkit.co.jp/ait/articles/1608/23/news019.html>（2016.11.9最終閲覧）

唐沢正和（2016b）「『プログラミング教育』はICTを活用した新たな“学び”のシンボル―小学校で成功させるためのポイントと実践事例」『特集：小学生の「プログラミング教育」その前に（４）』atmarkIT <http://www.atmarkit.co.jp/ait/articles/1609/15/news020.html>（2016.11.9最終閲覧）

原田康徳（2016）「小学生に分かるコンピュータサイエンスとしてのプログラミング教育―ビスケットを用いて―」『情報処理学会誌』57（４），344-348

原田康徳（2015）「プログラミング言語ビスケットを用いた基礎としてのプログラミング教育の提案と実践」『デジタルプラクティス』 6（2），105-111
原田康徳「コンピュータってなに？ハカセのコラム」
<http://www.viscuit.com/column01/>（2016.12.25最終閲覧）
原田康徳「繰り返しとか条件分岐とか」『ビスケット開発室』
<http://devroom.viscuit.com/2016/10/29/繰り返しとか条件分岐とか/>（2016.12.25最終閲覧）
久野靖（2016）「プログラミング教育/学習の理念・特質・目標」『情報処理学会誌』57（4），340-343
石坂貴明（2016）「プログラミングの本質と学びへの効果＜阿部和広さんインタビュー前編＞」
<http://berd.benesse.jp/special/manabi/manabi_8.php>（2016.11.9最終閲覧）
高度情報通信ネットワーク社会推進戦略本部（2016）「世界最先端IT国家総合宣言（改訂版）」
<http://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20160520/sengen_kaitei.pdf>（2016.12.1最終閲覧）
高度情報通信ネットワーク社会推進戦略本部（2015）「世界最先端IT国家総合宣言（改訂版）」
<https://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20150630/siryou1.pdf>（2016.12.1最終閲覧）
高度情報通信ネットワーク社会推進戦略本部（2014）「世界最先端IT国家総合宣言（改訂版）」
<http://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20140624/siryou1.pdf>（2016.12.1最終閲覧）
高度情報通信ネットワーク社会推進戦略本部（2013）「世界最先端IT国家総合宣言（初版）」
<http://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20130614/siryou1.pdf>（2016.12.1最終閲覧）
松林弘治（2015）「子どもを億万長者にしたければプログラミングの基礎を教えなさい」KADOKAWA
丸山幸三（2014）「小学校教育課程におけるプログラミング教育の考察：プログラミング教育に最適なプログラミング言語」『近畿大学豊岡短期大学論集』11, 11−19
日本経済再生本部（2013）「日本再興戦略− Japan is Back −」
<http://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/saikou_jpn.pdf>（2016.12.25最終閲覧）
Resnick, M.（1996）Distributed Constructionism. Proceedings of the 1996 international conference on Learning sciences, 280-284
Resnick, M.（2013）Learn to Code, Code to Learn. EdSurge News
<https://www.edsurge.com/news/2013-05-08-learn-to-code-code-to-learn>（2016.11.10最終閲覧）
Resnick, M.（2015）A Different Approach to Coding. International Journal of People-Oriented Programming, 4（1），1−4
産業競争力会議（2016）第26回産業競争力会議　文部科学大臣提出資料
<http://www.kantei.go.jp/jp/singi/keizaisaisei/skkkaigi/dai26/siryou2.pdf>（2016.12.25最終閲覧）
総務省（2015）「プログラミング人材育成の在り方に関する調査研究　報告書」
<http://www.soumu.go.jp/main_content/000361430.pdf>（2016.12.25最終閲覧）
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（2016）「小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）」
Wing, J. M.（2010）Computational Thinking: What and Why?
<http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf>（2016.8.18最終閲覧）
山本利一，本郷健ほか（2016）「初等中等教育におけるプログラミング教育の教育的意義の考察」『教育情報研究』，32（2）

（注）

ⅰ）… 解決の方法が直ぐには分からない問題状況を理解し、問題解決のために、認知的プロセスに関わ

ろうとする個人の能力。そこには建設的で思慮深い一市民として、個人の可能性を実現するために、自ら進んで問題状況に関わろうとする意思も含まれる。