幼児期からの論理的思考の発達

～プログラミング的思考に着目して～

1190436 織田佳乃子
高知工科大学経済・マネジメント学群

1． 概要

本研究は、人の論理的思考の発達上重要な転換期にあたる5歳～7歳までのプログラミング的思考の育成において、どのような教育内容・活動が適当かを考察するものである。論理的思考の発達様相に関する先行研究や米国のプログラミング教育の具体的なカリキュラムを参考に、5歳～7歳という幼児期のプログラミング的思考の発達のための教育活動に必要な観点として、①仮説形成を行うこと②協同的に学ぶことの2つを設定した。この2つの観点をもとに3歳から遊ぶことができるキュベットを用いた物語を創造する活動が、プログラミング教育として適当な教育活動の一つとして挙げることができた。

2. 序論

2020年全面実施となる小学校学習指導要領の改訂ポイントの一つとして、学習の基盤となる資質・能力に情報活用能力を言語能力と同様に位置づけられている[1]点が挙げることができる。小学校学習指導要領（2017）によれば、情報活用能力を育成する上では、コンピュータ等の活用やプログラミングの体験を各教科で実施することが求められている。プログラミングを体験する中で、児童がコンピュータに意図した処理を行わせるために論理的に思考するプログラミング的思考を身に付けることが期待されている[2]。小学校学習指導要領のプログラミング教育に関する記述には、論理的思考やプログラミング的思考という単語が頻出しており、これらが重要視されていることの証左と言える。

内田・津金(2014)によれば、人の論理的思考（ここで論理的思考とは、常に科学的根拠に基づく思考をさすのではなく、「〔幼児が：引用者注〕自分なりの身体感覚で捉え、生活体験と結びつけて因果的な認識をするという生活概念レベルの論理的な思考」[3]をさしている。）は、遊びや生活体験での試行錯誤や人とのコミュニケーションを通じて3歳から発達し始める。そして語彙力が増えることによって、科学的な真実に近い仮説を立て検証することができるようになるなど、5歳で論理的思考が質的に高まるとされている[3]。つまり5歳児であってもプログラミング的思考を育成することは可能であるといえるだろう。

またプログラミング的思考（英語ではComputational Thinkingが概念として近い。以下CT）育成のカリキュラムの先行事例として、本研究では米国の、特にCSTA（Computer Science Teacher Association）のK-12 Computer Science Standards(以下CSS)を取り上げることとする。中條(2015)によれば、CSSの前身である「K－12段階のコンピュータ科学（CS）のモデルカリキュラム」（“A Model Curriculum for K-12 Computer Science”）は、「世界各国の情報科教育関係者などから注目を浴びたこともあり，国際的普及を視野に入れた前書きを加えた改訂版が2006年に刊行された。」[7]とあり、CSSは世界の情報教育カリキュラムを牽引する存在であると判断したためである。

3. 5歳児のプログラミング的思考育成に必要な要素

3.1 プログラミング的思考と論理的思考をつなぐ仮説形成

プログラミング教育や論理的思考、そして論理的思考の一つの要素であるプログラミング的思考は、2006年に発表されたJeannette M. WingのComputational Thinking（邦訳：計算論的思考）に関するエッセイに基づいている。Wingによれば、「計算論的思考は、コンピュータ科学者だけではなく、すべての人にとって基本的な技術である」[4]とし、「ヒューリスティックな推論により解を発見することである」[4]とも定義している。つまり計算論的思考は、特に仮説形成からもたらされる論理的思考といえる。

文部科学省は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)において、プログラミング的思考は、計算論的思考を踏まえて「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」[5]と定義している。組み合わせを考え修正を加え改善していく過程には、子供自身が仮説を立てて検証する過程が含まれており、プログラミング的思考にも計算論的思考を踏襲し論理的思考における仮説形成を重要視していることがわかる。

内田・津金(2014)によれば、生後4か月ごろに因果知覚が始まり、6カ月で随意運動が発達し、さらに10カ月頃にイメージが誕生することで、例えばガラガラで遊ぶときに、子どもは手をどのように動かしたら、どんな音が出るかについて予測し、仮説を立て、実際に随意的に手の動かし方を変えて検証するようになる[3]とあるため、人の論理的思考の発達段階において仮説形成は、おおよそ最初に起こる論理的思考であると言うことができ、したがって5歳～7歳の子供でもできることである。

よって一つ目の観点としては仮説形成であり、子どもなりの仮説を立てたうえで検証させ試行錯誤させる体験が必要であると考える。

3.2 協同性

内田・津金(2014)によれば、1歳7カ月頃から始まる語彙爆発によって子どもが多言文を話すようになると、言葉は認知や感情と結びつき、知識獲得の手段となり知識の階層構造化が始まる。3歳には母語文法が獲得され、時系列因果関係や理由付ける表現ができるようになる。そして平均的な知能の5歳児は1日に20語の語彙を獲得するようになり、論理的思考が質的に高まるとしている。このような発達は、子どもが周りの人と話すことや会話を聞く中で起こり、幼児期の論理的思考は保育者や仲間との社会的やり取りの中で磨かれなければならない[3]と指摘している。そしてそれは協同的に学ぶ必要を示唆している。

藤谷(2017)は、内田・津金(2014)の幼児の論理的思考の分類規準を参考にしながら5歳児のエピソード記録の分析から幼児期の協同性の発達における論理的思考を調査した。そのうち5歳児の協同性と論理的思考力の関係性では、以下の図1からもわかるように、まず規則性に関する論理的思考は多くの協同的な遊びでみられた。後述するが5歳児のコンピュータサイエンスは、パターン、つまり規則性を発見することから始まるため、本研究と整合性がある結果となっている。また「伝えあいながら遊ぶ」においてアイデア・仮説が創出され、「相談しながら遊ぶ」ことでアイデア・仮説を創出する際に他者視点を入れていることが分かった。[6]

Table 1. 5歳児の協同性と論理的思考力

論理的思考＼協同性	2.ともに過ごす心地よさ	3.繰り返し同じ遊び	5.友だちの遊びに参加	6.イメージ・発見を伝える	7.楽しさに共感	8.伝え合いながら遊ぶ	9.折り合いをつけながら遊ぶ	10.相談しながら遊ぶ	11.達成感を共に味わう
(1)規則性	3	1	10	4	11	5	1	3	
(2)比較・分類				3	2	2	1		
(3)全体部分			1		2			1	
(4)因果関係					2	4	1	7	2
(5)アイデア・仮説		1		4	1	14	1	15	1
(6)他者視点				2	5	4	3	11	3

図1　5歳児の協同性と論理的思考力（出典：藤田(2017):37）[6]

よって二つ目の観点は、協同的に学ぶことである。以上2点の観点を踏まえ、さらに米国の、特にCSTAのCSSの先行事例を紹介するなかで、実際に5歳～7歳のこどもが行うべきプログラミング体験とはどのようなものであるかを考察する。

4. 先行事例

4.1 CSTAとCSSの概要

初めに4節で引用する英語の文献の訳は引用者によるものである。2011年版CSSまでの概要は、中條(2015)の解説があるため引用しつつ解説する。

米国は我が国のように全国共通の学習指導要領にあたるものはなく、かつ『「情報科教育」はこれまでほとんどの州で主要（コア）科目とはみなされていなかったので，区や学校による実施状況の差異・格差が大きいことが問題視され，アメリカを代表するコンピュータ関連の学会であるACM（Association for Computing Machinery）は「K―12カリキュラム特別委員会」を設置し，２００３年に最終報告書として「K―12段階のコンピュータ科学（CS）のモデルカリキュラム」（“A Model Curriculum for K―12 Computer Science”）を刊行した。このカリキュラムはアメリカのみならず世界各国の情報科教育関係者などから注目を浴びたこともあり，国際的普及を視野に入れた前書きを加えた改訂版が２００６年に刊行された。２０１１年には（中略）「K―12学 年を通したCSスタンダード」（CSTA K―12Computer Science Standards）が刊行』[7]され、2016年に改訂され、翌2017年にも一部改訂されている。

「CSS のフレームワーク運営員会は、ACM、CSTA、Cyber Innovation Center, National Math+Science Initiative の代表者からなっており、フレームワークの執筆部門は、この活動に参加している州・学区、教育者、高等教育施設、非営利団体の代表者からなっている。」[8]またこれらのメンバーからフレームワークに対するフィードバックを受け付けることで逐次フレームワークに改訂を入れている。[8]

4.2 2016 年改訂版 CSS の特徴

2011 年版 CSS では幼稚園から第 12 学年を 3 つのレベルに分けており、レベル別に学習内容が決められていた。

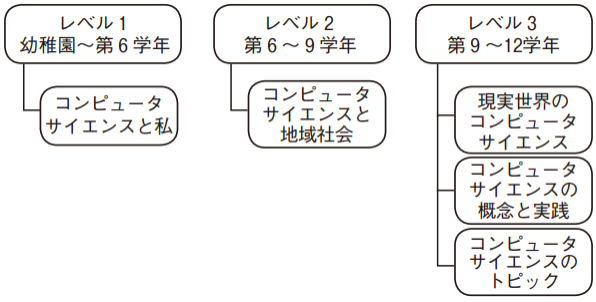

図 2 コンピュータサイエンス規準のための組織構造（出典：大森ら（2016）:275）[9]

コア・コンセプト	コア・プラクティス
1. コンピューティングシステム 2. ネットワークとインターネット 3. データ分析 4. アルゴリズムとプログラミング 5. コンピューティングの影響	1. 包括的なコンピューティング文化の促進 2. コンピューティング関連におけるコラボレーション 3. コンピューティング問題の認識と定義 4. 抽象の使用と発展 5. コンピューティング作品を制作 6. コンピューティング作品を検証して洗練させる 7. コンピューティングに関するコミュニケーション

図 3 The K-12 Computer Science Framework（出典:CSS Fiure0.1 を参考に作成）[8]

一方 2016 年改訂版 CSS では、すべての子どもに CT 学習を提供するためのベースラインとなることを目的としている。そのため 2016 年改訂版 CSS は、図 3 のように 5 つのコア・コンセプトと、7 つのコア・プラクティスを明示している。

CSS(2016)によれば、「コア・コンセプトとは、コンピュータサイエンス分野において主要な内容を表したもので、コア・プラクティスは、コア・コンセプトを最大限に活用できるコンピューティング能力がある生徒の行動を表す。」[8]としている。そして、CSS を利用したいと考えている主体（州、学校区、学校、教員）が、それぞれの CT 学習の状況に合わせて、CSS 内のコア・コンセプト（図 4 では KNOW にあたる）とコア・プラクティス（図 4 では DO にあたる）を組み合わせて独自の基準やカリキュラム開発を行うことを推奨しており、[8]2011 年版 CSS よりも柔軟性の高いフレームワークを実現している。

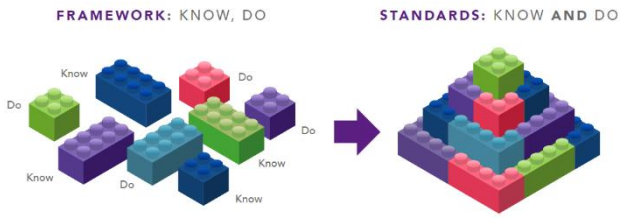

図 4 Building blocks for standards(出典：CSS Figure1.1)[8]

しかし 2017 年の一部改訂では、図 5 のようにコア・コンセプトに対応するコア・プラクティスを明記し、新たにスタンダード（図の右側）を加えている。本研究における CSS の事例研究はより実践的で具体的なプログラミング教育の先進事例を扱うため、5 歳～7 歳のコア・コンセプトとコア・プラクティスを中心に解説し、スタンダードについては取り扱わない。

4.3 5 歳～7 歳でできること

初めに CSS 内で多用されるコンピュータサイエンス（Computer Science）について触れておきたい。CSS は、先述の『「K ―１２段階のコンピュータ科学（CS）のモデルカリキュラム」（“A Model Curriculum for K―１２ Computer Science”）におけるコンピュータサイエンスの定義をもとに 5 つのコア・コンセプトを作った』[8]としている。そこでのコンピュータサイエンスの定義は、「コンピュータとアルゴリズム的処理、及びそれらの本質やハードウェア・ソフトウェアデザイン、アプリケーション、社会への影響に関する科目である」[10]とし、我が国のプログラミング的思考の概念よりも広

義であることを留意したい。

CSTA のサイト内にある Progressions Chart は、年齢とコア・コンセプト、コア・プラクティスの関係を簡潔に表にしており、このうち LEVEL1A(5 歳～7 歳)の部分を抜粋し筆者が日本語に訳したものが図 5 である。LEVEL1A のコア・コンセプトとコア・プラクティスの内容から、5 歳～7 歳の子どもができるプログラミング教育について紹介する。

まずコア・コンセプトに関しては図 5 のように、コア・コンセプトからさらに 2～5 つのサブコンセプトに分かれている。図にある概要を見ると、一見 5 歳～7 歳の子供には難易度の高いように思われる内容も含まれているが、CSS 内の 6 章でコア・コンセプトの詳細記述から各レベルに応じた学習内容を提案されていることが分かった。1. コンピューティングシステムのうち 2 項目目のハードウェアとソフトウェアを例にすると、全 Grade を通じたハードウェアとソフトウェアの概要は、「コンピューティングシステムは、デジタル方式で情報処理を行うことやコミュニケーションを行うために、ハードウェアとソフトウェアを用いています。低学年では、情報処理や情報表現のためにシステムがどのようにハードウェアとソフトウェアを運用しているかについて学びます。生徒が成長すれば、コンピューティングシステム内の多様な段階におけるハードウェアとソフトウェアの間の相互関係に関して深い理解を得られます。」[8]とあるが、Grade2 のハードウェアの項目では、「(コンセプトの意図として) コンピューティングシステムは、ハードウェアとソフトウェアでできています。ハードウェアは物理的な構成要素からなるのに対して、ソフトウェアはシステムに対して命令を与えるものです。これらの命令はコンピュータが理解できる形式で表されます。(詳細と具体例として) ハードウェアの例としては、情報を表示するスクリーンや、情報を得るためのボタンやキー、ダイヤルがあります。ソフトウェアは、ウェブブラウザやゲームのような特定の目的を持つプログラムのアプリをさします。ある人が、新しいウェブページに進むために、ウェブブラウザ (ソフトウェア) 内に表示されているボタンをマウス (ハードウェア) でクリックするとします。するとコンピューティングシステムは、「印刷」「保存」「トリミング」といった命令をコンピュータが理解できる特別な言語に変えます。このレベルでは、コンピュータシステムがデータを符号に置き換えている (コードされている) ということは理解できるが、「バイト」といった単語を用いた正確な理解はできず、次のレベルに持ち越しになります。」[8]とあり、ハードウェアやソフトウェアの例は子供にとって身近なものをさしていたり、複雑な専門用語を使わないようにしていたりするなどの年齢にあった学習内容に工夫されている。

次にコア・プラクティスについて紹介する。CSS(2016)によれば、「コア・プラクティスのうち 3～6 は計算論的思考に関連した実践で、1, 2, 7 はコンピュータサイエンスの一般的な実践例として独立したものである」[8]としている。図 5 の概要文の後ろにあるコア・プラクティス番号とプラクティス詳細番号に注目すると、5 歳から 7 歳までの子供の活動には、4.Developing and Using Abstractions (抽象概念の利用とその発達) と 7.Communication About Computing (コンピューティングに関するコミュニケーション) の実践方法を取り入れるコア・コンセプトが際立って多いことがわかる。4.Developing and Using Abstractions (抽象概念の利用とその発達) の概要には、「抽象化は一般概念を作り出すために、パターンを特定することや、特定の例から共通の特徴を抽出することである。抽象化を行って全体または一部のプログラムを組むことができたらより複雑なプログラムを単純にする方略が分かっていることとなり、幅広い分野で使えます。」[8]とある。特に幼い子供には、物理現象や図や写真といったイメージから特定のパターンや流れを認識させ、かつその仕組みを理解し、絵にかいたりするなどして説明させる体験をさせるべきだとしている[8]。

7.Communication About Computing (コンピューティングに関するコミュニケーション) の概要には、「コミュニケーションは、個人の表現と、他人との意見の交換があります。コンピュータサイエンスでは、生徒は、多様な聴衆に対して、使用方法やコンピューティングの効率化、適切なコンピューティングを選択することについてコミュニケーションを取らなければならない。生徒は、明瞭な文書を書き、課題を整理し、多様なメディアを通じてコミュニケーションを行います。またわかりやすいコミュニケーションには、最適な言葉を使い、十分に配慮された聞きかたをすることが含まれています。」[8]とある。そして早い段階から、絵コンテ (紙芝居) やフローチャートやグラフに表したり、自分たちが行っていることに関して専門用語を用いたり、明瞭な言葉で発言できるようにするべきだと

している[8]。

ここではCSSの5歳～7歳のコア・コンセプトとコア・プラクティスについてのみ扱ったが、5歳～7歳の子供でもコンピュータサイエンスにおいて理解できることが数多くあることが分かった。CSS（2016）の記述にもあるように5歳～7歳では、複雑な概念やプログラムについては理解できない。しかしハードウェアとソフトウェアの関係性など基本的知識や、グラフや図を描いて物事を説明するといったことは、5歳～7歳の子供でも理解し実行することができ、かつ仮説検証や言語活動を含んだ活動になるために、プログラミング的思考を育成するためのプログラミング教育として行うべき活動に含むべきではないだろうか。

またCSSの特に5歳～7歳のカリキュラム内容を見てみると、仮説形成はまず規則性を見つけることから始まっていることがわかる。そして他人とコミュニケーションをとることや協同的に学ぶことはコア・プラクティスの大半の実践において前提とされていた。一方で他人にわかりやすく説明できるようにする能力もまた大半のコア・プラクティスで繰り返し言われており、重要視されていることがうかがえた。CSS(2016)内の解説には、コア・プラクティス7のような言語活動を多く含むものは、地域によって子どもの英語能力の差が大きいアメリカ独特の背景に考慮し、英語能力の程度に関わらずコンピュータサイエンスに触れることができるように制度設計されたものである[8]という記述もあったが、それを差し引いても、我が国のプログラミング教育にはまだ少ない要素である。

CSS（2016）の先行事例研究では、プログラミング体験内で、重視すべき活動内容の示唆が得られた。

コア・コンセプト	サブコンセプト	LEVEL 1A（5歳～7歳） Grade2の終わりまでに児童たちは以下のことができるようになっているだろう
コンピューティングシステム	デバイス	様々な課題に適切なソフトウェアを選び、操作を行い、ユーザーによって科学技術に対する期待や需要が異なることを認識する（P1.1）
	ハードウェアとソフトウェア	専門用語を用いて、コンピューティングシステム（ハードウェア）に共通した物理的構成要素の機能を特定し、説明する（P7.2）
	トラブルシューティング	正確な専門用語を用いて基本的なハードウェアとソフトウェアの問題を説明する(P6.2,P7.2)
ネットワークとインターネット	ネットワークコミュニケーションとネットワーク組織	無し
	サイバーセキュリティ	どのようなパスワードがあって、なぜ使われているのかを説明し、不正アクセスから情報やデバイスを守るために強度の高いパスワードを使う(P7.3)
データ分析	ストレージ	コンピュータデバイスを用いて情報の保存・コピー・検索・取り込み・修正・削除を行い、データとして格納された情報として定義する(P4.2)
	収集・可視化・転換	様々な可視化フォーマットで同じデータを収集し表示する(P7.1,P4.4)
	推測とモデル化	予測を立てるために、チャートやグラフのような可視データ内のパターンを認識し説明する（P4.1）
アルゴリズムとプログラミング	アルゴリズム	課題を達成するために、アルゴリズムに従うことや、アルゴリズムを作り出すことで日々の成果をモデル化する（step-by-stepの要領で教える）（P4.4）
	変数	情報を表すために、数値やほかの記号を用いて、プログラムがデータを格納及び操作をする方法をモデル化する(P4.4)

	制御	考えを表現し、問題に取り組むために、シークエンスや単純なループを使ってプログラムを発展させる(P5.2)
	モジュール方式	問題を解決するために必要なステップを正確な命令シークエンスに分解する(P3.2)
	プログラム開発	プログラムの一連のイベントや目標、そして予想される結果を記述する計画を立てる(P5.1,P7.2)
		プログラム開発中にアイデアやほかの創作物を使うときに要因分析を行う(P7.3)
		シークエンスや単純なループを含むアルゴリズムやプログラミングのデバックを行う(P6.2)
		プログラム開発の間、正確な専門用語を用いて目的に沿ったプログラムを組み、プログラム開発にはいくつかステップがあることが分かり、選ぶことができる(P7.2)
コンピューティングの影響	文化	人間がコンピューティング技術を受け入れる前と後で生活がどのように変わったか比較する(P7.0)
	社会的交流	他者と礼儀正しく責任感をもって働く(P2.1)
	安全性、法律、倫理	適切に非公開でログインし、デバイスからログオフする(P7.3)

図5　5歳～7歳でできるコンピューティング(出典：CSTA　Progression Chart　https://www.csteachers.org/page/standards を参考に作成)

表内の（P1.1）といった表記は、(P[コア・プラクティス番号].[プラクティス詳細番号])

5.　考察・提案

3節で5歳～7歳児のプログラミング的思考発達のための教育活動に必要な観点として、①仮説形成を行うこと②協同的に学ぶことの2点が明らかになった。そして4節において、仮説形成はパターンを認識させる活動から始めることや、協同的に学ぶ中で他者にわかりやすく説明するといった言語活動を取り入れることの重要性も踏まえ、Candlewick社のキュベットを用いた活動を考察する。

PRIMOキュベットのホームページ[11]をもとにキュベットの概要を紹介する。キュベットはコーディングを学ぶ教材であるが、コーディングを行うボードや、コーディングをもとに動くロボットであるキュベットはともに木製であり、パソコン画面で操作することや複雑なプログラミング言語を用いる必要がなく、幼い子供から使うことができる。そのおかげもあってか100ヶ国以上で2万人を超える教育関係者や保護者が使用しており、幼稚園から小学校、プレスクール、自宅学習、特別支援学級、学童保育、公立図書館、コミュニティセンターなどの現場で活用されている。同ホームページにある日本での活用例には老人ホームでも使われており、幼い子供からお年寄りまで学べる教材であることがわかる。

基本的な遊び方は、4種類の異なるコマンドをもつブロック（緑：キュベットは前に進む　黄：キュベットは左に90度向く　赤：キュベットは右に90度向く　青：ファンクションラインに並べたブロックの通り動く）をボードに埋め込むことでキュベットの動き方をコーディングし、別売のワールドマップ上で動かす。ボードにおけるブロックは12個までであるため、例えばキュベットを目的地に移動させるのに12個以上のコマンドがいる場合は、ボード上の別枠にあるファンクションラインで最大で4つまでのコマンドでキューを作り、適当な箇所に青のブロックを入れる必要がある。ワールドマップは様々な種類がある上に、マップ上に障害物やアイテムを置くことで遊び方は自在に変えられる。またワールドマップ自体を自ら作ることも可能である。[11]

図6　キュベット　(出典：PRIMO https://www.primotoys.jp/) [11]

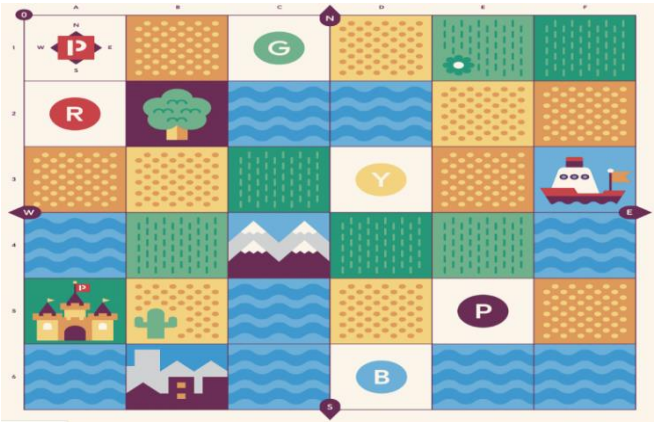

図7 ワールドマップ（出典：PRIMO https://www.primotoys.jp/）[11]

ここでプログラミング教育普及プロジェクトComputer Science for Allで紹介されている山形大学の加納寛子氏らによるキュベットを用いた授業をモデルケースとして提案する。以下はプログラミング教育普及プロジェクトComputer Science for Allのサイト[12]や日本教育工学会のホームページ[13][14]をもとに説明する。加納寛子氏らは日本教育工学会において、SIG（現代的教育課題に対するSIG（Special Interest Group）活動[13]という、学会内の特定の重要なテーマに対して研究者がグループになり研究を進めるという活動を行っている[13]。加納氏らのSIG11では、『新しい時代を生きる子どもたちが学ぶ必要のある情報リテラシーの主な柱には、「情報通信技術」「情報システム」「問題解決」「情報分析」「情報モラル」「情報の歴史」「情報機器の操作」等がある』[14]とし、「それらを系統的に、小学校1年生から高校3年生さらには大学生まで、情報リテラシー教育の体系的カリキュラムや到達目標を検討していくこと」[14]を設立趣旨としており、キュベットを用いた授業は、初等教育における情報教育の学習方法の開発として研究されていると推測した。

授業の対象は2歳児～小学5年生（小学校2年生～4年生中心）の32人で、キュベットを使って「プログラミングの考え方と想像力、創造力を育むこと」[12]を狙いとしている。まず授業の導入30分間でプログラミングやプログラミング的思考がなぜ必要なのかを講義している。ここでキュベットが単なる遊びとならないように、「（中略）どのような職に携わる人にも、プログラミング的な思考、つまり自分でソースコードを書く必要はないが、目的に合わせたアルゴリズムを自分で組み立てる思考が必要となります。そういった思考は小さいときに学んでおけば、身体的な感覚として身につけることができます。」[12]という内容を話したことを指導・ファシリテーション上の留意点・工夫として挙げている。次に展開前半でキュベットの動かし方を説明しており、目的に照らし、どのようにプログラムを組めばよいかを考えることができるようにしている。子供たちが実際にマップ上でキュベットを動かす中で、ループを使うことを見つけさせたとあるため、ファンクションのブロックを使って難易度を高くしていることがわかる。後半ではリンゴ太郎の物語を見せ、各自物語を考えさせ、物語に即して動くようにキュベットをプログラミングさせる活動を行っている。ここで授業全体のねらいでもあるように、創造性を育ませることを目的としている。そして最後に授業のまとめとして自分の物語を発表し、他の人の物語を聞くことでいろいろな物語ができることを知る。[12]

まず①仮説形成を行うことに関しては、特に授業の展開の前半において、キュベットの動かし方に関する基本的な説明をした後は、例題を使って子供たちが実際にプログラミングを行う中でループを使うことまで自ら発見できるように時間を取っている。キュベットを始めて使う子供にとっては、ブロックをボードにはめてキュベットを動かすこと自体仮説と検証を行っていることになり、ループを使うことに気づくためには、目的地までのキュベットの道筋にパターンを見つけ、何通りもコマンドを並び替えて適切なコーディングを見つけなければならない。CSSのコア・コンセプト4アルゴリズムとプログラミングの制御の詳細説明においても、5歳～7歳の子供は単純なループは使えるとの記載もあり、よって①の観点は満たしているといえる。

次に②協同的に学ぶという観点においては、授業の展開の前半からまとめまで3人ずつグループになっており、特に展開の前半では3人ずつグループになってお互いが意見を出し合いながらキュベットの使い方を学んでいたようであった。また授業の最後にまとめとして、自分の作った物語を一人ずつ発表させているが、加納氏らの授業のねらいにあるように、「自分以外の人の物語も聞くことにより、いろいろな物語ができることを知る」[12]ということも、協同的に学ぶことによってできる学習であるといえる。この狙いに加えて、他人に分かりやすい言葉で説明を行うことができるということも4節のCSSの事例研究よりプログラミング教育上重要な観点であるといえる。よって②の観点も満たせているといえる。

キュベットを用いた加納寛子氏らの活動は以上のように本研究で

設定した２つの観点を含むものであり、５歳～7 歳児のプログラミング教育のモデルケースとして有意義なものであると考える。

6.　今後の課題

本研究では、５歳児～7 歳児のプログラミング的思考の発達に重要な教育カリキュラムや教材を紹介してきた。５歳～7 歳までの子供のプログラミング的思考は、プログラミング体験そのものだけから発達するものではなく、プログラミング的思考に必要な能力を養うほかの活動からも発達することが分かった。プログラミング的思考という言葉から、教員には高度なプログラミング知識が必要であるかのように感じられるが、決してそうではなく思考力を養うために効果的な活動を重視することが求められている。プログラミング教育の意義の周知が不十分な現状を如何に打開するかは今後の課題であろう。

また新井紀子氏は、基本的読解力を測定するリーディングスキルテストにおいて、「2017 年 11 月末までに約３万１千人を対象として本テストを実施した結果、中学３年生のうち、推論では 54.4%、同義文判定では 42.8%、理数系の定義を理解できるかを問う具体例同定問題では、実に 74.0%の生徒がランダム並み、言い換えると「ほとんどできていない可能性が高い」こと」[15]が分かったとしており、プログラミングを教える前に読解力の育成が急務であると主張している[16]。プログラミング的思考を含む論理的思考は、国語能力や、数理能力、推論力等が複雑に関与していることは、それぞれの分野の研究で明らかになっているが、論理的思考の発達に必要な能力を統合的に研究しているものは少ない。新井氏の指摘は、論理的思考を鍛える前に、各教科において読解力を十分育成する必要性を示唆しており、その方略についてもプログラミング教育をより効果的に実施する上で今後の課題といえるだろう。

引用文献

[1] 文部科学省「新学習指導要領（小学校及び中学校：平成２９年３月告示）～情報教育・ICT 活用関連部分のポイント」
〈http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/03/30/1375607_01.pdf〉
2019 年３月６日アクセス

[2]文部科学省(2017) 小学校学習指導要領

[3]研究代表者:内田伸子, 津金美智子 (2014)「乳幼児の論理的思考の発達に関する研究 ―自発的活動としての遊びを通して論理的思考力が育まれる―」日本保育協会, 5, pp131-139

[4] Jeannette M. Wing (2006) 「Computational Thinking」中島秀之訳, 『計算論的思考』「情報処理」56 (6), pp584-587

[5]文部科学省初等中等教育局教育課程課教育課程企画室(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)」,
〈http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm〉 2019 年 1 月 10 日アクセス

[6]藤谷智子(2017)「幼児期の協同性の発達における論理的思考力―5 歳児の発達過程に着目して―」武庫川女子大学紀要. 人文・社会科学編, 64, pp31-39

[7] 中條道雄(2015)「アメリカにおける情報科教育カリキュラム標準化の動向－ACM/CSTA の「K-12 の computing Standard」を中心に―」実教出版, 情報教育資料 40 号, pp5-8

[8] K-12 Computer Science Framework. (2016). Retrieved from http://www.k12cs.org.

[9] 大森康正・磯部征尊・山崎貞登 (2016)「STEM 教育 Computational Thinking 重視の小・中・高等学校を一貫した情報技術教育の基準に関する日イングランド米比較研究」上越教育大学研究紀要, 35, pp269-283

[10] Allen Tucker, Fadi Deek, Jill Jones, Dennis McCowan, Chris Stephenson, Anita Verno(2003)「A Model Curriculum for K-12 Computer Science:Final Report of the ACM K-12 Task Force Curriculum Committee」

[11]PRIMO プリモトイズ日本販売総代理店
〈https://www.primotoys.jp/〉 2019 年 1 月 25 日アクセス

[12] Computer Science for All プログラミング教育普及プロジェクト　授業・ワークショップ紹介「自分で物語を創作 - プログラミングで創造力と想像力を育む」
(http://csforall.jp/case/2983/) 2019 年 1 月 27 日アクセス

[13]日本教育工学会　SIG 活動
(https://www.jset.gr.jp/sig/index.html)
2019 年 2 月 3 日アクセス

[14]日本教育工学会　SIG-11　情報教育
(https://www.jset.gr.jp/sig/sig11.html)
2019年2月3日アクセス
[15]RST 一般社団法人教育のための科学研究所
(https://www.s4e.jp/about-s4e)2019年2月8日アクセス
[16]ITmedia Inc.(2018) 特集：小学生の「プログラミング教育」その前に (8)：「東ロボ」を主導した数学者が「読解力がない子どもにプログラミングを教えても、意味がない」と主張する理由
(http://www.atmarkit.co.jp/ait/articles/1804/26/news018.html)
2019年2月8日 アクセス