Computational Thinkingを育成する学習支援システムの開発に関する研究

(A Study of the Development of Learning Support System for the Promotion of Computational Thinking)

学位取得年月　2021年3月

D183489

福井　昌則

概要

高度情報化が加速する 21 世紀のデジタル時代において，Computational Thinking を身につけることが重要であると指摘されている．Computational Thinking とは，問題をコンピュータで解決できる形で整理し，表現するための思考プロセスのことであり，各国の Computer Science 教育などにおいて育成が求められている重要な概念である．我が国のプログラミング教育では，Computational Thinking 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「プログラミング的思考」の育成が掲げられており，その思考をプログラミングの授業のみならず，コーディングを必須としない形で，各教科内で育成することが求められている．本研究では，Computational Thinking をプログラミング教育および各教科の内容で育成することを目指している．その実現に向け，以下の 2 つの研究を実施した．

1 つ目の研究は，創造性とプログラミングに対する意識との関連性の検討である．プログラミング教育において，Computational Thinking をどのように育成するかについて様々な研究が行われており，プログラミングのシステムも多数存在している．プログラミングは自由度が高く，自身のアイディアを具現化することが可能であり，創造性を育成することが期待できる．Computational Thinking と創造性には関連性があること，プログラミングによって Computational Thinking の育成が可能であることが指摘されている．創造性育成の重要性が指摘される中，創造性を身につけることで Computational Thinking を高めることができれば，今後のプログラミング教育の充実化も期待できる．しかし，我が国のプログラミング教育において育成が期待されている，プログラミングに対する様々な意識と創造性の関連性については明らかにされていない．よって本研究では，プログラミング教育充実化に向けて，我が国のプログラミング教育で育成が期待されている，プログラミングに対する様々な意識と創造性との関連性について検討する．

2 つ目の研究は，各教科の内容で Computational Thinking を育成する学習支援システムの開発である．各教科で Computational Thinking を育成するために，分類課題に着目する．網羅的に分類を行うことは，既存の学習内容を抽象化した上で手続き的に表現することが必要となる．その表現において Yes/No チャートを用いることで, 各教科内で Computational Thinking を育成することが期待できる. しかし, 分類課題を用いた Computational Thinking 育成を行うシステムについては十分に行われていない．また，それまでのプログラミング経験が Yes/No チャートを用いた Computational Thinking 育成に対してどのような影響を与えるかについては不明である．これらを解決するために，各教科の内容での Computational Thinking 育成を指向した題材開発，および学習支援システムを開発・評価を行った．

得られた結果の概要を示す．1 つ目の研究では，高校生の創造的態度とプログラミングに対する様々な意識との関連性についての調査を行い，創造的態度とプログラミングに対する様々な意識が関連性を有すること，性別によって関連性を持つ創造的態度因子とプログラミングに対する様々な意識の項目に相違があることが把握された．また，創造的態度の柔軟性が 2 つに分かれ，柔軟性 2 因子とプログラミングに対する様々な意識の間に関連性があることが示された．

2 つ目の研究では，各教科で Computational Thinking を育成するために，図形の分類課題を題材とし，正誤判定機能を実装したベン図，Yes/No チャート学習システムを開発しその学習効果について検討した．その結果，Yes/No チャート学習システムが Computational Thinking 育成に有効であることが示された．そして，本学習システムを用いた Computational Thinking 育成にそれまでのプログラミング経験が影響しないことも示された．

本論文の構成を示す．第 1 章では，本研究の位置付けと意義について述べる．第 2 章では，先行研究および各概念・定義について述べる. 第 3 章では, 創造性とプログラミングに対する意識との関連性について述べる. 第 4 章では, Computational Thinking を育成するための題材および学習支援システムの開発と評価について述べる．第 5 章では，本研究を総括し，今後の展望を述べる．

目次

図目次

表目次

1. 序論

今日の世界においては，加速する高度情報化社会や，先進国に顕著に見られる少子高齢化への対策が重要視されている．さらに，テクノロジーの進展に伴い，2045 年には人間の能力をコンピュータが超えるという，シンギュラリティ(技術的特異点)到来についての予測がされており[1][2]，従来の対策では解決困難な問題が増えてくることが想定される．そして我が国が目指すべき未来社会の姿として，狩猟社会(Society 1.0)，農耕社会(Society 2.0)，工業社会(Society 3.0)，情報社会(Society 4.0)に続く，新たな社会を指す Society 5.0(サイバー空間(仮想空間)とフィジカル空間(現実空間)を高度に融合させたシステムにより，経済発展と社会的課題の解決を両立する，人間中心の社会) が提唱された[3]．これからさらに進展する情報化社会においては，人工知能(AI)やテクノロジー，情報技術はその根幹をなす技術であり，それらの現状に対応できる力の育成が求められている．そのような力として，例えば 21 世紀以降に必要とされる 21 世紀型スキルなどが掲げられている[4]．

このような状況を踏まえて，普通教育の中でプログラミング教育を必修化すべきだという要請が産業界から見られるようになった[5]．政府は「産業競争力の源泉となるハイレベルな IT 人材の育成・確保」を目的とし，「IT を活用した 21 世紀型スキルの修得」と「人材のスキルレベルの明確化と活用」を行うことを掲げている[6]．また，人工知能等の活躍によって，仕事の内容や働き方は劇的に変化していくと考えられるが，データを利用して付加価値を生み出すのは「人材」であるとし，第 4 次産業革命を支える人材の確保・育成に向けて，2020 年から，義務教育段階においてプログラミング教育を必修化することを決定した[6]．高校における教育においても，これまで行われていたプログラミング教育の内容をさらに充実化することが決定し[7]，その内容がまとめ直されて公開されている．また小中学校の児童生徒に一人一台の端末が配備される GIGA スクール構想が進められるなど[8]，我が国の ICT をとりまく状況は目まぐるしく変化している．

一方，世界各国におけるプログラミング教育は様々な形で実施されているが，その中で高度情報化が加速する 21 世紀のデジタル時代を生き抜くために，Computational Thinking を身につけることが重要であると指摘されている．Computational Thinking は，問題をコンピュータで解決できる形で整理し，表現するための思考プロセス[9][10]のことであり，各国の Computer Science 教育などで育成が求められている重要な概念である．Tabesh は，Computational Thinking は 21 世紀型スキルであり，テクノロジーの発達した今日の世界においてますます重要となっていると指摘している[11]．このことから，Computational Thinking は，21 世紀を生き抜くために必要な力として不可欠である．また Wing は，Computational Thinking はすべての人にとって不可欠なスキルであると述べており[12]，学生の Computational Thinking を高める取り組みが不可欠であると考えられる．

我が国の教育に目を向けると，2020 年から完全実施されている小学校・中学校学習指導要領では，プログラミング教育が必修化された．そしてプログラミング教育の必修化につい

ての方向性については，文部科学省有識者会議[13]によって示された．文部科学省有識者会議[13]によれば，我が国のプログラミング教育では，プログラミング的思考(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，記号の組み合わ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)の育成，そしてコーディングを必須としないこと，各教科内でプログラミング的思考を育成することが掲げられている．プログラミング的思考とは，Computational Thinking 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概念であり[13], Computational Thinking の育成は，我が国のプログラミング教育においても重要である．以上のことから，「プログラミング教育において Computational Thinking を育成すること」，「各教科内でコーディングを用いずに Computational Thinking を育成すること」の 2 点を実現することが求められる．

1 点目の「プログラミング教育において Computational Thinking を育成すること」についての事例は多く見られ，Scratch などのシステムを用いた事例が見られる[14]．Wong and Cheung は，児童生徒がプログラミング活動を用いて Computational Thinking を身につけることができると述べており[15]，Computational Thinking を育成する上でプログラミングを活用することは有用である．プログラミングは，他の教科より自由度が高く，学習者が問題を自由に変更することや様々な実現・実装方法などが存在するため，学習者それぞれが自身で考えたことを実現するために有用な手法の一つである．プログラミング教育では創造性の育成が期待されており，プログラミング教育で創造性を育成するようなカリキュラムを策定することが重要である．Rotem et al., Doleck et al.は，Computational Thinking と創造性には関連性があると指摘している[16][17]．また，プログラミングで Computational Thinking の育成が可能であること，プログラミングで創造性を育成する事例など多く報告されている[18][19][20]. 創造性育成の重要性が指摘される中，創造性を身につけることで Computational Thinking を高めることができれば，今後のプログラミング教育の充実化も期待できる．しかし，我が国のプログラミング教育において育成が期待されている，プログラミングに対する様々な意識と創造性の関連性については明らかにされていない．よって，これらの詳細について把握することは，創造性と Computational Thinking を育成するプログラミング教育の具現化に向けて重要であると考えられる．本研究では，プログラミングに対する様々な意識と創造性の関連性について把握することを試みた．

一方，2 点目の「各教科内でコーディングを用いずに Computational Thinking を育成すること」については，アンプラグドコンピューティングなど，コーディングを伴わない紙面上で行う実践が多い．また，各教科内容とは別の課題を準備した上で Computational Thinking の育成を目指すことが多い．各教科で Computational Thinking を育成するためには，既存の教科内で扱われている題材を活用する方法を確立することが必要である．そのために，どの教科においても適用可能な Computational Thinking 育成教材として，分類課題

に着目する．「分類」は小学校においても教えられている．そして，論理的で網羅的な分類が論理的思考の基礎として重要である．しかし，分類に関する学習が適切には成立していない可能性があり，分類についての適切な思考経験は十分に得られていないことが想定される．つまり，分類は教えられてはいるものの，それについての適切な思考経験は与えられておらず，その結果として，分類のための論理的思考が十分に習得されていないことが想定される．本研究では，分類課題を Computational Thinking の課題として捉え直すために，ベン図，Yes/No チャートを活用し，図形を網羅的に分類させることを目指す．分類を Computational Thinking の観点から見れば，物事を分類するために，Computational Thinking の「パターン抽出 (一般化)」と「抽象化」が不可欠である．分類の実行には，Computational Thinking の「手続化」が必要となる．また，分類を段階的に行っていくことは，分類という課題に対して Computational Thinking の「分解」を行うことである．よって分類課題は，Computational Thinking の基本要素をすべて含む課題となっていると考えられ，各教科における分類課題を Computational Thinking 育成課題として捉えることで，各教科内で Computational Thinking を育成することが実現できる．その実現のために，ベン図，Yes/No チャートを用いたシステムを開発し，学習者それぞれの Computational Thinking 育成を目指す．

以上に基づき本研究では，21 世紀のデジタル時代に必要なスキルである Computational Thinking をプログラミング教育および各教科内で育成することを目指す. この実現に向けた第一歩として，「プログラミングと創造性の関連性の把握」，「各教科内での Computational Thinking 育成」という 2 点について研究を行った．

本研究では，以下の 2 点の知見が得られた．1 点目として，プログラミング教育において創造性を育成するために，レディネスとしての創造的態度に着目し，創造的態度とプログラミングに対する様々な意識との関連性についての調査研究を行い，男子の方が女子よりも創造的態度とプログラミングに対する様々な意識の各項目が高いこと，性別によって関連性がある創造的態度とプログラミングに対する様々な意識の項目に相違があることを示した．そして，性別によって創造的態度を高めるプログラミング教育の方法について差異がある可能性が示唆された．2 点目として，Computational Thinking を育成するために，各教科で用いられている題材の活用を前提とした上で，図形の分類課題を Computational Thinking 育成課題として設定した．その育成のために，ベン図と Yes/No チャートを導入し，自動判定機能を実装したベン図，Yes/No チャート学習システムを開発した．その上で，Yes/No チャート学習システムの学習効果を，経験学習を取り入れた実践モデルに基づいて測定した．その結果，Yes/No チャート学習システムが Computational Thinking 育成に有効であることを示した．また，システム経験前の Computational Thinking テストの得点によって学習効果が異なることも合わせて示した．そして，Yes/No チャート学習システムを用いた Computational Thinking 育成にそれまでのプログラミング経験が影響しないことも合わせて示した．

本論文の構成を示す．第 1 章では，本研究の位置付けと意義について述べる．第 2 章で

は，先行研究及び各概念について整理する．第 3 章では，創造的態度とプログラミングに対する意識との関連性について述べる．第 4 章では，Computational Thinking を育成するための題材を開発し，Computational Thinking を育成するベン図，Yes/No チャート学習システムについて述べる．そして，Yes/No チャート学習システムの学習効果，プログラミング経験が Yes/No チャート学習システムに与える影響について述べる．第 5 章では，これらの研究を総括する．

2. 関連研究と諸概念の整理

2.1. Computational Thinking

第 1 章において Computational Thinking の育成が重要であると述べたが，Computational Thinking という語を初めて利用したのは Papert である[21][22]．Denning は，Computational Thinking とは，これまでにも Algorithmic Thinking などで使われていた概念を今の時代に合うようにしたものであると指摘している[23]．Algorithmic Thinking と Computational Thinking の違いとしては，まずプログラミングなどを専門的に行う人以外にも計算機科学で用いる考え方を身につけることが重要であると指摘している点にある．そして Computational Thinking の概念の一つに Algorithmic Thinking が内包されている．プログラマやエンジニアを専門的に養成することが目的ではない一般教育においては，Computational Thinking の育成は重要な課題の一つである．

そして，現在 Computational Thinking についてよく知られるようになったのは，Wing が Computational Thinking について述べた部分が大きい[9]．Wing によれば，Computational Thinking とは「問題をコンピュータで解決できる形で整理し, 表現するための思考プロセス」と定義されている[9][10]．また Wing は，教育における Computational Thinking について，計算機で実行できる形で問題を解決する方法の集まりであるとし，Computational Thinking の構成概念や要素について紹介している[12]．BBC Bitesize は，Computational Thinking の概念として，論理，アルゴリズム，分解，パターン化，抽象化，評価の六つをあげている[24]．Google for Education は，Computational Thinking の概念として，心理的なプロセスとして抽象化，アルゴリズム設計，分解，パターンの認識，目に見える結果として，自動化，データ表現，パターンの一般化などがあるとし，それらはコンピュータを用いた問題解決と関係があると指摘している[25]．また ISTE and CSTA も Computational Thinking の操作的定義を示している[26].

このように Computational Thinking には様々な概念や定義が掲げられているが，文部科学省の資料において Wing の Computational Thinking について触れられていること[13]，そして他の研究においても Wing の定義がよく用いられていると報告されていることから[27]，本研究では，Wing の Computational Thinking の定義を用いる．

Computational Thinking は，コンピュータでも実行できる形で表現することであり，このことは，論理的に説明を記述すること，そして他者が見ることで実行・理解・再現可能な表現を行うことと捉えることができる．つまり，Computational Thinking を身に付けることは，単にコンピュータを用いた活動や作業の効率化に対して効果的であるということにとどまらず，思考を整理し，他者に伝えるといった力の育成にもつながる．よって，プログラミング的思考を育成するために，より広い概念としての Computational Thinking を育成すること，および各教科の中で実施できるような Computational Thinking 育成モデルを構築することが重要であると考えられる．

Computational Thinking を育成するという試みは多くなされている．アメリカではCS(Computer Science)教育の中でプログラミングが実施されている[28]．イギリスでは教科「Computing」において，Computational Thinking の育成に着目している[29]．そして Scratch を用いたプログラミングや，Computational Thinking を高めるような様々なカリキュラムが実施されており[30][31][32]，コーディングを行うことを取り入れた Computational Thinking 育成が行われている．我が国においても，プログラミング教育の事例が多く提示されており，その中でプログラミング的思考の育成が目指されている[32][33].

その一方で，実際にコーディングを行うような教科以外での Computational Thinking を育成する研究については，アンプラグドコンピューティングなど，コーディングを伴わない紙面上で行う実践が多い[34]．これらの題材は，各教科内容とは別の課題であることが大半であり，そのような題材を用いて Computational Thinking の育成を目指すことが多い．各教科で Computational Thinking を育成するためには，既存の教科内で扱われている題材を活用する方法を確立することで．各教科内で Computational Thinking を育成するということが可能となる．そのために，どの教科においても適用可能な Computational Thinking 育成教材として，分類課題に着目する．以下，分類課題について述べる．

2.2. 分類，Yes/No チャート

「分類」は小学校においても教えられており，論理的で網羅的な分類が論理的思考の基礎として重要である．しかし，分類に関する学習が適切には成立しておらず，分類についての適切な思考経験が得られていないことが想定される．つまり，分類は教えられてはいるものの，それについての適切な思考経験を与えられておらず，その結果として分類のための論理的な思考が習得されていないことが想定される．分類を Computational Thinking の観点から見れば，物事を分類するために，Computational Thinking の「パターン抽出 (一般化)」と「抽象化」が不可欠である．分類の実行には，Computational Thinking の「手続化」が必要となる．また，分類を段階的に行っていくことは，分類という課題に対して Computational Thinking の「分解」を行うことである．よって分類課題は，Computational Thinking の基本要素をすべて含む課題となっていると考えられ，各教科における分類課題を Computational Thinking 育成課題として捉えることで，各教科内で Computational Thinking を育成することが実現できる．

各教科内でプログラミング的思考や Computational Thinking を育成するために，平嶋，林らは，分類によって弁別を行うベン図，Yes/No チャートを用いたモデルを提案し，その実践について報告している[35][36]．平嶋らは，ベン図と Yes/No チャートを用いたプログラミング的思考課題を提案し，図形を題材とした例について述べている[36]．ベン図は学習対象の分類の宣言的な記述，Yes/No チャートはその手続きとしての記述を行うために用いている．またそれらの内容を発展させ，ロボホンを用いたインタラクティブなブロックプログラミング授業を提案している[37].

Yes/No チャートとは，分類を実行するために必要な手続きを段階的に示す決定木を作成するために，決定木を構成する質問に対する回答を Yes/No の二値だけになるように限定したものである[61]．この Yes/No チャートを用いて Computational Thinking を育成する試みについては，上述したように平嶋，林が試行的実践を行っている[35][36][37].

これらの研究をさらに進め，各教科内で活用可能な Computational Thinking 育成題材の開発を行う必要がある．そして，学習者の作成した誤ったチャートと最も近い正解チャートを検出し，そのチャートへの変更の方法を見つけることで，チャートの修正を誘導することができると考えられる．しかし，ベン図の解および Yes/No チャートの解が多様であることから，指導者が適切に学習者個々に指導するのは困難であることが想定される．しかし，そのことを適切に支援する学習システムは存在しない.

よって，学習者が自身で条件ブロックや図形を動かすことによって，ベン図や Yes/No チャートを組み立てることができる組み立て式ベン図学習システム (以下，ベン図学習システムと表記)，および組み立て式 Yes/No チャート学習システム (以下，Yes/No チャート学習システムと表記)を設計・開発することが重要であると考えられる．

2.3. 我が国のプログラミング教育の現状

世界的にプログラミング教育が本格的に実施されるようになり，必修化となった国も数多くあることが報告されている[38]．我が国においても，義務教育段階におけるプログラミング教育が必修化されることが決定し，2020 年から完全実施される新学習指導要領の中に盛り込まれている[39]．高等学校段階では情報 I，情報 II が設置され，プログラミングの学習内容が拡充された[40].

高等学校学習指導要領解説 情報編によれば，「情報 II」の目標として，情報システムや多様なデータの活用について理解を深め技能を習得し，情報技術の発展と社会の変化について理解を深めること，問題の発見・解決に向けて情報と情報技術を適切かつ効果的，創造的に活用する力を養うこと，情報と情報技術を適切に活用し，新たな価値の創造を目指し情報社会に主体的に参画し，その発展に寄与する態度を養うことが掲げられている[41]．これらの内容は，「情報 I」での学習内容を踏まえ，さらに発展的・創造的に情報技術を活用するための基本的なスキルを身につけ，それを効果的・創造的に活用するスキルの育成，およびそのような態度の涵養が期待されていることと捉えることができる.

文部科学省有識者会議「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)」には，「将来の予測が困難な時代の中で，これからの子供たちに求められるのは，これまでにないような全く新しい力ということではなく，従来からも重視されてきている読解力や論理的・創造的思考力，問題解決能力，人間性等について，加速度的に変化する社会の文脈の中での意義を改めて捉え直し，しっかりと発揮できるようにすることであると考えられる」と記載されており，この実現のために，「プログラミング的思考」の育成が掲げられている[13]．さらに我が国の高校におけるプログラミング教育では，コンピュー

タの働きを科学的に理解するとともに，知識・技能として，実際の問題解決にコンピュータを活用できるようにすること，学びに向かう力・人間性等として，発達の段階に即して，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の涵養，また，思考力・判断力・表現力として，どのような進路・職業を選択しても，これからの時代に共通に求められる「プログラミング的思考」を育成することが掲げられている[13]．総務省は，プログラミング人材育成を，「プログラミングに関する高度な技術者を直接的に育成するものではなく，発達段階に応じたプログラミングに関する教育を通じて，将来の高度 ICT 人材としての素地の構築・資質の発掘を図るもの」とした上で，年齢や校種を問わず，創造力，課題解決力，表現力，合理性，論理的思考力，意欲，コーディング・プログラミングスキル，コンピュータの原理の理解といった 7 項目を身につけることが期待されている[42]．

プログラミング的思考は，Computational Thinking 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[13]．またプログラミング的思考は，プログラミングの取り組みだけで育まれたり，働いたりするものではなく，思考力，判断力，表現力などを育む中にプログラミング的思考の育成につながるプログラミング体験を計画的に取り入れていくことが必要であると指摘されている[32]．つまり，我が国のプログラミング教育では，コンピュータを用いた汎用的な問題解決能力の育成，そして教科横断的でより汎用的なスキルの育成という観点が重要視されている．

しかし赤堀は，既存の各教科の中にプログラミング的思考を導入するクロスカリキュラムが育成において困難であること，そしてこのことが諸外国と異なっていると述べている[43]．また阪東らは，プログラミング的思考の考え方は Computational Thinking の概念に比べて矮小であることを指摘している[22]．よって，プログラミング的思考の育成を各教科内で育成することが求められている中で，既存の各教科内容とどう対応付けていくかについて検討すること，諸外国の Computational Thinking 育成に関する動向を踏まえること，そして，より広義な概念である Computational Thinking の育成も合わせて検討していくことが重要である．

また，我が国におけるプログラミング教育では，専門家養成という観点ではなく，コンピュータを用いた問題解決能力の育成およびその態度の涵養，教科を問わない形で創造的に問題解決を行おうとする態度の育成が重要である．よって，プログラミングに対する興味・関心や意義，理解の重要性を高めるとともに，プログラミングが実際に有用であるといった意識，プログラミングの知識は他にも応用できるといった意識といった，プログラミングに対する様々な意識を高めていく必要があると考えられる．しかし，プログラミングの有用感は興味・関心だけではなく，自分にとって役立つものと感じているか，内容の理解が重要であると感じているか，プログラミングで学んだ内容が他の分野や教科にも使えるものと気付いているかなど，その指し示す範囲は広い．よってこのような意識を育成することを志向したプログラミング教育を実施することが重要であると考えられる．

ここで，情報に関する意識の実態把握に関する先行研究として谷田らは，中学校技術・家

庭，技術分野「情報とコンピュータ」に関して，興味・関心，自己評価について調査を行い，学習者を類型化している[44]．他にもプログラミング学習における教材が学習者の情意的側面に及ぼす影響や，中学生の情報とコンピュータにおける学習の有用性についての先行研究が存在する[45][46]．しかし，生徒のレディネスとしての「プログラミングに対する様々な意識」の詳細については，十分な検討がなされていないのが現状である．よって，これらの調査を行うことが重要であると考えられる．

2.4. 創造性

本研究では Computational Thinking をプログラミング教育，およびプログラミング教育以外の教科において育成することを目指しているが，「Computational Thinking をプログラミング教育で育成する」ということに対し，Computational Thinking を育成することでプログラミングのコーディングスキル以上のものを身につけることが期待できる．そのようなスキルとして，創造性に着目する．

創造性は長い期間にわたって重要であると指摘されてきた．そして創造性には様々な定義が存在し，育成方法も多様である．例えば Guilford は，創造的な思考を支える特性には「感受性」，「流暢性」，「柔軟性」，「独創性」，「精緻性」，「再定義する力」の 6 つがあると指摘している[47]．そして，「収束的思考 (既知の情報から論理的に思考や推論を進め，唯一の正解に正しく・早く到達する思考)」と，「発散的思考 (既知の情報から様々な考えをめぐらせることによって，新たな物を生み出していく思考)」の 2 つがあり，創造性育成には発散的思考の育成が重要であると述べている[47]．Wallas は，創造的思考の過程として，「準備期」，「あたため期」，「啓示期」，「検証期」の 4 段階があると述べ，その過程が重要であると指摘している[48]．ムンツァートは，創造の過程に着目した上で，32 項目からなる創造性を測定する尺度を構成している[49]．Csikszentmihalyi は，創造性には，「creativity」と「Creativity (Big-C)」があるとし，creativity はごく個人的な範囲における創造性であり，個人の生活や空間の中で発揮され，その結果が他者から認められずとも自分自身の人生を豊かにしてくれるもの，Creativity は起点が個人であっても，公共における創造性，文化や物事の仕組み，生活のあり方などに改革をもたらす創造性であり，社会として人々が期待し求めるものであるとし，新規性のある事実であることを認める他者が必要であると述べている[50]．恩田は，「創造性とは新しい価値あるもの，またはアイディアを作り出す能力すなわち創造力，およびそれを基礎づける人格特性すなわち創造的人格である」と定義し，学校教育においては，児童・生徒一人一人にとって価値のある新しさを大切にする「自己実現の創造性」が基本的であると指摘している[51]．以上のように創造性には様々な研究があるが，Plucker et al. は，新規性・有用性を含む成果物を生み出す態度やプロセス，環境間の相互作用が関係していることが創造性研究の大きな共通項であると指摘している[52]．

このように創造性に関する研究は多く行われているが，創造性には「世の中にとって新しいもの」と「本人にとって新しいもの」という文脈がある．教育的には本人にとって新しい

ものを生み出し，それを世の中にとって新しいものを生み出そうすることへ接続することが重要である．そして各教科においても創造性を育成することを視野に入れていることから，創造性を「与えられた問題に存在する制約を，学習者がわかりながら超えること」と定義し，その育成を目指すこととする．

そして，プログラミングで創造性の育成が重要であると指摘されており[16][17][19]，プログラミングで創造性を育成する実践も報告されていることから[20]，プログラミングと創造性には一定の関連性が認められる．創造性を高める実践として，レズニックらによれば，創造性は育成できるものとした上で，創造性を育成するために Projects (プロジェクト), Passion (情熱)，Peers (仲間)，Play (遊び)の 4 項目を掲げ，Scratch を用いた成果について報告している[18]．以上のように，プログラミング教育においては，創造性を高めることが期待されていること，プログラミングは創造性育成に有用であることが指摘されている．普通教育で行うプログラミング教育では，コンピュータやプログラミングによる問題解決力の育成が重要である．そして新しいことを生み出す人格や態度の育成，および創造的問題解決へ生徒を向かわせるような方略の検討，題材の設定が重要となる．これらの基礎として生徒の情意面に着目した上で，レディネスとしての創造性の実態を把握することが重要であると考えられる．そのような創造性として「創造的態度」に着目する．

創造的態度とは，Schank and Childers によれば，「決まった方法やパターン化された問題解決ではなく，自ら問いを発し好奇心を持つこと,そして恐れずに常に改良しようとし，失敗から新しいものを産み出そうとする態度」[53]，久保薗によれば，「他人・自分自身・世界などに自分自身を関与させるときの，その人特有の様式であり，芸術などの創造に必要な才能や技術を指すものではなく，誰にでも備わっているもの」[54]などと定義されている．

創造的態度の先行研究として，例えば恩田らは，創造的態度として「自己統制力」「自発性」「衝動性持続性」「探求心」「独自性」「柔軟性」「精神集中力」の 8 項目を掲げている[51]．林は，理工系学生を対象とした調査を実施し，「挑戦性・探求心」「積極性・自信」「持続性・集中力」「独自性」「好奇心」「柔軟性」の 6 因子を構成している[55][56]．豊島・庭瀬は，恩田の創造的態度の項目をもとに中学生の創造的態度尺度を構成し，「努力・持続性」「自主・独自性」の 2 つの主要因子からなることを明らかにしている[57]．繁桝らは，日米の大学生対象に調査を実施し，「柔軟性」「分析性」「進取性」「持続性」「想像性」「協調性」の 6 因子を構成している[58].

しかし，これまでのプログラミング教育に関する先行研究では，創造性の概念が多義的であり，その育成方法やレディネスとしての役割については十分な検討はなされていないのが現状である．よって，創造的態度とプログラミングに対する様々な意識の関連性を把握することは重要な課題であると考えられる．

2.5. 柔軟性

Computational Thinking はコンピュータが行う一連の問題解決方法を明示的に表現する力，

すなわちコンピュータを用いた問題解決力として捉えられる. よって Computational Thinking と問題解決力の関連性に着目することが重要であり，Computational Thinking 育成によって問題解決力に付随するスキルを高めることが期待できる.

問題解決力の重要性は，他の教科や科目においても，もちろん重要視されてきた．もちろん，このことはプログラミングにおいてもその他の教科においても同様であり，世の中の多様な問題に対し取り組もうとする姿勢や態度の育成も合わせて重要である．OECD は，3 年に一度，義務教育修了段階である 15 歳の生徒を対象に学習到達度調査を行なっており，その中で問題解決における習熟度ならびに柔軟性について問うている[59]．Star and Johnson は，問題解決分野において，学習者が複数の戦略を知っていること，そして効率的な戦略を適切に選択するという柔軟な知識が重要であると指摘した上で，問題解決の柔軟性を向上させる方法について検討している[60]．つまり，問題解決力には柔軟性が関係していると考えられる.

以上のことから，Computational Thinking を育成することでコーディングスキル以外のスキルである軟性に着目し，柔軟性とプログラミングに対する様々な意識との関連を把握することが重要であると考えられる.

2.6. 問題の所在

上述したように，本研究では 21 世紀のデジタル時代に必要なスキルである Computational Thinking をプログラミング教育および各教科内でコーディングを用いずに育成すること，つまり「プログラミング教育において Computational Thinking を育成すること」，「各教科内でコーディングを用いずに Computational Thinking を育成すること」の 2 点を実現することが求められる.

1 点目の「プログラミング教育において Computational Thinking を育成すること」では，Scratch などのシステムを用いた事例が存在する．Wong and Cheung は，児童生徒がプログラミング活動を用いて Computational Thinking を身につけることが可能であると報告している[15]．よって，Computational Thinking を育成する上でプログラミングを活用することは有用である．その一方，Computational Thinking を身につけることでプログラミングのコーディングスキル以上のスキルを身につけることが期待できる. プログラミングは，他の教科より自由度が高く，学習者が問題を自由に変更することや様々な実現・実装方法などが存在するため，学習者それぞれが自身で考えたことを実現するために有用な手法の一つである．よってコーディングスキル以上のスキルとして創造性に着目する．Rotem et al.，Doleck et al.は，Computational Thinking と創造性には関連性があると指摘[16][17]している．また，問題解決力と関連性がある柔軟性にも着目する．Star and Johnson は，問題解決分野において，学習者が複数の戦略を知っていること，そして効率的な戦略を適切に選択するという柔軟な知識が重要であると指摘している[60]．また，プログラミングで Computational Thinking が育成できることを前提とすれば，プログラミングの力と創造性や柔軟性の向上には関連性がある

と想定される．また，我が国のプログラミング教育で育成が期待されているプログラミングに対する様々な意識と創造性との関連性を詳細に把握することで，プログラミング教育の意義も，より高まるのではないかと考えられる．しかし，プログラミングに対する様々な意識と創造性，柔軟性との関連性については十分に明らかにされていない．よって，本研究でこれらの関連性について検討を行う．

2 点目の「各教科内で Computational Thinking を育成すること」については，アンプラグドコンピューティングなど，コーディングを伴わない紙面上で行う実践が多い[32]．また，各教科内容とは別の課題を準備した上で Computational Thinking の育成を目指すことが多い．各教科で Computational Thinking を育成するためには，既存の教科内で扱われている題材を活用することで．各教科内で Computational Thinking を育成するということが可能となる．そのために，どの教科においても適用可能な Computational Thinking 育成課題として，分類課題に着目する．「分類」は小学校においても教えられており，論理的で網羅的な分類は，論理的思考の基礎として年齢を問わず重要である．そして分類に関する学習が成立しておらず，分類についての適切な思考経験が得られていないことが想定される．

平嶋，林らは，各教科で用いられている題材を活用するために，図形の分類課題を Computational Thinking 育成課題として設定し，その育成にベン図と Yes/No チャートを導入している．そして，Computational Thinking の育成に対し，学習者に個別最適化した上で適切な学習を提供するシステムの開発が重要であると考えられる．しかし，そのような機能を実装したシステムは存在しない．よって本研究では，Computational Thinking を育成するために，図形の分類課題を実装したベン図と Yes/No チャートを開発する．そして，Yes/No チャート学習システムの学習効果について評価を行った．

3. 創造性とプログラミングに対する様々な意識との関連性

本章では，創造性とプログラミングに対する様々な意識との関連性を把握するために，高校生を対象とした調査研究について述べる．最初に，用いた測定尺度について述べた上で，調査結果について述べる．

3.1. 測定尺度の決定

3.1.1. 創造性

創造的態度に関して，様々な尺度が開発されている．豊島・庭瀬[57]による創造的態度尺度は，主要因子が 2 つと単純化されており，創造的態度を詳細に検討する上では不十分である可能性がある．そして林による創造的態度尺度は大学生を対象としており，高校生にも理解できる内容であると考えられる．しかし，参考にしている穐山[62]や青柳[63]の文献がそれぞれ 1975，1980 年と古く，予備調査を経ていないことから，今後より詳細な検討が必要であると考えられる．繁桝らによる創造的態度尺度は，大学生を対象とした調査であり，国内外での比較も行っていることから，高校生の創造的態度を調査するにあたっての援用は十分に可能ではないかと考えられる．よって本研究では，繁桝ら[58]の創造的態度の項目を用いることとした．

3.1.2. 柔軟性

本研究における柔軟性は，上述した繁桝らの創造的態度尺度の因子である「柔軟性」を用いる．繁桝らの柔軟性因子は 12 項目から構成されており，その項目の内容も多岐にわたっている．よって柔軟性因子を再検討した上で，再検討後に抽出された因子とプログラミングに対する意識との関連性について分析を進めることが重要であると想定される．

3.1.3. プログラミングに対する様々な意識

文部科学省の議論の取りまとめや高等学校学習指導要領解説 情報編では，今後の社会において，決まった方法で手続きを効率的にこなすだけでなく，試行錯誤しながら新たな価値を生み出していくことや，コンピュータやプログラミングの仕組みを理解し，それらは人間が作り出したものであると理解すること，コンピュータやプログラミングを通した社会について理解すること，そして創造的な活用の重要性を感じることなどが重要である主旨が記載されている[41]．また，プログラミングに関係する周辺領域の理解の重要性について指摘している．よって高等学校では，小学校から系統的・体系的なプログラミング教育を行うためにコーディングなどを活用しながら，コンピュータを用いた問題解決や創造的に情報を活用する力の素地を身につけることが重要である．そして上述したプログラミングに対

する有用感などの様々な意識を高めるための育成方法を検討することが重要である.

ここで，我が国のプログラミング教育で身につける力として提言された議論の取りまとめ[13]を参考にし，プログラミングに対する様々な意識として「プログラミングに興味・関心がある」「プログラミングは自分にとって，いろいろ役立つと思う」,「プログラミングの内容を理解することは，自分にとって重要だと思う」,「プログラミングでやった内容は，プログラミング以外でも役立つと思う」,「プログラミングをやる意義はある」の 5 項目を抽出した.

以上のことを踏まえ, 創造性を高めるプログラミング教育の方向性を探るために, 創造的態度と有用感などのプログラミングに対する様々な意識に着目し，その関連性の実態を把握することが重要であると考えられる.

しかしこれまでに，創造的態度とプログラミングに対する様々な意識の関連性については，十分な検討がなされていない．よって，創造的態度とプログラミングに対する様々な意識の関連性を把握することは重要な課題であると考えられる.

3.2. 創造的態度とプログラミングに対する様々な意識との関連性

高校生のプログラミングの学習に際し, 身につけることが期待されている「プログラミングに対する様々な意識」と創造性の関連性を把握することを目指す．調査にあたっては，前節で述べた「創造的態度」尺度および議論の取りまとめを参考にして作成した「プログラミングに対する様々な意識」の項目を用い, 創造的態度とプログラミングに対する様々な意識の関連性について把握する調査を行った.

3.2.1. 調査の手続きと調査対象

本調査は, 2016 年 12 月に, 調査対象校の情報科の授業時間内に実施した. 調査対象者は, A 県内の公立高等学校 3 校の 1 年生計 226 名 (男子 92 名，女子 134 名)を対象とした．有効回答は計 197 名，有効回答率は 87.2%であった．全員高等学校でプログラミングは未履修であった.

3.2.2. 質問項目

文部科学省 有識者会議 議論の取りまとめを参考にして作成したプログラミングに対する様々な意識を把握する 5 項目，繁桝らが作成した「創造的態度」尺度 6 因子を準備した．いずれも「4:とても 3:まあまあ 2:あまり 1:まったく」の 4 件法で回答を求めた.

(1) プログラミングに対する様々な意識

生徒のプログラミングに対する様々な意識を把握するために，議論の取りまとめを元に, 5 項目を準備した．用いた項目を表 1 に示す (各項目の略称を文末の丸括弧内に示す).

表 1 プログラミングに対する様々な意識の各項目

(1)	プログラミングに興味・関心がある (プログラミングの興味・関心)
(2)	プログラミングは自分にとって, いろいろ役立つと思う (プログラミングの有用感)
(3)	プログラミングの内容を理解することは, 自分にとって 重要だと思う (プログラミング理解の重要性)
(4)	プログラミングでやった内容はプログラミング以外でも 役に立つと思う (プログラミングの応用期待感)
(5)	プログラミングをやる意義はあると思う (プログラミングに対する意義)

(2) 創造的態度尺度

生徒の創造的態度の状況を把握するために，調査項目として，繁桝ら[58]が作成した創造的態度尺度 (柔軟性，分析性，進取性，持続性，想像性，協調性の 6 因子)を準備した．

柔軟性は「連想が次々浮かぶ」など，視点の転換や多様な発想を生み出す因子，分析性は「問題を解く前にその問題の構造をよく考える」など，問題に対する見方や取り組み方についての因子，進取性は「新しいものや珍しいものが好きだ」など，新しいもの，珍しいものを観察し発見したりする因子，持続性は「物事を中途半端に終わらせるのは嫌いだ」など，問題の発見から解決までを自分の考えに基づき,諦めないで粘り強く追求していくような因子，想像性は「現実と異なることをよく考える」など，様々な新しいことを空想し,考えつくような因子，協調性は「集団で行動するときは全体の和が重要である」など，集団全体の調和を重視するような因子である．繁桝らは，大学生を対象とした調査を通して尺度を構成しているが，今回の調査対象者は高校生であるため，繁桝らの表現をそのまま用いた．また，繁桝らの示した項目において，生徒の回答に対する負担減のため，因子負荷量が 0.40 未満の項目をカットし，柔軟性 13 項目，分析性 9 項目，進取性 5 項目，持続性 5 項目，想像性 4 項目，協調性 5 項目を使用した．用いた項目を表 2 に示す．

表 2 創造的態度尺度の各項目

因子Ⅰ: 柔軟性 (13項目)	因子Ⅲ: 進取性 (5項目)
1 たとえ話がうまい.	1 好奇心が強い.
2 困ったとき, 解決方法をすぐにおもいつく.	2 生活を便利にするためにいろいろと工夫する.
3 いろいろな立場からの見方ができる.	3 新しいものや, 珍しいものが好きだ.
4 頭のきりかえは早いほうだ.	4 ものを作るのが上手だ.
5 連想が次々に浮かぶ.	5 今まで誰も考えたことのないようなすばらしいものを作りたい.
6 激しい議論を楽しむことができる.	因子Ⅳ: 持続性 (5項目)
7 話題が豊富である.	1 集中すると周りのことが気にならない.
8 人に個性的だとよく言われる.	2 ものごとにこだわりがある.
9 幅広い知識がある.	3 ものごとを途中でやめたり, 中途半端に行うのは嫌いだ.
10 人が思いつかないようなことを考え出すと, よく言われる.	4 容易に物事をあきらめない.
11 違ったものの中に共通点をみつけるのがうまい.	5 確固とした意見を持っている.
12 人からどう解決したらよいか聞かれることが多い.	因子Ⅴ: 想像性 (4項目)
13 ものわかりがよいほうだ.	1 新しいことを考えつくと, とてもうれしい.
因子Ⅱ: 分析性 (9項目)	2 一つのことに集中するよりも, あれこれいろいろやっている方がよい.
1 問題を解く前に, その問題の構造をよく考える.	3 空想したり, 現実と異なることを考えることが多い.
2 問題が解決しても, 他にもっとよい解き方はないか考える.	4 自分は変わっていると思う.
3 アイディアが浮かんだとき, その実現手段も考える.	因子Ⅵ: 協調性 (5項目)
4 ものごとに取り掛かる前に, まずその手順を考える.	1 集団で行動するときは, その集団の「和」が大事だと思う.
5 問題をいくつかの小さな問題にわけることが容易にできる.	2 目上の人に賛成してもらうことはとても重要だと思う.
6 ものごとの本質を理解する力には自分にはあると信じている.	3 誰かと協力して作業することが多い.
7 細かく観察することが好きだ.	4 やり方がわからないときは, 人によく相談する.
8 ものごとの本質を考えようとする.	5 人が困っているのを見れば, 助けようとすることが多い.
9 ものごとの構造をよく理解することができる.	

(3) 柔軟性

柔軟性の状況を把握するために，調査項目として，繁桝ら[58]が作成した創造的態度尺度の柔軟性を用いた．ここで，柔軟性因子は「連想が次々浮かぶなど，視点の転換や多様な発想を生み出す態度」である．繁桝らは，国内外の大学生を対象とした調査により尺度を構成しているが，今回の調査対象者は高校生であり，その内容は十分に理解できると判断できるため，繁桝らの下位尺度をそのまま用いた．また，因子負荷量 0.45 未満の項目，およびほとんど同じ意味の項目をカットし，10 項目(それぞれ項目 1,2,..,10 とする)を調査に用いた．柔軟性因子の下位尺度について表 3 に示す．

表 3 柔軟性に関する項目

項目 1	たとえ話がうまい.
項目 2	話題が豊富である.
項目 3	違ったものの中に共通点をみつけるのがうまい.
項目 4	困ったとき, 解決方法をすぐにおもいつく.
項目 5	幅広い知識がある.
項目 6	人が思いつかないようなことを考え出すと, よく言われる.
項目 7	連想が次々に浮かぶ.
項目 8	激しい議論を楽しむことができる.
項目 9	人からどう解決したらよいか聞かれることが多い.
項目 10	いろいろな立場からの見方ができる.

3.2.3. 分析の手続き

分析に先立ち，「プログラミングに対する様々な意識」と「創造的態度」尺度について，全体，性別ごとに集計を行った．次に，「プログラミングに対する様々な意識」と「創造的態度」尺度のそれぞれの項目・因子間における関連性を把握するために，Pearson の積率相関係数を求めた．

そして，創造的態度の柔軟性因子に対し，最尤法・Promax 回転を用いた探索的因子分析を行い，柔軟性因子の再検討を行なった．そして，柔軟性因子間，およびプログラミングに対する様々な意識間における関連性について検討するために，Pearson の積率相関係数を求め，項目間の相関を確認した．柔軟性に関する分析では，因子構造を明らかにするため，性別による区分を行わずに分析を実施した．

3.3. 結果と考察

3.3.1. 記述統計量

(1) プログラミングに対する様々な意識

「プログラミングに対する様々な意識」の各項目における全体平均値，男女別平均値を求めた．そして男女間の平均値を対応のない *t* 検定で比較した．その結果を表 4 に示す．

表 4 より，「プログラミングに対する興味・関心」，「プログラミングの応用期待感」の全体平均値は中位点である 2.50 より低く，その他のプログラミングに対する様々な意識の各項目の全体平均値は，中位点である 2.50 より高かった．男子だけで見れば，全ての項目が中位点の 2.50 より高く，女子だけで見れば，全ての項目が中位点の 2.50 より低かった．次に，プログラミングに対する様々な意識の各項目の平均値に性差があるかについて，対応のない *t* 検定を行った．その結果，プログラミングに対する様々な意識の全ての項目に対し，有意差が見られた．そしていずれの平均値も男子の方が女子よりも高かった．

このことから，男子の方が，プログラミングの様々な観点において興味や意識が高く，女子よりもプログラミングに関心を示す傾向があることが示唆された．

表 4 プログラミングに対する様々な意識の状況

		性別		全体	群間の差の検定
		男子	女子		
		(n=81)	(n=116)	(n=197)	(t 値)
プログラミングに対する興味・関心	平均	2.56	2.04	2.25	$t(195) = 3.56$ **
	S.D.	1.06	0.95	1.02	
プログラミングの有用感	平均	2.84	2.49	2.69	$t(195) = 3.85$ **
	S.D.	0.86	0.91	0.90	
プログラミング理解の重要性	平均	2.98	2.29	2.51	$t(195) = 4.48$ **
	S.D.	0.80	0.85	0.90	
プログラミングの応用期待感	平均	2.77	2.29	2.49	$t(195) = 3.96$ **
	S.D.	0.81	0.83	0.85	
プログラミングに対する意義	平均	2.93	2.31	2.56	$t(195) = 5.12$ **
	S.D.	0.85	0.82	0.88	

注: **p<.01

(2) 創造的態度尺度

創造的態度尺度各因子における全体平均値，男女別平均値を求めた．そして男女間の平均値を対応のない t 検定で比較した．その結果を表 5 に示す．

表 5 より，「柔軟性」，「分析性」の全体平均値は中位点である 2.50 より低く，その他の創造的態度各因子の全体平均値は，中位点である 2.50 より高かった．男子だけで見れば，「柔軟性」を除いた「分析性」，「進取性」，「持続性」，「想像性」，「協調性」因子では，中位点の 2.50 より高かった．女子だけで見れば，「柔軟性」，「分析性」を除いた「進取性」，「持続性」，「想像性」，「協調性」因子では，中位点の 2.50 より低かった．

次に，創造的態度尺度の各因子の平均値に性差があるかについて，対応のない t 検定を行った．その結果，「柔軟性」は 5%水準，「分析性」は 1%水準で有意差が見られ，男子の方が女子より平均値が高かった．他の項目については，有意差は見られなかった．

このことから，男子の方が，連想が次々浮かぶような視点の転換や発想を生み出すような柔軟性や，問題を解く前にその問題を考えるなど，問題に対する見方や取り組み方である分析性が高く，そのような内容を含む問題に対して女子よりも関心を示す傾向があることが示唆された．

表 5 創造的態度の状況

因子		性別 男子 (n=81)	性別 女子 (n=116)	全体 (n=197)	群間の差の検定 (t 値)
柔軟性	平均	2.45	2.29	2.35	$t(195)$= 2.22 *
	S.D.	0.46	0.54	0.51	
分析性	平均	2.52	2.22	2.34	$t(195)$ = 3.87 **
	S.D.	0.56	0.52	0.55	
進取性	平均	2.78	2.64	2.70	$t(195)$ = 1.63
	S.D.	0.58	0.62	0.60	
持続性	平均	2.76	2.65	2.70	$t(195)$ = 1.29
	S.D.	0.56	0.56	0.56	
想像性	平均	2.81	2.77	2.78	$t(195)$ = 0.48
	S.D.	0.53	0.55	0.54	
協調性	平均	2.91	3.02	2.97	$t(195)$ = 1.41
	S.D.	0.50	0.50	0.50	

注: *p <.05, **p <.01

(3) 柔軟性の各項目

生徒の柔軟性の各項目の集計結果について表 6 に示す．表 6 より，全体の「連想が次々に浮かぶ」, 男子の「連想が次々に浮かぶ」「人からどう解決したらよいか聞かれることが多い」「いろいろな立場からの見方ができる」の項目の平均値は，中位点の 2.50 より高く，その他の項目の平均値は，中位点の 2.50 より低かった．そして，標準偏差は「激しい議論を楽しむことができる」の項目が一番大きく，生徒間でばらつきが見られた．

表 6 生徒の柔軟性の各尺度の状況

		平均	S.D.
1.	たとえ話がうまい	2.18	0.77
2.	話題が豊富である	2.12	0.78
3.	違ったものの中に共通点を みつけるのがうまい	2.25	0.76
4.	困ったとき, 解決方法を すぐにおもいつく	2.22	0.73
5.	幅広い知識がある	2.36	0.82
6.	人が思いつかないようなことを 考え出すと, よく言われる	2.29	0.88
7.	連想が次々に浮かぶ	2.52	0.84
8.	激しい議論を楽しむことができる	2.34	0.95
9.	人からどう解決したらよいか 聞かれることが多い	2.45	0.80
10.	いろいろな立場からの見方ができる	2.38	0.77

(n=197)

3.3.2. プログラミングに対する様々な意識と創造的態度の関連性

(1) 創造的態度尺度とプログラミングに対する様々な意識の各項目間の相関係数

創造的態度とプログラミングに対する様々な意識との関係を把握するために，創造的態度とプログラミングに対する様々な意識の相関係数 (ピアソンの積率相関係数)を求め，相関係数の検定を行った．男女を分けずに全体で求めた結果を表 7 に示す.

表 7 より，有意な相関を持つ項目は多く見られた一方で，想像性とプログラミングの興味・関心間, 想像性とプログラミングの有用感間, 想像性とプログラミング理解の重要性間, 想像性とプログラミングの応用期待感間, 想像性とプログラミングに対する意義間, 想像性と協調性間，そして協調性とプログラミングの興味・関心間，協調性とプログラミングの有用感間, 協調性とプログラミング理解の重要性間，協調性とプログラミングの応用期待感間, 協調性とプログラミングに対する意義間においては，有意な相関関係が認められなかった.

表 7 創造的態度，プログラミングに対する様々な意識の相関係数および検定結果 (全体)

	(2)	(3)	(4)	(5)	柔軟性	分析性	進取性	持続性	想像性	協調性
(1)	0.59 **	0.60 **	0.60 **	0.56 **	0.25 **	0.34 **	0.37 **	0.30 **	0.14	0.11
(2)		0.79 **	0.74 **	0.66 **	0.28 **	0.45 **	0.25 **	0.23 *	0.03	0.03
(3)			0.77 **	0.63 **	0.34 **	0.48 **	0.26 **	0.24 *	0.04	0.13
(4)				0.64 **	0.35 **	0.47 **	0.32 **	0.24 *	0.05	0.14
(5)					0.30 **	0.45 **	0.27 **	0.22 *	0.11	0.16
柔軟性						0.64 **	0.49 **	0.46 **	0.26 **	0.23 *
分析性							0.55 **	0.52 **	0.28 **	0.31 **
進取性								0.63 **	0.48 **	0.41 **
持続性									0.43 **	0.38 **
想像性										0.16

(1) プログラミングの興味・関心, (2) プログラミングの有用感, (3) プログラミング理解の重要性
(4) プログラミングの応用期待感, (5) プログラミングに対する意義
*p<.05, **p<.01

次に，男子のみで求めた結果を表 8 に示す.

表 8 より，男子では有意な相関を持つ項目は多く見られたが，柔軟性とプログラミングの興味・関心間，進取性とプログラミングの有用感間，想像性とプログラミングの有用感間，想像性とプログラミング理解の重要性間，想像性とプログラミングの応用期待感間，想像性と協調性間，協調性とプログラミングの有用感間では相関が認められなかった.

表 8 創造的態度，プログラミングに対する様々な意識の相関係数および検定結果 (男子)

	(2)	(3)	(4)	(5)	柔軟性	分析性	進取性	持続性	想像性	協調性
(1)	0.60 **	0.58 **	0.58 **	0.59 **	0.20	0.32 **	0.36 **	0.38 **	0.23 *	0.27 **
(2)		0.77 **	0.66 **	0.58 **	0.27 **	0.36 **	0.20	0.30 **	0.11	0.00
(3)			0.72 **	0.60 **	0.30 **	0.40 **	0.27 **	0.33 **	0.06	0.23 *
(4)				0.67 **	0.40 **	0.46 **	0.37 **	0.31 **	0.16	0.32 **
(5)					0.29 **	0.39 **	0.35 **	0.39 **	0.27 **	0.32 **
柔軟性						0.60 **	0.47 **	0.45 **	0.21 *	0.29 **
分析性							0.59 **	0.54 **	0.39 **	0.32 **
進取性								0.63 **	0.47 **	0.54 **
持続性									0.43 **	0.40 **
想像性										0.20

(1) プログラミングの興味・関心, (2) プログラミングの有用感, (3) プログラミング理解の重要性
(4) プログラミングの応用期待感, (5) プログラミングに対する意義
*p<.05, **p<.01

次に，女子のみで求めた結果を表 9 に示す.

表 9 より，女子においても有意な相関を持つ項目は同様に多く見られたが，進取性とプログラミングの意義間，持続性とプログラミングの有用感間，持続性とプログラミング理解の重要性間，持続性とプログラミングの応用期待感間，持続性とプログラミングに対する意義間，想像性とプログラミングの興味・関心間，想像性とプログラミングの有用感間，想像性とプログラミング理解の重要性間，想像性とプログラミングの応用期待感間，想像性とプログラミングに対する意義間，想像性と協調性間，協調性とプログラミングの興味・関心間，協調性とプログラミングの有用感間，協調性とプログラミング理解の重要性間，協調性とプログラミングの応用期待感間，協調性とプログラミングに対する意義間で相関は認められなかった.

表 9 創造的態度，プログラミングに対する様々な意識の相関係数および検定結果(女子)

	(2)	(3)	(4)	(5)	柔軟性	分析性	進取性	持続性	想像性	協調性
(1)	0.54 **	0.56 **	0.57 **	0.47 **	0.22 *	0.26 **	0.36 **	0.21 *	0.07	0.04
(2)		0.77 **	0.76 **	0.66 **	0.22 *	0.45 **	0.25 **	0.16	-0.02	0.10
(3)			0.77 **	0.58 **	0.31 **	0.47 **	0.22 *	0.15	0.03	0.12
(4)				0.55 **	0.28 **	0.40 **	0.26 **	0.17	-0.03	0.07
(5)					0.24 *	0.40 **	0.17	0.07	0.00	0.14
柔軟性						0.65 **	0.49 **	0.46 **	0.30 **	0.23 *
分析性							0.52 **	0.50 **	0.21 *	0.37 **
進取性								0.62 **	0.49 **	0.36 **
持続性									0.43 **	0.38 **
想像性										0.13

(1) プログラミングの興味・関心, (2) プログラミングの有用感, (3) プログラミング理解の重要性
(4) プログラミングの応用期待感, (5) プログラミングに対する意義
*p<.05, **p<.01

これらの結果から，創造的態度因子間では，想像性と協調性間を除いて全体，男女いずれ

においても相関が認められ，柔軟性，進取性，持続性，想像性，協調性とプログラミングに対する様々な意識の各項目間における相関については，性差がある実態が把握された．

なお，比較する創造的態度とプログラミングに対する様々な意識のそれぞれ 2 つの変量の間に線形関係を想定すること，そして相関関係は因果関係を含まないことに注意する必要がある．

(2) 創造的態度尺度の柔軟性の相関係数

柔軟性の各項目間における関連性を把握するために，それぞれの項目間に対して Pearson の積率相関係数を求め項目間の相関を確認した．その結果を表 10 に示す．表 10 より，柔軟性の各項目間では正の相関が見られ，相関係数に関する検定の結果，いずれの項目間においても 1%水準で有意であった．

表 10 生徒の柔軟性の各項目間の相関

	項目 1	項目 2	項目 3	項目 4	項目 5	項目 6	項目 7	項目 8	項目 9	項目 10
項目 1	1.00	0.38**	0.38**	0.65**	0.39**	0.29**	0.32**	0.38**	0.30**	0.47**
項目 2		1.00	0.35**	0.33**	0.45**	0.40**	0.44**	0.33**	0.40**	0.25**
項目 3			1.00	0.35**	0.41**	0.53**	0.35**	0.35**	0.39**	0.45**
項目 4				1.00	0.36**	0.40**	0.30**	0.36**	0.33**	0.52**
項目 5					1.00	0.43**	0.33**	0.29**	0.41**	0.37**
項目 6						1.00	0.45**	0.42**	0.44**	0.40**
項目 7							1.00	0.47**	0.29**	0.35**
項目 8								1.00	0.22**	0.44**
項目 9									1.00	0.23**
項目 10										1.00

注: $^{**}p<.01$

項目 1: たとえ話がうまい, 項目 2: 話題が豊富である

項目 3: 違ったものの中に共通点をみつけるのがうまい

項目 4: 困ったとき, 解決方法をすぐにおもいつく

項目 5: 幅広い知識がある, 項目 6: 人が思いつかないようなことを考え出すと, よく言われる

項目 7: 連想が次々に浮かぶ, 項目 8: 激しい議論を楽しむことができる

項目 9: 人からどう解決したらよいか聞かれることが多い

項目 10: いろいろな立場からの見方ができる

(3) 創造的態度尺度の柔軟性の再検討

創造的態度の柔軟性 10 項目に対し，最尤法・Promax 回転を用いた探索的因子分析を行い，柔軟性因子の再検討を行なった．この分析は，田中ら[64]の文献を参考にして進めた．

柔軟性因子の 10 項目の天井効果・フロア効果を，天井効果は平均+標準偏差「取り得る値以上」，フロア効果は平均-標準偏差が「取り得る最低値以下」となるという基準で確認したところ[64]，いずれの項目も問題はなかった．

柔軟性因子の再検討には，最尤法，Promax 回転による探索的因子分析を用いた．その際，因子抽出法としては，最尤法で初期解を得た後，固有値が 1.0 以上で極端な減衰が生じる直前の因子数を採用する Guttman-Kaiser 基準[65]により，因子数 2 が適切であると判断した．そして，探索的因子分析を行い，因子負荷量が 0.450 未満の 2 項目 (項目 8,10)を削除した[66]．さらに探索的因子分析を行い，6 項目からなる因子 I と 2 項目からなる因子 II に統合された．全項目の因子負荷量は 0.50 以上であった．

次に，内部一貫性の確認のために，Cronbach の α 係数を求めたところ，因子 I の Cronbach の α 係数は 0.80，因子 II の Cronbach の α 係数は 0.79 であり，いずれの項目も 0.70 より大きく，内部一貫性が高いことが確認された[67]．そして，確認的因子分析を行い，尺度の一次元性，収束妥当性，弁別妥当性を確認した．適合度指標は，CFI＝0.982，RMSEA＝0.047，SRMR＝0.034 であり，尺度の一次元性が確認された (基準値は，CFI＞0.900, RMSEA＜0.050, SRMR＜0.080[68]，尺度の一次元性は，確認的因子分析で求められる適合度指標が満足できる値かどうかで確認できる[64])．

さらに，収束妥当性を確認するために，因子負荷量が 0.500 以上および標準誤差の 2 倍以上であるという基準で評価したところ[65]，因子 I の標準誤差は 0.042，因子 II の標準誤差は 0.196 であることから，いずれの項目においても収束妥当性を満たしていることが確認できた．弁別妥当性は，各因子間の相関係数の 95%信頼区間がいずれも 1 を含んでいないことを基準として評価したところ，因子間相関が 0.53 であったことから，弁別妥当性が満たされていることが確認された．以上の結果から，これらの因子の適合度は良好であると結論づけた．

因子 I は，「人から思いつかないようなことを考え出すとよく言われる」「人からどう解決したらよいか聞かれることが多い」などといった他者から指摘され，話題が豊富であるなどのコミュニケーションにおける柔軟性，および幅広い知識がありいろいろなことが思い浮かぶなどの柔軟性から構成されている．これらは，他者との関わりの中で発揮され，そして知識を活用できるといった柔軟性であることから，因子 I を「知識・対外型柔軟性」とした．因子 II は，たとえ話がうまいなど，その場に応じて適切な言葉や考えを用いることができ，困った時に解決方法をすぐに思いつくなど，状況に応じて適切な解決方法を用いることができる柔軟性から構成されている．これらは状況に応じた思考ができる柔軟性であることから，因子 II を「臨機応変・問題解決型柔軟性」とした．上述した内容についての結果を表 11 に示す．

表 11 柔軟性因子に対する探索的因子分析の結果

	因子 I	因子 II
人が思いつかないようなことを考え出すと，よく言われる．	0.867	
違ったものの中に共通点をみつけるのがうまい．	0.635	
人からどう解決したらよいか聞かれることが多い．	0.634	
幅広い知識がある．	0.587	
話題が豊富である．	0.574	
連想が次々に浮かぶ．	0.566	
たとえ話がうまい．		1.090
困ったとき，解決方法をすぐにおもいつく．		0.537
因子負荷量平方和	2.620	1.522
因子寄与率 (%)	0.328	0.190
累積寄与率 (%)	0.328	0.518
因子間相関		0.531

上述の「知識・対外型柔軟性」,「臨機応変・問題解決型柔軟性」因子の平均値と標準偏差を表 12 に示す．表 12 より,「知識・対外型柔軟性」および「臨機応変・問題解決型柔軟性」のいずれにおいても，中位点である 2.50 より平均値が低かった．

表 12 柔軟性 2 因子の状況

	平均	S.D.
知識・対外型柔軟性	2.33	0.58
臨機応変・問題解決型柔軟性	2.20	0.68

(n=197)

(4) 柔軟性因子とプログラミングに対する様々な意識の各項目間の関連性

表 12 に示した柔軟性 2 因子とプログラミングに対する様々な意識の関連性を把握するために，柔軟性の各因子とプログラミングに対する様々な意識それぞれの項目間に対し，Pearson の積率相関係数を求め，項目間の相関を確認した．その結果を表 13 に示す．

表 13 プログラミングに対する様々な意識と柔軟性 2 因子との相関係数

	知識 · 対外型柔軟性	臨機応変 · 問題解決型柔軟性	相関係数の差の検定 (p 値)
プログラミングの興味 · 関心	0.16 *	0.28 **	0.22
プログラミングの有用感	0.19 **	0.25 **	0.54
プログラミング理解の重要性	0.21 **	0.32 **	0.24
プログラミングの応用期待感	0.20 **	0.39 **	0.04 *
プログラミングに対する意義	0.17 *	0.34 **	0.07

注: $^{*}p < .05$, $^{**}p < .01$

表 13 より，プログラミングに対する様々な意識 5 項目と「知識・対外型柔軟性」因子，「臨機応変・問題解決型柔軟性」因子のいずれにおいても正の相関が見られた．そして，相関係数に関する検定の結果，「知識・対外型柔軟性」と「プログラミングの興味・関心」および「プログラミングの意義」それぞれの相関係数は 5%水準で有意，その他の相関係数は 1%水準で有意であった．そして，プログラミングに対する様々な意識のそれぞれの項目に対する「知識・対外型柔軟性」因子との相関係数および「臨機応変・問題解決型柔軟性」因子との相関係数の差について，相関係数の有意差検定を行ったところ，「プログラミングの応用期待感」において，「知識・対外型柔軟性」因子と「臨機応変・問題解決型柔軟性」因子の相関係数に 5%水準で有意差が見られた．その他の因子間では有意差は見られなかった．

3.3.3. 考察

本調査の結果を以下に示す．

(i) 男女間におけるプログラミングに対する様々な意識の全項目の各平均値には有意差が見られ，いずれも男子が女子より平均値が高かった．

(ii) 男女間における創造的態度尺度の全因子の各平均値には有意差が見られ，いずれも男子が女子より平均値が高かった．

(iii) 相関については，プログラミングに対する様々な意識の全項目間，各創造的態度の関係，柔軟性，分析性，進取性とプログラミングに対する意識の大半の項目間で，相関が認められた．一方，持続性，想像性，協調性とプログラミングに対する様々な意識の項目間での相関は性差が認められる項目が多数見られた．また，プログラミングに対する様々な意識の 5 項目と創造的態度の多くの因子間において，関連性が認められた．

(iv) 柔軟性，分析性，進取性については，プログラミングに対する様々な意識の 5 項目のいずれも有意な関連性が認められ，いずれも男子が女子よりも平均値が高かった．

(v) 柔軟性因子を再検討し，「知識・対外型柔軟性」因子，「臨機応変・問題解決型柔軟性」

因子が抽出された.

(vi) 柔軟性 2 因子とプログラミングに対する様々な意識のそれぞれの項目には関連性が認められた．また「プログラミングの応用期待感」において,「知識・対外型柔軟性」因子と「臨機応変・問題解決型柔軟性」因子の相関係数に有意差が見られ,「臨機応変・問題解決型柔軟性」の方が強い相関があることが認められた.

(i), (ii), (iii), (iv)について考察を行う.

プログラミングの知識や技術が他の分野においても使えるといった意識および創造性を高めるようなプログラミング教育を推進することが重要であるが，男女を分けてプログラミングの授業を行うことは困難であることから，プログラミングに対する様々な意識および創造的態度には性差が存在することを踏まえたカリキュラムの設計が求められる．また，プログラミングに対する様々な意識の各項目間において正の相関が見られた. つまり，プログラミングの各項目のいずれかが高い生徒は，プログラミングの他の項目も高い傾向がみられる．このことから，例えばプログラミングの興味・関心を高める活動の中で，プログラミングで学んだ内容がプログラミング以外でも活用できることについて考察させる活動が効果的である可能性がある.

男女間で比較すると，男子の方がプログラミングやコンピュータに対して興味・関心・有用性を感じている一方で，女子はあまりプログラミングやコンピュータに対して興味・関心・有用性を感じていない現状があるということが示唆された. 男女全体では，プログラミングやコンピュータに対して興味・関心・有用性を感じている生徒とそうでない生徒間では, プログラミングに対する興味・関心に差が見られた. 創造的態度に関しては, 全体として「柔軟性」「分析性」「進取性」「持続性」「想像性」の項目が高い生徒の方がプログラミング教育に対する興味・関心が高い傾向があること，および男女間でプログラミングに対する興味・関心に関連性のあるレディネスとしての創造的態度が異なり，男子の方が「柔軟性」「分析性」が高く，そのような内容を含む問題に対して女子よりも関心を示す傾向が見られた.

このことから，男子では，プログラミングの興味・関心を高めるために，プログラミングの有用感や重要性を感じさせることやコンピュータの仕組みを理解させるような活動が有効である可能性がある．また，持続性，想像性，協調性とプログラミングに対する様々な意識と関連性が見られることから，そのような要素を取り入れた実践が，少なくとも男子では有効であると考えられる．ここで，持続性は物事を中途半端に終わらせるのは嫌いだなどといった，問題の発見から解決までを自分の考えに基づき，諦めないで粘り強く追求していくこと，想像性は現実と異なることをよく考えるなど，様々な新しいことを空想し,考えつくこと，協調性は集団で行動するときは全体の和が重要であるなど，集団全体の調和を重視するという因子である[58].

女子は，プログラミングやコンピュータに対して興味・関心・有用性を感じている生徒および「コンピュータを作業の効率化を図るために使うより，創造的な活動に使うことの方が

重要だ」と感じている生徒間では，プログラミングに対する興味・関心に差が見られた．このことから，創造的な活動に使うことの方が重要だと感じさせる活動を展開することが重要であると考えられる．また，柔軟性，分析性，進取性を優先的に高めることが有効ではないかと考えられる．ここで柔軟性は連想が次々浮かぶなど，視点の転換や多様な発想を生み出すこと，分析性は問題を解く前にその問題の構造をよく考えるなど，問題に対する見方や取り組み方のこと，進取性は新しいものや珍しいものが好きだなど，新しいもの，珍しいものを観察し発見したりするという因子である[58]．さらに持続性，想像性，協調性も，柔軟性や分析性，進取性と相関を有することから，創造的態度をバランスよく育成することが，プログラミングに対する様々な意識を高める上で重要であると考えられる．

よって，性別によって有効な実践方法に相違があることが示唆された．そして創造的態度を全体的に高めていくことが，プログラミングに対する様々な意識を高める上で有効となる可能性が示唆された．また，これらの性差があるという現状を踏まえた取り組みや，レディネスの様々な段階に対応した題材を複数準備することなども重要であると考えられる．

しかしこれは，男女でレディネスとしての創造的態度の中で「柔軟性」「分析性」に違いがあること，プログラミングに対する様々な意識が異なることを示唆しているにとどまっている．よって今後，調査結果を踏まえた題材を開発した上で，その効果を実践的に検討する必要が求められる．

次に(v), (vi)について考察を行う．

柔軟性の各項目間では正の相関が見られ，いずれの項目間も 1%水準で有意であった．このことから，話題が豊富な生徒や，たとえ話がうまい生徒に対し，違ったものの中にパターンがあるかについて考えさせるなどといった活動が効果的である可能性がある．

次に，プログラミングに対する様々な意識 5 項目と「知識・対外型柔軟性」因子，「臨機応変・問題解決型柔軟性」因子のいずれにおいても正の相関が見られ，いずれの項目も 5%水準もしくは 1%水準で有意であった．そして，「臨機応変・問題解決型柔軟性」と「プログラミングの応用期待感」の項目間の相関係数は，他の項目間における相関係数と比較して大きい傾向が見られた．よって，臨機応変に問題解決を行なう中で，プログラミングが他でも活用できることを意識させる活動が効果的である可能性がある．

また，「知識・対外型柔軟性」および「臨機応変・問題解決型柔軟性」とプログラミングに対する様々な意識との間に相関が認められることから，知識を活用してパターンを見つけること，連想を促すこと，そしてある話を別の話に置き換えるパラフレーズなどを行わせるような活動や題材が有用である可能性が想定される．

そして，違ったものの中にパターンがあるかについて考えさせる機会を与える活動を行うことは，Computational Thinking の「分解」や「パターン化」を育成することにも繋がり，共通部分を見つけてそこから要素を取り出す「抽象化」と関連性があると考えられる．よって，柔軟性を高める活動が，Computational Thinking 育成を指向した活動に有用となる可能性がある．

よって，創造的問題解決を行うためのレディネスとしての柔軟性がプログラミングに対する様々な意識と関連性があり，柔軟性を高めることを意図した活動が効果的である可能性があることが示唆された．そして，新しく構成した柔軟性 2 因子を満たすような内容を取り入れていくことが効果的である可能性が示唆された．

4. Computational Thinking を育成する学習支援システムの開発

本章では，Computational Thinking を育成する実践モデルと課題を開発する．そして既存の教育で用いられている既存の課題を Computational Thinking の課題として捉え，これまでの学習内容を深く理解させることを目指す.

4.1. 分類課題

本研究では，各教科で Computational Thinking もしくはプログラミング的思考を育成することを目指している．そして本研究では，その実現のために既存の教科内で扱われている題材を活用した上で Computational Thinking を育成することに着目する．このことを実現することによって，我が国のプログラミング教育で求められている「各教科内でプログラミング的思考を育成する」ということが可能となる．そのために，どの教科においても適用可能な Computational Thinking 育成教材として，分類課題に着目する．

「分類」は，小学校においても図形の分類を行う活動などで指導されている．そして論理的に，網羅的に分類することが論理的思考の基礎となると考えられる．そして，論理的思考の重要性はさらに増しており，生徒，学生のみならず，社会人においてもそのことは同様に重要である．

その一方で，現在行われている分類に関する学習は，論理的思考を育成する上で適切ではない可能性がある．小学校などの学校教育で行われている分類は，分類条件があらかじめ与えられており，その条件を満たす図形などを配置するといったものである．このことは，与えられた条件に合うように「ただ配置する」といった活動になっている可能性がある．網羅的に分類するためには，図形などから特徴を見出し，自身で条件を作成することが必要となるが，そのような学習とはなっていない．よって，学習者に上述したような学習を提供することが重要であり，分類について適切な学習が成立していないのは，分類についての適切な思考経験が得られていない/与えることができていないためであると想定される．

また，人が学ぶためには，経験することが重要であると同時に，単に経験するだけでは不十分であると報告されている[70][71][72]．つまり，分類は教えられてはいるものの，それについての適切な思考経験を与えられておらず，その結果として分類のための論理的な思考が習得されていないということである．

物事を論理的に，網羅的に，他者がわかるような形に分類するためには，Computational Thinking の「パターン抽出(一般化)」と「抽象化」が不可欠である．さらに，分類の実行には，Computational Thinking の「手続化」が必要となる．また，分類を段階的に行っていくことは，分類という課題に対して Computational Thinking の「分解」を行うことである．よって分類課題は，Computational Thinking の基本要素をすべて含む課題となっていると言える．

ここで本研究では，平嶋らの Kit building [73][74]の考え方をベースとし，必要十分な部品を学習者に提供し，それを組み立てるためにベン図，Yes/No チャートの作成やプログラム化を学習者に行わせる．それによって関係的理解[75]を促すことに焦点を当てた課題を開発する．分類課題に取り組む際においては，個々の部品に相当するものについては既に知っていることが前提としており，それらの「関係性」を把握することが学習すべき内容である．この実現のために，学習者に Computational Thinking の課題として捉え直した課題を提供する．ここで，必要十分な部品を学習者に与えることは，本課題の学習意義を本質的に損なうことにはならない．以上のことを踏まえ，Computational Thinking 育成課題を開発する．

4.2. 分類課題の Computational Thinking 課題化

分類とは，対象が属するカテゴリ表象を決定することである．そして，この分類を実行するための，段階的な手続きを決定木と呼ぶ．決定木を構成する質問に対する回答を Yes/No の二値だけになるように限定したものを Yes/No チャートと呼ぶ[61]．本研究では，この Yes/No チャートに着目する．

分類の基本活動は，ある基準(条件)を設定し，それを満たすか満たさないかを判定すること，つまり二分することである．三分以上の分類は，二分に関しての同類の概念化と縮退を行った結果であって，分類の活動としては高度なものである，と考える．よって，Yes/No チャートで条件に当てはまる，もしくは当てはまらない，の 2 つに分類することを繰り返すことで，分類の基本形を習得することが期待できる．

Computational Thinking を教科学習の中に取り込むために，学習内容の抽象化による分類と，その結果に基づくアルゴリズム的思考としての手順化，そして，プログラミングを通じた実行による評価を，ベン図，Yes/No チャート，プログラミングによって実現することを目指す．ベン図は学習対象の分類の宣言的な記述，Yes/No チャートはその手続きとしての記述，そしてプログラミングやプログラミング可能な対話型ロボットとのやりとりは，その手続きの実行および評価のためのツールとして用いる．

ここで，ベン図，Yes/No チャートに対し，学習者に利用してよい必要十分な部品を提供しているため，分類課題は答えが決定する課題である．しかし，ベン図，Yes/No チャートの解は多様であり，一意な解答があるわけではない．よって，学習者それぞれがそれぞれの解を作成することとなる．このことから，学習者それぞれがベン図，Yes/No チャートを作成することを通して，その間違いから学び取り，正解に近づいていくような学習を提供することが必要であると考えられる．このことによって，「学習者の考えに従うことで，学習者が誤りだと判断することのできる解答を導く」という Error-Based Simulation[79]を実現することが期待できる．正解に属さないチャートは間違いであるが，学習自身の作成したチャートに従うと，不適切な分類となる例を見つけることが誤りの診断の一つとなり，その結果を用いることで，学習者の作成した誤ったチャートと最も近い正解チャートを検出し，そのチャ

ートへの変更の方法を見つけることで，チャートの修正を誘導することができると考えられる．この実現のために，学習者に正解か不正解であるかについてフィードバックを与えるシステムを開発すること，そして経験学習に基づく枠組みの中で，システムを用いて Computational Thinking を育成することを目指す．別の観点から見れば，ベン図の解および Yes/No チャートの解が多様であることから，指導者が適切に学習者個々に指導するのは困難であることが想定される．よって，学習者自身が個別に自身の解から学ぶことは，指導者にとっても有効である．

経験学習とは，自分が実際に経験した事柄から学びを得る学習である．Kolb によれば，単に経験するだけでなく経験を次に生かすためのプロセスである経験学習モデルを提唱した．そのプロセスは，「具体的経験」，「省察(内省)」，「概念化」，「試行(実践)」の 4 段階の学習サイクルで構成されており，人材育成などを行う現場では特に着目されている重要な学習である[70]．従来から，従弟制など経験から学ぶ学習の有効性が示されており，認知的徒弟制の研究も進められるなど[80]，多様な観点から経験学習の研究が進められている．ここで，システムを有効に活用するために，経験学習を取り入れた実践を行う．

経験学習を取り入れた実践では，全体としてどのように説明文を記載するかについての方向性，つまり他者が読んで再現できるように記述する旨を示し，ベン図，Yes/No チャート学習システムにより学習(経験)させる．そしてベン図，Yes/No チャートの構成を理解し，適切に分類を行えるようになれば，Computational Thinking が高まったと捉えることができる．そしてベン図，Yes/No チャート学習システムは多くの学習者に対し，正解・不正解の即時的フィードバックを返すことから，学習者が個別に課題に取り組む状況が実現でき，その機能を活用することで経験学習を取り入れた実践が可能となる．

このモデルで目指す学習目標をブルームタキソノミー[76][77]で位置付けると，「概念的知識」に対する「理解」「応用」「評価」の段階に対応すると想定される．ベン図は「理解」を深めるため，Yes/No チャートは「応用」を深めるためであり，プログラミング可能な対話型ロボットは「理解」と「応用」ができているかを「評価」するためのツールとなる．ブルームのタキソノミー改訂版では「応用」と「評価」の間に「分析」があり，学習内容を構成する概念やそれらの間の関係を整理することに相当するが，この部分に相当することは指導者が事前に準備する．これは，「分析」の結果として指導者が学習者に得て欲しいと考えている学習内容の構成要素を提示して「理解」「応用」のための部品として用意することで，その多様な結果を学習者自身でも比較して「評価」しやすくするためである．本来であれば「分析」も学習者が行う必要があるが，概念化は難しい活動であり，「理解」「応用」「評価」を実施するための足場掛けとして，このモデルでは「分析」の部分は学習者のタスクから外している．

なお，ベン図，Yes/No チャートを利用したプログラミング教育の実践例はいくつか報告されているが[78]，ベン図，Yes/No チャートを単なるツールではなく，学習者自身で「理解」「応用」「評価」を連携させるための仕組みとして利用することを目指す．ここで，

Yes/No チャートは，一般的なプログラミング言語(例えば C 言語)の if-else を用いた学習と同等の思考プロセスから作成すると想定される．しかし，プログラミング言語において，コンピュータ固有の記述が多く含まれることになり，分類課題に対する理解のために必ずしも適しているとは言えない．また，プログラミングを用いた業務や学術活動などに携わることがない学習者に課すことは，困難であると想定される．さらに，分類を繰り返すことで複雑な入れ子構造となってしまい，可読性が著しく低下することが初学者の理解を妨げてしまうことが想定される．

このことから，本研究では，分類課題の解決法を Yes/No チャート[61]を用いて記述することで Computational Thinking の育成を目指す．

4.3. 図形の分類課題の Computational Thinking 課題化

教科内で Computational Thinking およびプログラミング的思考を育成するために，図形の分類課題を用いる．そして図形の分類課題を Computational Thinking の課題とするために，ベン図と Yes/No チャートを活用する．以下，ベン図と Yes/No チャートについて述べる．

4.3.1. ベン図

ベン図は閉曲線によって複数の集合の関係を表すものであり，あるものがどの集合に属するかを一つの閉曲線または複数の閉曲線の交わりによる領域で表す．ベン図は，ある条件 A，B が与えられたとき，A，B，A∩B，もしくは¬(A∪B)の四つのうちいずれか一箇所に図形を配置することができ，これにより与えられた条件に対する関係性を整理・理解することができる．また，ベン図を用いた学習では，条件や図形の数，種類を調整することにより学習者に多様な課題を提供でき，学習者のレベルによって難易度を調整することが可能である．ベン図の作成は，学習対象それぞれの特徴を明確にし，差別化，弁別ができるようにするためであり，「概念的知識」の「理解」に対応する．

例えば，分類対象となる図形に，円，正三角形，直角三角形，直角二等辺三角形，不等辺三角形，正方形，長方形，ひし形，不等辺四角形の 9 種類を準備し分類を行うとすると，条件をどのように設定するかでその分類方法は異なる．分類の一例を図 1 に示す．この例では，条件 A に「直角がある」，条件 B に「辺が三つ」を設定した上で分類を行っている．ここで条件 A を「辺が四つ」，条件 B を「辺が三つ」に変更すれば分類方法は異なる．他にも様々な条件の組み合わせがあり，多様な解が存在する．

ベン図を用いた図形の分類課題により，Computational Thinking の概念である分解，パターン化(パターンの発見)，抽象化を学習できると考えられ，関係性理解を深めることができる．しかし，関係性理解をするだけでは，図形の分類課題を通した Computational Thinking の中で，手続化を学習することができない．図形の分類課題を通して，Computational Thinking の手続化を行うために，Yes/No チャートを用いる．次の節では，Yes/No チャートについて

述べる．

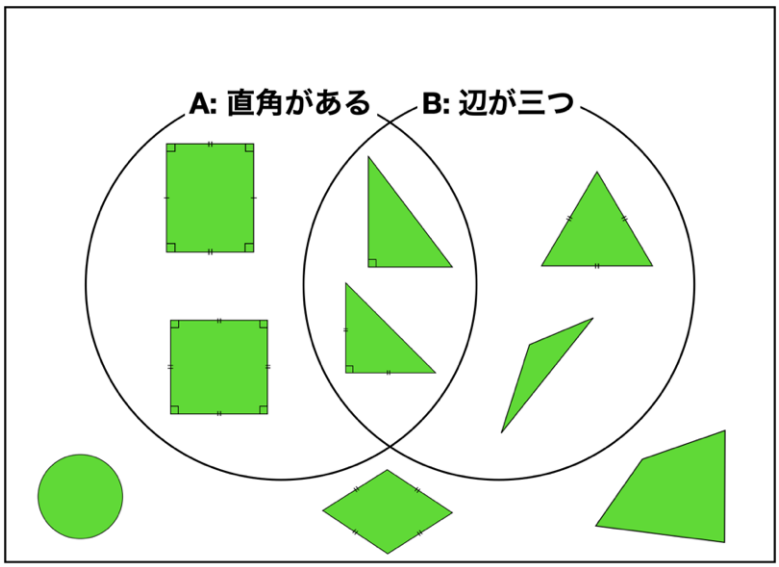

図 1 ベン図を用いた図形の分類

4.3.2. Yes/No チャート

前述したように，ベン図を用いた学習は，関係的理解を深めることが期待できるが，手続きの理解ができているかについては明らかではない．関係的理解をコンピュータに実行させる形にするために，手続化するプロセスが必要となる．そのとき，与えられた複数の図形に対し，複数の条件を用いて一意に分類するにはどうすればよいかという課題を行うために，Yes/No チャートを用いた分類を行う．Yes/No チャートとは与えられた条件に対し Yes/No の判定を繰り返すことで一意に分類できるチャートのことである．図 1 で用いた九つの図形を Yes/No チャートを用いて分類すると，例えば図 2 のようになる．図 2 から，正方形は，「辺が四つ」，「すべての辺の長さが等しい」，「直角がある」の三つの条件を満たしており，その条件を辿ることで図形を一意に特定することができる．他の八つの図形についても Yes/No チャートを用いて同様に一意に特定することができる．

図 2 はあくまで一例であり，解には多数のパターンが存在する．また，分岐する条件の個数および図形の個数を増減すること，条件を違ったものにすること，そして条件を自身で考えさせることなどによって，難易度の調整を行うことができる．ここでの Yes/No チャートは末端以外が条件分岐となっているフローチャートとする．ベン図を Yes/No チャートとして変換することで，同定したいものがどの種類であるかを調べる手続きを可視化することを通して明確にすることができる．なお Yes/No チャートは，ベン図で整理した分類を手続化して利用できるようにするために用いる．これが「概念的知識」を利用するという意味で「応用」に対応していると想定される．

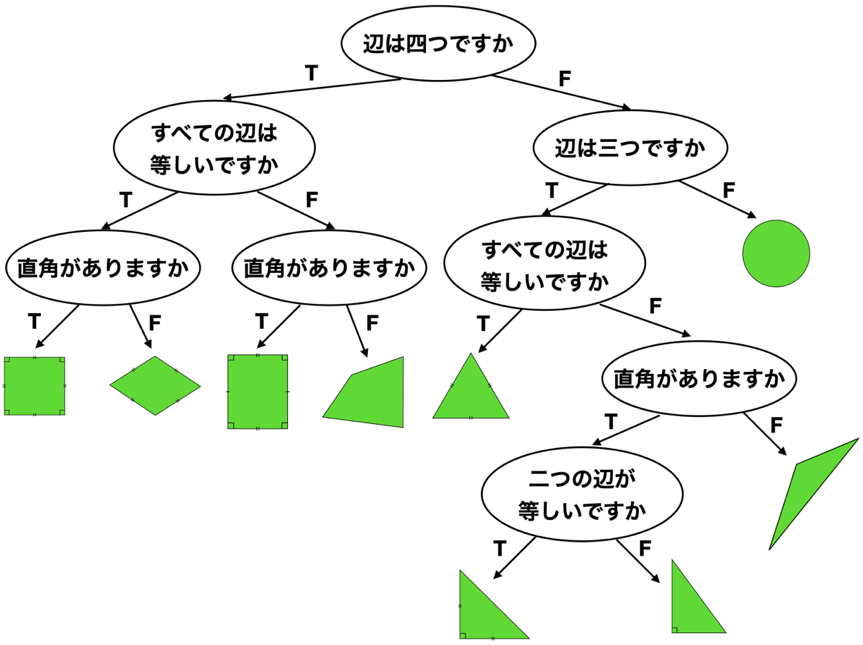

図 2 Yes/No チャートを用いた分類の手続化

Yes/No チャートを用いた図形の分類課題により，Computational Thinking の概念である分解, パターン化(パターンの発見), 抽象化を学習できると考えられる. ここで, ベン図, Yes/No チャートを用いた図形の分類課題と Computational Thinking との関連性について表 14 に示す. しかし，これらの要素はそれぞれ関連性を有していることが想定され，完全に峻別することは困難である. よって，Computational Thinking が全体として高まったことを評価する方向で研究を進める.

表 14 図形の分類課題と Computational Thinking との関連性

項目	内容
分解	扱う対象を分類できる属性 **(**性質**)** を見つけること **(e.g., "**直角がある**")** 個々の **Yes/No** にすること
パターン化	属性の組み合わせで個々の対象を特徴付けられること **Yes/No** の判定基準を見つけること
抽象化	個々の対象を属性の組み合わせで表現すること **(e.g., "**正方形は直角があり，辺が **3**つではない**")**
手続化	属性チェックの繰り返しで弁別できる具体的な手続き **(Yes/No** チャート，もしくはそれに類するもの**)** を作成すること

ここで，ベン図の解が多様であれば，手続化の方法も同様に多様である．つまり，ベン図と Yes/No チャートの多様性は直接的な関連性を有することから，ベン図の難易度に合わせて Yes/No チャートの難易度も調整可能である．このことから，学習段階に合わせた内容を用いて個別に Computational Thinking を育成することが期待できる.

以上に示した課題は，どの教科においても適用可能なプログラミング的思考の育成教材としての活用が期待できる．分類課題自体はどのような教科においても存在している．また，Yes/No チャートを作成するためには，分類対象に対しての，「パターン抽出(一般化)」と「抽象化」が必要となる．さらに，それをチャート化することは，「手続化」になっている．また，分類をチャートに従って段階的に行うことは，分類課題の「分解」を行っていることになる．よって，教科の内容と目的に沿った Computational Thinking の育成教材と位置付けることが可能となる．

4.3.3. プログラミング

プログラミングは Yes/No チャートで手続きとして記述した内容が妥当かどうかを「評価」するために行う．そして，プログラミングには，初学者の Computational Thinking の育成も視野に入れ，ブロック型プログラミングを用いる．

ブロック型プログラミングとは，Scratch のように，命令のあるブロックを適切に配置してプログラミングを行うシステムのことであり，学校現場でもよく使われていることや，実践事例も多く報告されていることから，ブロック型プログラミングは抵抗感なく使用することが期待できる．

本研究では，このプログラミングに対し，2 つの方向で開発を進めている．1 点目は，ベン図，Yes/No チャートとして整理した教科の学習内容を，プログラミングを実際に行い，TA としてのコミュニケーションロボットに教え，実行させることによって，学習者が自身の理解を振り返ることを目標としたものである．ここでのプログラミングは，実際にコーディングを行うこと，および学習内容をプログラミング的思考や Computational Thinking で整理した結果をロボットに教え，実行させるための手段となっている．

2 点目は，Yes/No チャートと同一の条件や分類対象を準備したブロック型プログラミング学習システムを用いて実際に正解/不正解のフィードバックを返すことを目標としたものである．このことによって，Yes/No チャートとブロック型プログラミングの学習効果の比較やベン図作成，Yes/No チャート作成，ブロック型プログラミングのどこに間違いがあるのか，躓き点があるのかについて明示化することが期待できる．これは，ベン図をクラス図やオブジェクト図などの静的な関係，Yes/No チャートをアクティビティ図などの動的な関係，そしてブロック型プログラミングを実際にコーディングするプロセスと捉え，その中でどこに間違いがあるかについて把握することを目指したものである．

さらに，これらのシステムによる学習を深化させるために，相互評価機能，同時編集機能の実装を行っている．また，ブロック型プログラミングで HTML と JavaScript からなる Web ページを作成できるブロック型プログラミング学習システムを開発している．

今後，上述したシステムの学習効果を検討することが必要である．

4.4. Computational Thinking を育成するベン図，Yes/No チャート学習システムの設計・開発

以下，Computational Thinking を育成する学習支援システムの開発について述べる．まず，Computational Thinking を育成するための課題として図形の分類課題を取り上げ，その分類課題の Computational Thinking 課題化について述べる．そしてその課題を実装した学習支援システムについて述べる．

本研究では，学習者が自身で作成したベン図，Yes/No チャートの間違いから学習し，正解に至るという学習を行う．その実現のために経験学習の枠組みを用い，Computational Thinking 育成を目指す．

これらを踏まえ，学習者それぞれの Computational Thinking を育成するために，ベン図学習システムおよび Yes/No チャート学習システムを開発した．そして，Yes/No チャートで作成した結果を評価するブロック型プログラミングとして，ロブリックを用いた環境を開発した．さらに，Yes/No チャート学習システムの学習効果を検討するために，Yes/No チャート学習システムと同様の課題を実装したブロック型プログラミングを開発した．

これらのシステムは，学習者が自身で条件ブロックや図形を動かすことによって，ベン図や Yes/No チャートの作成，およびプログラミングのコードを組み立てることができる．そして，ベン図，Yes/No チャート，ブロック型プログラミングについては，学習者ごとに正解・不正解の即時的なフィードバックを返す機能を実装した．このことにより，指導者の負担が減るだけでなく，学習者それぞれが主体的に学習を行うことが期待できるとともに，学習者それぞれの Computational Thinking を育成することが期待できる．

これらの学習システムでは，バックエンドシステムとして Google Firebase，データベースは Firebase Realtime Database を用いた．いずれの学習システムも Web アプリケーションとして開発を行っており，インターネット接続があればソフトウェアをインストールすることなしに扱うことができる．Web アプリケーションの開発には，HTML5，JavaScript，ライブラリとして vis.js，Blockly，node.js を用い，Firefox および Google Chrome で動作確認を行っている．以下，開発したベン図，Yes/No チャート学習システム，ブロック型プログラミングシステムについて述べる．

4.4.1. ベン図学習システム

図 3 に，開発したベン図学習システムの画面を示す．これは，学習者が左側にある図形の画像を，ベン図に見立てた二つの四角が表示された領域に配置するというシステムである．二つの四角の内側は，それぞれの条件が成り立っていることを意味する．左側のペインにある「Condition」タブを選択すると，「頂点は三つか」や「直角があるか」など，複数の条件が選択できる．学習者は，この中から任意の条件を選び四角の辺にドラッグアンドドロップすることで，条件を割り当てる．そして，その条件に当てはまる図形を「Shape」タブから

選択し配置する．全ての図形を配置した上で右上にある判定ボタンをクリックすると，システムは学習者が設定した二つの条件と図形が正しく配置できているかを判定し，正誤を表示する．ここでどこに間違いがあるのかは表示しない．判定ボタンのクリック回数と操作したオブジェクト，操作にかかった時間は全てロギングし分析できるようになっている．

このシステムでは，従来の授業で行われている，あらかじめ条件が配置されており，それを満たす図形を配置するのではなく，自身で図形の特徴を見出し，その上で条件も自身で選択する活動を行うことができる．このことで，従来行われている図形の分類を Computational Thinking の課題として捉え直している．

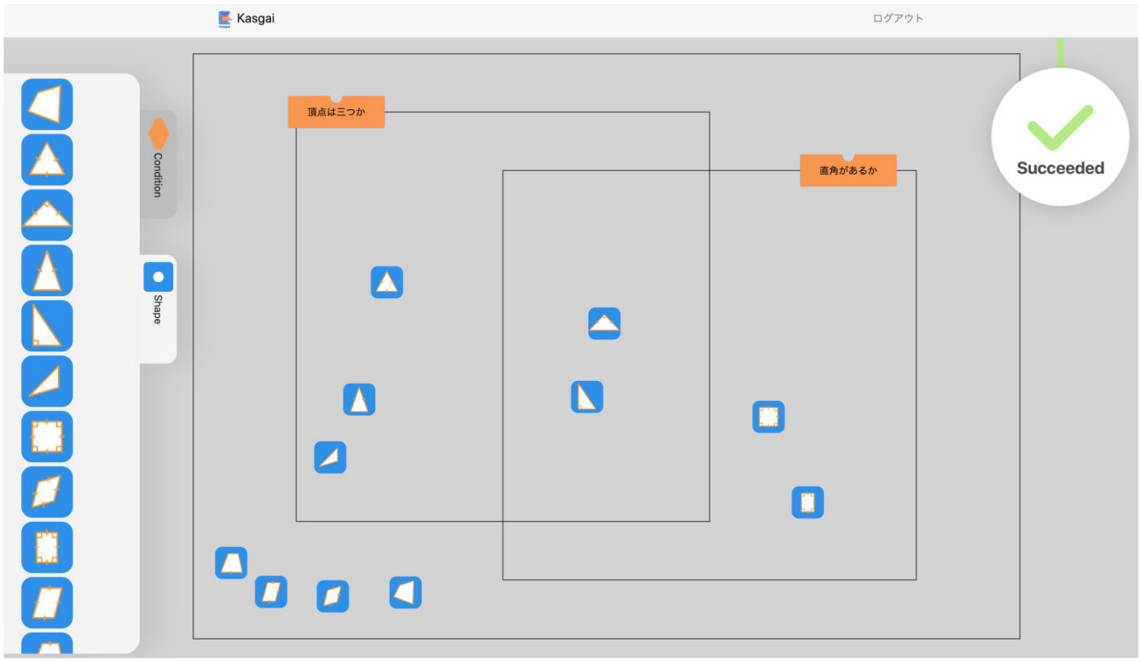

図 3 ベン図学習システム

4.4.2. Yes/No チャート学習システム

図 4 に開発した Yes/No チャート学習システムの画面を示す．これは，分類課題における Yes/No チャートの作成を支援するシステムである．そして，ベン図学習システムと同様に，学習者が様々なパターンの解を作成することができるように，条件および図形をマウスでドラッグアンドドロップできるようにし，学習者に正解もしくは不正解を返す自動正誤判定機能を実装した．

ここで操作方法について説明を行う．左側のペインにある「Condition」のタブをクリックすると，「頂点は三つか」や「直角があるか」などの条件ブロックが表示され，それらの条件ブロックをクリックすると，中央の領域に条件ブロックが表示される．そして，スタート地点を示す「Entry Point」をクリックすると接続ボタンが表示される．接続ボタンを選択後，接続したい条件をクリックすると接続される．「Shape」のタブをクリックすると，図形が表

示され，それらの図形をクリックすると中央の領域に図形が表示される．次に，条件ブロックの下に別の条件ブロックや図形を接続する場合，上の階層となる条件ブロックや図形をクリックすると，「Yes」として接続するか，「No」として接続するかの選択肢が表示される．そこで Yes/No のいずれかを選択後，接続したい条件ブロックや図形をクリックすると接続される．間違って接続したときは，その条件ブロックをクリックし，解除を選択することで接続を解除できる．全ての図形を配置し終えたら，右上にある判定ボタンをクリックすることで，正誤判定を行うことができる．ベン図学習システムと同様に，Yes/No チャート学習システムでは，学習者が作成した Yes/No チャートの条件と図形が正しく配置できているかを判定する．判定は，ベン図と同様に正誤のみを表示し，どこに間違いがあるのかは表示しない．そして判定ボタンのクリック回数と操作したオブジェクト，操作にかかった時間は全てロギングし分析できるようになっている．

ここで，図形を全て配置していない場合や冗長な表現があったときは，不正解と表示されるように設計を行っている．冗長な表現とは，同じブロックを連続して接続することや，強い条件 X の後に弱い条件 Y を接続するような絞り込みに意味をなさない接続方法のことを指す(例えば X,Y が X⊂Y を満たしている状態)．

以上のベン図，Yes/No チャート学習システムを用いることで，それぞれの学習者自身が配置した条件に沿って図形を分類することができ，指導者は操作方法を指導する以外の介入を行わずとも，学習者は図形の分類課題に取り組むことができる．

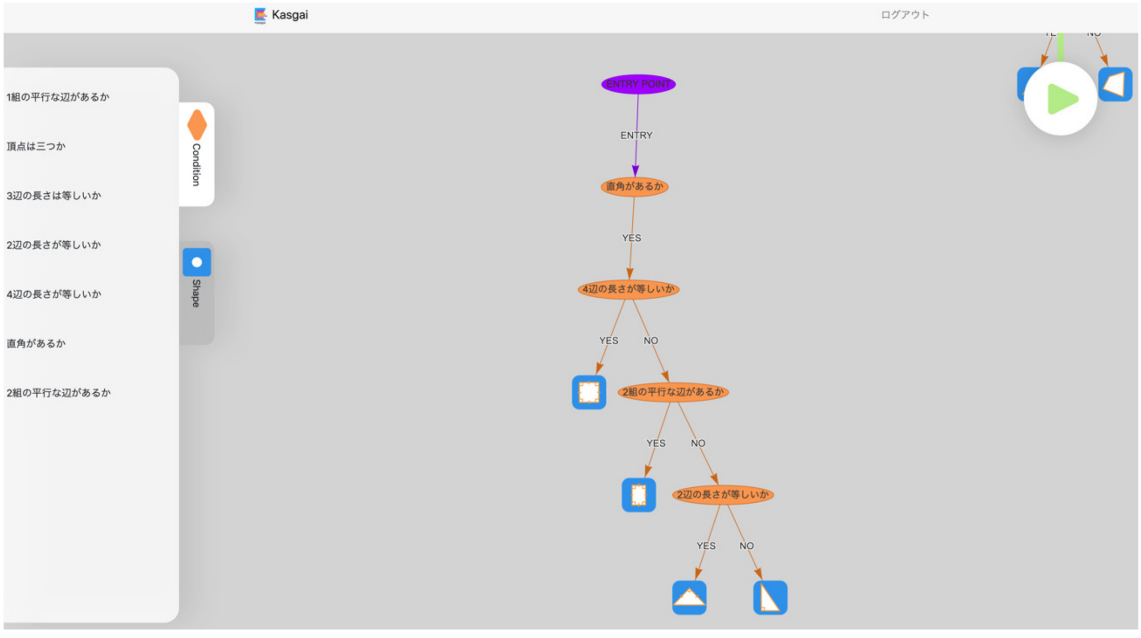

図 4 Yes/No チャート学習システム

4.4.3. ブロック型プログラミング学習システム

ここでは，試作したブロック型プログラミング学習システムについて示す．

(1) ロブリックを用いたブロック型プログラミング学習システム

本研究では，シャープのロボホンを用いる．ロボホンとはユーザと対話が可能なスマートロボットであり，ブラウザベースのプログラミング環境「ロブリック」を用いることでロボホンを操作することができる．これを用いて，学習者は Yes/No チャートの内容をプログラミングして実行すると，実際にロボホンを動かすことで自身の Yes/No チャートが正しいかどうかを評価することができる．

ロブリックにはロボットを動作させる様々な命令が用意されているが，Yes/No チャートの内容をプログラミングするにあたり，図 5 に示す質問ブロックという特殊なブロックを用意した．このブロックは「質問」「はい」「いいえ」という 3 つのパートから構成されており，「質問」パートは質問としてしゃべる内容，「はい」「いいえ」パートはロボットが「はい」「いいえ」という音声を聞いた時に実行する内容のブロックがはめ込まれる．2 つ以上の質問を含む Yes/No チャートは質問ブロックをネスト化することによって作成することができる．

図 5 質問ブロック

質問ブロックは基本的なブロックとして用意されている if-then ブロックを使って同等の機能を持つものを作成することもできるが，ここでは Yes/No チャートの内容を対話型ロボットで実行可能な形式にすることに思考の焦点を当てるために，一つのブロックとして用意している．よって，プログラミング学習の比重を大きくする場合には，このブロックも if-then，if-else ブロックのネストで作成することを学習活動に組み込むことが想定される．

このシステムを用いた試行的実践は実施している．今後，本システムを用いた実践を行い，その学習効果を検討する予定である．

(2) Blockly を用いた相互評価を可能とするブロック型プログラミング学習システム

筆者らは，今後の研究の発展性を検討するために，ブロック型プログラミング学習の中で相互評価を可能とするシステムを試作している．本システムは，Block Assembly System，Browsing System，Server の 3 つのプログラムから構成されており，前者の 2 つは Google が

提供するライブラリである Blockly を用いた．Server の開発には Python を用いた．Block Assembly System を図 6 に示す．この図 6 では，例として JavaScript を動かしている．Block Assembly System は，左側の領域にブロックを組み上げる部分，その下にコードを表示する Display ボタン，プログラムを実行する Run ボタンが配置されている．そして，右側の領域に，課題となる問題，他ユーザからのフィードバック，レーティングが表示されている．次に，Browsing System を図 7 に示す．Browsing System では，Block Assembly System の組み上げたブロックを閲覧することができ，それに対してコメントやフィードバック，レーティングを行うことができる．それぞれのシステムの関係について図 3 に示す．つまり，クライアント A が Block Assembly System で組み上げたブロックを，他のクライアント B,C,...は Browsing System の画面で閲覧することができる．そして，クライアント B,C,...は，クライアント A が組み上げたブロックを見て，コメントや評価，レーティングなどが行うことができる．このシステムを用いることで，従前のプログラミングの学習をさらに深めることが期待できる．

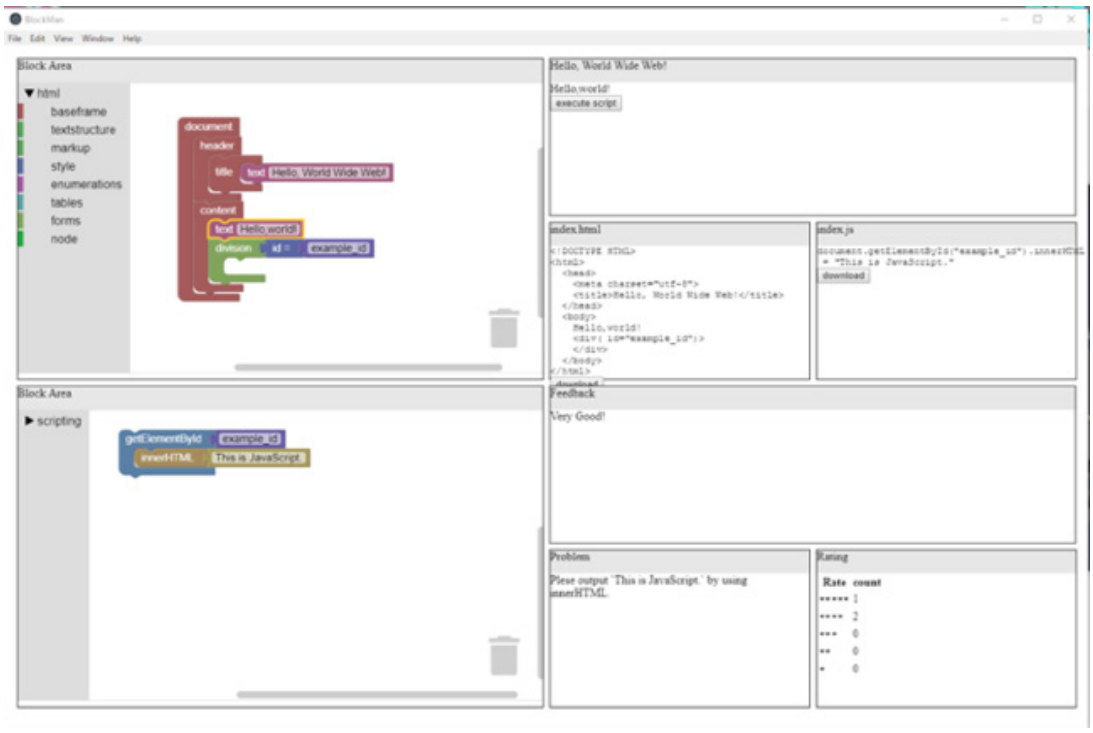

図 6 ブロック組み立てシステム

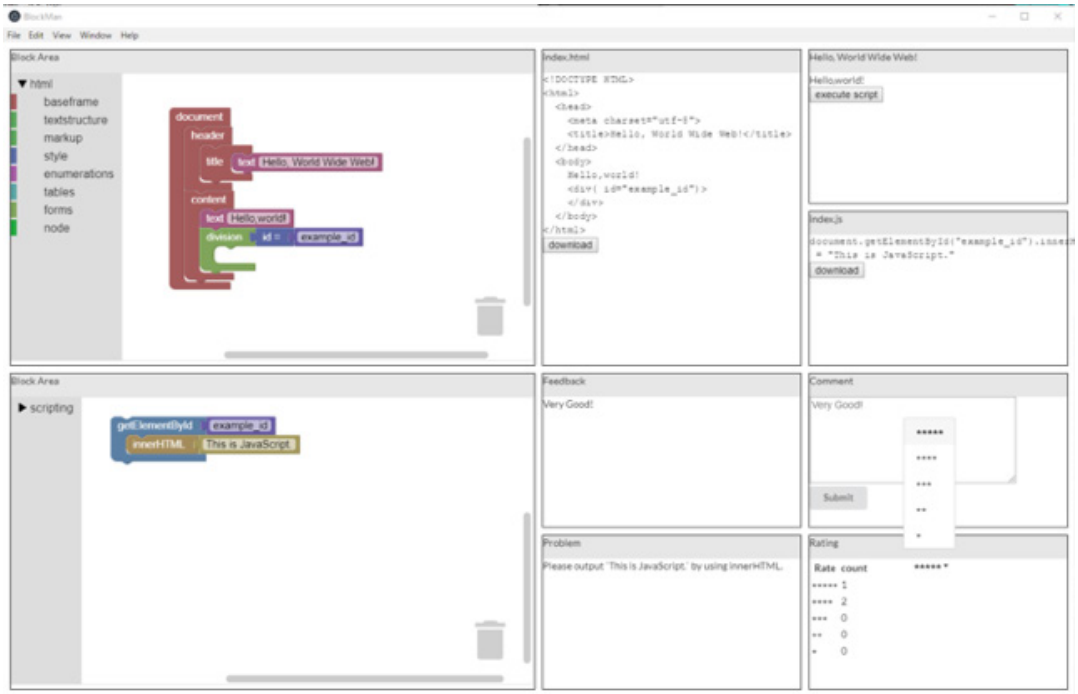

図 7 閲覧および評価システム

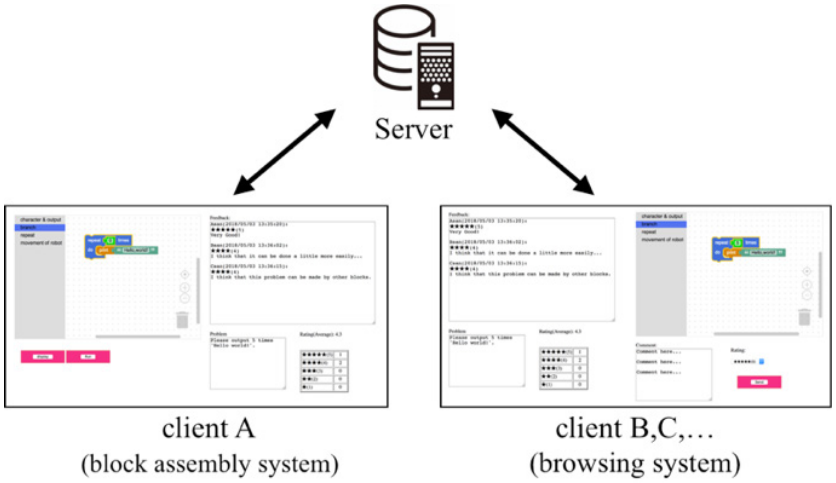

図 8 システムの構成

(3) 複数人が同時に編集できるブロック型プログラミング学習システム

ブロックによって表現されたコードを複数人が同時編集することができるブロック型プログラミング学習システムは，ブロックプログラミング環境において，協働学習を可能とする．本システムは，学習者が同じプロジェクトにログインすることによって，同じブロック・プレビュー，ソースコードが表示される．そして，ある学習者がブロックを操作すると，その結果は即時に反映され，参加している学習者全員の画面が更新される．そして，右上のボタンで編集制限モードを切り替えることができる．このモードを活用することで，教員が生

徒に操作させないようにすることなどが可能である．

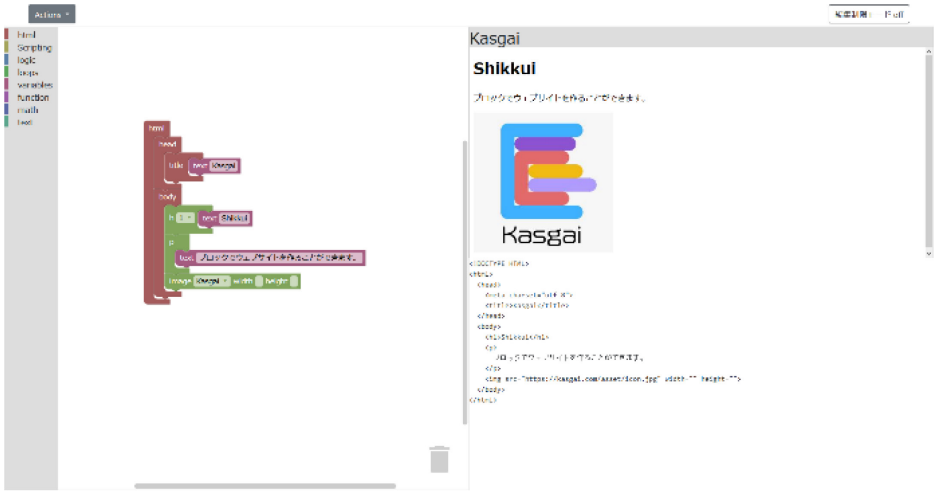

図 9 同時編集機能を実装したブロック型プログラミング学習システム

(4) Yes/No チャートと同一課題を実装したブロック型プログラミング学習システム

Yes/No チャート学習システムとブロック型プログラミングの学習効果を比較するためには，同じ題材・条件文を準備した上で行う必要があるが，既存のブロック型プログラミングシステムに，そのような条件を満たすものは存在しない．ここで本研究では，Yes/No チャート学習システムの学習効果を比較するために，Yes/No チャート学習システムと同一の図形や条件文を実装したビジュアルプログラミング学習システムを試作している．

図 10 に試作したビジュアルプログラミング学習システムの画面を示す．上述した Yes/No チャート学習システムの学習効果を測定するために，Yes/No チャート学習システムと同様の条件と図を採用している．また，学習者がマウスなどで条件や図形をドラッグアンドドロップすることで様々なパターンを作成し，学習者に正解・不正解について表示する自動誤判定機能を実装した．

ここで，操作方法について説明を行う．左側の「ベース」をクリックすると，そこに条件ブロックと図形ブロックが表示される．それらのブロックをエリアの中央にドラッグアンドドロップして配置する．条件ブロックと図形ブロックは Scratch と同じように扱うことができ，条件ブロックの Yes と No が当てはまる場所に条件ブロックと図形ブロックを入れ子にすることができる．ブロックを削除・複製したい場合は，ブロックを右クリックして削除・複製を選ぶことで指定の操作を行うことができる．条件ブロック内の条件文をクリックすることで条件を選択することができる．同様に，図ブロック内の図 n(ここで n は数字)をクリックすることで，図番号を選択することができる．

このシステムは，Yes/No チャート学習システムと同じ図と条件文を準備している．図には番号が振られており，対応する図のリストが学習者にはあらかじめ配布されている．

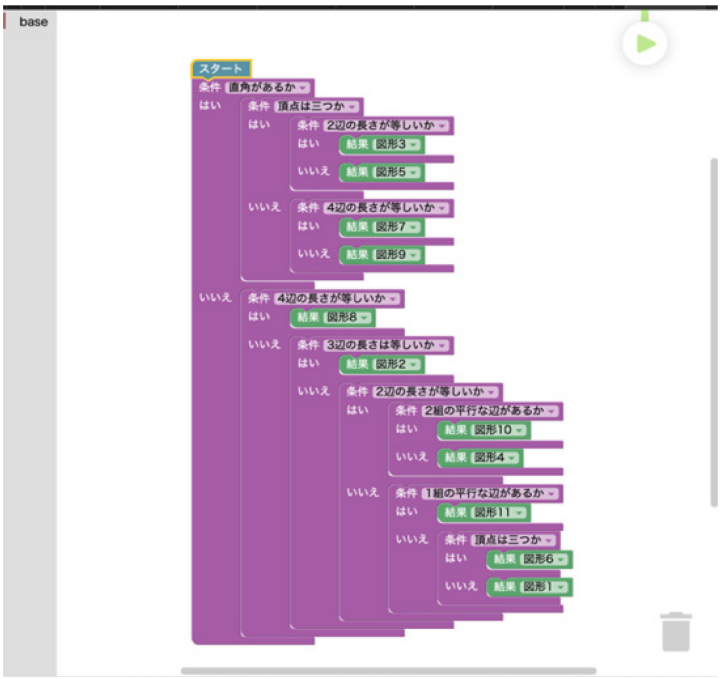

図 10 試作したブロックプログラミング学習システム

4.5. Computational Thinking を育成する実践モデルの開発

本節では，開発したベン図，Yes/No チャート学習システムの効果を測定するために，経験学習を取り入れた実践モデルを開発する．

4.5.1. 実践モデルの開発

この実践モデルは，三つのフェーズから構成されている．なお，フェーズ I と III は経験学習の範囲外と考える．実践モデルのフローを表 15，設問内容を表 16 に示す．まず，フェーズ I で図形の分類課題(ベン図，Yes/No チャート)を行い，その後ベン図，Yes/No チャート学習システムを用いて図形の分類課題に取り組み Computational Thinking を経験する(フェーズ II)．その後，図形の分類課題(ベン図，Yes/No チャート)を再度解く(フェーズ III)．この活動によって Computational Thinking が育成できれば，ベン図，Yes/No チャート学習システムの活用(フェーズ II)が Computational Thinking 育成に有効であると考えられる．

表 15 実践のフロー

フェーズ		内容
フェーズ I (事前)	課題1	ベン図を用いた図形の分類課題を解く
	課題2	ベン図を用いた図形の分類課題を解く対象をいくつかの条件を用いて，誰が見てもわかるように一意に分類する
フェーズ II (経験学習・システムの活用)	課題3	ベン図を用いた図形の分類課題を解く
	課題4	Yes/No チャートを用い，対象をいくつかの条件を用いて，一意に分類する
フェーズ III (事後)	課題5	ベン図を用いた図形の分類課題を解く
	課題6	ベン図を用いた図形の分類課題を解く対象をいくつかの条件を用いて，誰が見てもわかるように一意に分類する

表 16 設問内容

課題1・課題5	リストにある図形を以下のベン図に配置してグループ分けしたあと，条件 A，B を入れて下さい．
課題2・課題6	リストにある図形をいくつかの条件を用いて，完全に分類できる (他の人が見てもわかる) ように書いて下さい．

以下，実践モデルについて具体的に述べる．

フェーズ I では，(1)ベン図を用いた図形の分類課題を解かせる(以下，課題 1)．そして，(2)対象となる図形をいくつかの条件を用いて，誰が見てもわかるように一意に分類させる(以下，課題 2)．このとき，論理的に説明を記述させること，そして，その表現を他者が見ることで実行・理解・再現可能であるかを判断する．なお，小学生を対象とした実践においてベン図が最初から用いられており[35]，今回対象となる高校生および大学生はベン図について学習済みであることから，ベン図による分類を行うことができると想定される．よって，ベン図については条件を学習者に考えさせ，その条件に基づいて図形の分類を行わせる．

フェーズ II では，(3)ベン図学習システム(以下，課題 3)，(4)Yes/No チャート学習システムを用いて対象をいくつかの条件を用いて分類させる(以下，課題 4)．ここで，論理的に考えること，Yes/No チャートを用いた手続化は，他者が見ることで実行・理解・再現可能な表現であることに気づかせる．

フェーズ III では，(5)ベン図を用いた図形の分類課題(フェーズ I の(1)と同様の内容)を解かせる(以下，課題 5)．そして，(6)対象をいくつかの条件を用いて，誰が見てもわかるように分類させる(フェーズ I の(2)と同様の内容)課題を解かせる(以下，課題 6)．ここで，フェーズ I と III を比較し，図形の分類に関する説明および内容がどのように変容したかについて評価を行う．ここで，フェーズ I,III で扱う図形(図 11)は，フェーズ II のシステムが提供するものとは異なっている．また，図形から得られる特徴について自身で考えさせる．つま

り Computational Thinking の分解などを育成するため，図形の名称や特徴を文章で示していない．

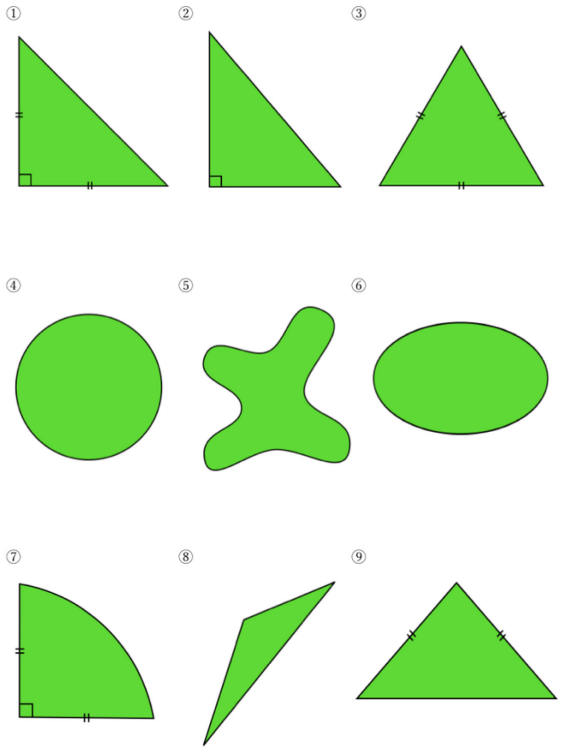

図 11 図形のリスト

次に，経験学習と実践モデルとの対応付けに関して表 17 に示す．表 17 より，全体の活動を通して Computational Thinking の分解，パターン化，抽象化，手続化を理解させるような経験をさせ，それを振り返り，他でも使えるような学習を提供している．そしてフェーズ II で学んだ内容を踏まえフェーズ III がフェーズ I の説明より具体的，明確になれば，フェーズ II の経験が転移していると想定される．

表 17 経験学習と実践モデルとの対応

項目	内容
具体的経験 (具体的な経験をする)	ベン図と Yes/No チャートの具体的な経験をさせる
内省 (経験を様々な観点から振り返る)	個別にフィードバックを返す機能により，学習者が自身で振り返りができる
概念化 (他でも応用できる概念を構築)	転移が認められるようフェーズ I,III と違った図形・条件とし，他でも用いることができるような概念化を提供する
試行 (新しい場面で実際に試す)	他でも用いることができるような学習を提供し，その内容を実際に活用させる

ここで，ベン図，Yes/No チャート学習システムと今回の実践モデルを関連付けると，システムで実際に分類させることで具体的経験を提供している．そして明示的に可視化して手続化が行えること，さらに正解・不正解のフィードバックを与えることから，頭の中で考えて分類を行うよりも，内省の一部を助ける効果があると考えられる．また，システムが提供した図形と紙ベースでの実践で扱う図形が異なることから，システムでの経験が別の図形の問題を解く際の概念化を提供していると考えられる．なお本研究では別課題を扱わないが，フェーズ II のシステムを用いた学習と，フェーズ I,III の実践では違った図形を扱っており，学習の近転移が生じることを想定している．

これらの実践モデルおよびシステムを活用することによって，Computational Thinking を育成することが期待できる．以下，これらのシステムを用いた 2 つの実験について述べる．

4.6. 実験 1:ベン図，Yes/No チャート学習システムの学習効果

ここでは，筆者らが開発した実践モデルを元に，ベン図，Yes/No チャート学習支援システム，およびブロックプログラミング学習システムを用いた実践を行う．そしてこれらのシステムの Computational Thinking を育成する効果について検証する．

本実験では，前章で開発した実践モデルを元に，ベン図，Yes/No チャート学習システムが Computational Thinking を育成する効果について検証する．

4.6.1. 実施日および実践対象者の状況

2019 年 6 月から 10 月に，関西圏の私立高校の生徒 4 名，関西圏の私立大学 4 校，関東圏の私立大学 1 校の学生 20 名の計 24 名を対象に実践を行った．

4.6.2. 実践計画

表 15 に示す実践フローに基づき，フェーズ I，II，III を順に実施した．フェーズ I，III では用紙を配布し，課題 1,2,5,6 について回答させた．フェーズ II では，ベン図，Yes/No チャート学習システムを用いて，図形の分類および分類に沿って図形を分類させた(課題 3,4)．なお，フェーズ II で用いた図形はフェーズ I，III で用いた図形と異なっている．そして実践はフェーズ I を 20 分，フェーズ II のベン図，Yes/No チャート学習システムを 30 分，そしてフェーズ III を 20 分で実施した．

フェーズ I，III の評価を行うにあたり，表 18 に示す評価基準を作成した．評価にあたっては，4 名が別々に実施し，その平均点を学習者の得点とした．ここで，手続化が身に付いていることが Computational Thinking 育成の観点で重要であることから，得点が 3.00 以上である場合，Computational Thinking が身に付いていると判定した．そして手続化は Yes/No チャートで身に付けることができることから，Yes/No チャート学習システムでの正解/不正解を評価した．なお，学習者の得点の標準偏差が 0.50 以下であることを条件とし，標準偏差

が 0.50 より大きいときは，評価者の中で対象となる学習者の得点に関して協議を行うこととした．

そして，今回の実験対象者は高校生，大学生であり，ベン図については学習済みであることから，ベン図条件文などを自身で考えさせることとした．

表 18 評価基準

得点	ベン図の理解	手続化の達成度 (Yes/No およびそれに類する分類方法の利用)
5	○	○
4	○	○ (ほとんど正解)
3	○	△ (3 段以上正解している部分がある)
2	○	△ (2 段正解しており, 手続化に気づいている)
1	×	×

4.6.3. 結果

(1) 得点

実践後，学習者の得点を算出したところ，全ての学習者の得点の標準偏差は 0.50 以下であったため，今回の採点結果をそのまま採用した．単純集計の結果について表 19 に示す．

表 19 の丸括弧内は標準偏差，Yes/No チャートという項目の列は，Yes/No チャート学習システムで正解したかどうかについて示している．回答にかかった時間については，本人ができたと申告して回答を終了した時間(以下，回答時間と表記する)であり，分数は四捨五入した値である．また，正解か不正解のいずれであったかは問うていない．ここで，Yes/No チャートで正解した学習者の群をシステム正解群，Yes/No チャートで不正解の学習者の群をシステム不正解群と定義する．

表 19 より，システム正解群は 12 名，システム不正解群は 12 名であった．そしてシステム正解群全員のフェーズ III における得点はいずれも 3.00 以上であった．そしてシステム不正解群では，フェーズ III における得点が 3.00 以上の学習者は 6 名，3.00 未満の学習者は 6 名であった．そして 7,8,14 以外の学習者は回答時間がフェーズ I と III と比較して短くなっていた．

表 19 学習者の得点，回答にかかった時間

学習者番号	得点		Yes/Noチャート	回答時間	
	フェーズ I	フェーズ III		フェーズ I	フェーズ III
1	4.50(0.50)	4.75(0.43)	○	8	5
2	3.75(0.43)	4.75(0.43)	○	7	4
3	3.00(0.00)	4.50(0.50)	○	8	7
4	2.50(0.00)	5.00(0.00)	○	10	5
5	3.00(0.00)	4.00(0.00)	○	9	6
6	1.75(0.43)	5.00(0.00)	○	13	8
7	5.00(0.00)	4.00(0.00)	○	12	20
8	5.00(0.00)	4.75(0.43)	○	9	20
9	4.75(0.43)	4.75(0.43)	○	9	5
10	5.00(0.00)	5.00(0.00)	○	7	3
11	5.00(0.00)	5.00(0.00)	○	11	4
12	5.00(0.00)	5.00(0.00)	○	10	6
13	1.50(0.50)	2.25(0.43)	×	15	12
14	1.75(0.43)	2.00(0.00)	×	16	17
15	1.50(0.50)	2.75(0.43)	×	12	10
16	4.00(0.00)	5.00(0.00)	×	13	8
17	2.00(0.00)	3.50(0.50)	×	10	9
18	2.00(0.00)	3.50(0.50)	×	8	8
19	3.25(0.43)	4.75(0.43)	×	11	6
20	2.75(0.43)	5.00(0.00)	×	9	7
21	2.00(0.00)	1.75(0.43)	×	20	14
22	2.50(0.50)	2.50(0.50)	×	16	13
23	2.00(0.00)	2.00(0.00)	×	19	14
24	5.00(0.00)	5.00(0.00)	×	12	8

注: ( ) 内は標準偏差

“Yes/No チャート” の項目は, Yes/No チャート学習システムでの正解/不正解を表す

次に，表 19 の得点が正規性を有するかについて，Shapiro-Wilk 検定で分析したところ，正規分布に従わなかった(フェーズ I: $W=0.86, p<.01$; フェーズ III: $W=0.79, p<.01$)．よって，以下得点をノンパラメトリックデータとして扱う．そして，フェーズ I，III における得点の中央値の差について，全体，システム正解群，システム不正解群それぞれに対し，Wilcoxon 符号付き順位検定で分析した．その結果を表 20 に示す．

表 20 より，全体の得点の中央値に 1%水準で有意な向上($T=12.50, p<.01$)，システム不正解群の得点の中央値に有意な向上 ($T=1.50, p<.05$) が認められた一方，システム正解群の得点の中央値には有意な向上が見られなかった ($T=5.50$, *n.s.*).

表 20 フェーズ I とフェーズ III における得点

		得点		検定結果
		フェーズI	フェーズIII	
全体	中央値	3.00	4.75	12.50**
	最大値	5.00	5.00	
	最小値	1.50	1.75	
システム正解群	中央値	4.63	4.75	5.50
	最大値	5.00	5.00	
	最小値	1.75	4.00	
システム不正解群	中央値	2.00	3.13	1.50*
	最大値	5.00	5.00	
	最小値	1.75	1.75	

*p <.05, **p <.01

次に，システム正解群，システム不正解群間でフェーズ I，III における得点の中央値に差があるかについて，Wilcoxon 順位和検定で分析した．その結果を表 21 に示す．

表 21 より，フェーズ I，III のいずれにおいても得点の中央値に有意な差が見られ，システム正解群の得点の中央値の方が，不正解者の得点の中央値よりも大きかった (フェーズ I:T =117.50, p < .01; フェーズ III: T =110.50, p < .05).

表 21 システム正解群とシステム不正解群の得点

		得点		検定結果
		システム正解群	システム不正解群	
フェーズI	中央値	4.63	2.00	117.50**
	最大値	5.00	5.00	
	最小値	1.75	1.50	
フェーズIII	中央値	4.75	3.13	110.50*
	最大値	5.00	5.00	
	最小値	4.00	1.75	

*p <.05, **p <.01

(2) 回答時間

フェーズ I およびフェーズ III の回答時間が正規性を有するかについて，Shapiro-Wilk 検定で分析したところ，正規分布に従わなかった (フェーズ I: W =0.91, p <.05; フェーズ III: W =0.89, p < .05).

次に，フェーズ I，III における回答時間の中央値の差について，全体，システム正解群，システム不正解群それぞれに対し，Wilcoxon 符号付き順位検定で分析した．その結果を表 22 に示す．

表 22 より，全体では，回答時間の中央値に 1%水準で有意に減少 (T=229.00, p < .01)，システム不正解群の回答時間の中央値に 1%水準で有意に減少 (T =64.50, p < .01) していることが認められた．一方，システム正解群の回答時間の中央値には有意な減少が認められなかった (T =55.00, *n.s.*).

表 22 フェーズ I とフェーズ III の回答時間

		回答時間		検定結果
		フェーズI	フェーズIII	
全体	中央値	10.50	8.00	229.00**
	最大値	20.00	20.00	
	最小値	7.00	3.00	
システム正解群	中央値	9.00	5.50	55.00
	最大値	13.00	20.00	
	最小値	7.00	3.00	
システム不正解群	中央値	12.50	9.50	64.50**
	最大値	20.00	17.00	
	最小値	8.00	6.00	

**p <.01

さらに，システム正解群，システム不正解群間でフェーズ I，III における回答時間の中央値に差があるかについて，Wilcoxon 順位和検定で分析した．その結果を表 23 に示す．

表 23 より，フェーズ I，III のいずれにおいても回答時間の中央値に有意な差が見られ，システム正解群の回答時間の中央値の方が，システム不正解群の回答時間の中央値よりも小さかった (フェーズ I: T =25.50, p < .01; フェーズ III: T =30.00, p < .05).

表 23 システム正解群とシステム不正解群の回答時間

		回答時間		検定結果
		システム正解群	システム不正解群	
フェーズI	中央値	9.00	12.50	25.50**
	最大値	13.00	20.00	
	最小値	7.00	8.00	
フェーズIII	中央値	5.50	9.50	30.00*
	最大値	20.00	17.00	
	最小値	3.00	6.00	

*p <.05, **p <.01

4.6.4. 考察

表 20 より，フェーズ III はフェーズ I と比較すると，得点の中央値は 1%水準で有意に向上していた．よって，本システムを活用することは，Computational Thinking 育成に有効であることが認められた．よって，システムの有用性を示すことができた．

ここで，システムの有用性について詳細に検討するために，学習者の得点およびシステムでの正解/不正解を元に，学習者を表 24 のようにグルーピングを行った．ここで，表 24 の表内の項目欄ではフェーズ I，フェーズ III の得点をそれぞれ I，III と略記している．以下，表 24 に示したそれぞれのグループについて考察を行う．

表 24 学習者のグルーピング

グループ	n	学習者番号	Yes/Noチャート	特徴
A	6	1~6	○	I < III
B	2	7,8	○	I > III
C	4	9~12	○	I = III
D	3	13~15	×	I < III, I < 2.00
E	5	16~20	×	I < III, I ≥ 2.00
F	1	21	×	I > III
G	2	22,23	×	I = III, I ≤ 2.50
H	1	24	×	I = III, I > 2.50

注:「Yes/No チャート」の項目は，Yes/No チャート学習システムでの正解/不正解を表す

(1) Yes/No チャート学習システムで正解した学習者

フェーズ II において Yes/No チャート学習システムで正解した学習者(システム正解群)は

12 名であった．そして，得点が上がった学習者 6 名，下がった学習者 2 名，維持した学習者 4 名であった．システム正解群を表 24 に示した 3 グループに分割した上で，それぞれのグループの特徴について述べる．

§1 グループ A

グループ A では，フェーズ I と III を比較すると，グループ A の得点の中央値は向上していた．そしてフェーズ II の経験により，フェーズ III ではフェーズ I より回答に対する記述が具体的になっていた．フェーズ I における分類条件は「きれいな円」,「きれいじゃない円」などと曖昧なものが見られたが，フェーズ II の経験後に行ったフェーズ III では「二辺が等しい」,「円の長径と短径の長さが異なる」など数学的に適切なものとなっていた．フェーズ II の経験がなければフェーズ III の分類条件を記述できなかったと想定され，フェーズ II の経験を通して Computational Thinking が向上した可能性が示唆された．よって，グループ A の学習者には，Yes/No チャート学習システムが効果的であることが示された．

§2 グループ B

グループ B では，フェーズ I と III を比較すると，グループ B の得点の中央値は減少していた．しかし，フェーズ I の段階ですでに課題 2 の分類ができていた．フェーズ III では，制限時間内に回答できず，得点が下がっていたが，「四分位円」や「閉曲線の一部に直線を持つ」など，数学的により難しい分類条件を用いて部分的に正解していた．よって，分類課題を理解した上でフェーズ III の課題 6 の回答をより高度にしようと挑戦していたため，Computational Thinking の得点上では下がってはいるものの，質的にはより高まっている可能性が示唆された．よって，グループ B の学習者には，Yes/No チャート学習システムが効果的であることが示された．

§3 グループ C

グループ C では，フェーズ I と III を比較すると，グループ C の得点の中央値は変化しなかったが，フェーズ II の経験により，フェーズ I とフェーズ III を比較すると，分類の条件に関して別の条件を試していた．よって，得点は変わらないながらも，問題を理解した上で分類の他の条件を試しており，Computational Thinking が向上したと考えられる．よって，グループ C の学習者には，Yes/No チャート学習システムが効果的であることが示された．

§4 システム正解群の特徴のまとめ

システム正解群の得点の中央値には有意な向上が見られず(T=5.50, *n.s.*)，回答時間も短くなっていなかった(T=55.00, *n.s.*)．しかし，フェーズ I より III の方が数学的に正確・高度な分類，もしくは他の条件を用いた分類を行っていたことから，フェーズ I での得点とは関係なく，Computational Thinking が高まったことが示唆された．そして，システム正解群全員の

フェーズ III における得点は 3.00 以上であり，手続化が身に付いていると判定される条件を満たしていた．フェーズ II の経験がなければフェーズ III の分類条件を記述できなかったと想定される学習者が見られたこと，フェーズ I よりフェーズ III の回答がより数学的に具体的・詳細な記述となっていた学習者が見られたことから，フェーズ II の経験を通して Computational Thinking が向上したことが認められた．よって，フェーズ II の経験が有効であること，つまり Yes/No チャート学習システムの有効性が示された．

(2) Yes/No チャート学習システムで不正解の学習者

フェーズ II の Yes/No チャート学習システムで正解できなかった学習者(システム不正解群)は 12 名であった．そして得点が上がった学習者 8 名，下がった学習者 1 名，維持した学習者 3 名であった．表 24 の分類に基づき，それぞれについて述べる．

§1 グループ D

グループ D では，フェーズ I とフェーズ III を比較すると，グループ D の得点の中央値は向上していた．しかし，フェーズ I の得点が全員 2.00 未満であり，ベン図の理解ができていなかった．また，フェーズ III の得点が 3.00 未満であり，Computational Thinking が身に付いているという判定基準を上回っていなかった．システム経験前にベン図の理解ができていなかったこと，そしてシステム経験後に Computational Thinking が身に付いていると判定することができなかったことから，フェーズ I の得点が 2.00 未満であった場合，フェーズ II の経験が Computational Thinking 育成にそれほど寄与しない可能性がある．つまり，グループ D の学習者には，Yes/No チャート学習システムはさほど有効に働かない可能性が示唆された．

§2 グループ E

グループ E では，フェーズ I とフェーズ III を比較すると，グループ E の得点の中央値は向上していた．そしてフェーズ III の得点は全員が 3.50 以上であり，Computational Thinking が身に付いているという判定基準を上回っていた．また，フェーズ I の得点が全員 2.00 以上であり，ベン図の理解ができていると判定された．そしてフェーズ I の得点が 2.00 であった 17,18 番の学習者は，フェーズ I の課題 2 では手続化のきっかけを掴めていなかったが，フェーズ II の経験を通して課題 6 では「一辺が等しい」,「直角がある」などの分類条件を用いて，段数は少ないながらも部分的に正解していた．また，フェーズ I の得点が 2.75 以上であった 16,19,20 番の学習者は，分類条件が「角がある」,「角がない」から，「60° の角がある」など，より具体的な数学的表現となっていた．これらはフェーズ II の経験がなければ回答できなかったと想定される．よって，システム経験前にベン図の理解ができており，システム経験後に Computational Thinking が身に付いていると判定されたことから，フェーズ I の得点が 2.00 以上の場合，グループ A,B,C と同様にフェーズ II の経験が

Computational Thinking の育成につながる可能性が認められた．よって，グループ E の学習者には，Yes/No チャート学習システムが効果的であることが示された．

§3 グループ F

グループ F(学習者 21 の 1 名)では，フェーズ I と III の得点を比較すると，フェーズ III で得点は減少していた．そして，フェーズ I と III の得点はいずれも 2.00 以下であり，ベン図の理解ができているにとどまっていた．ベン図の条件文も「角がある」,「角がない」と単純なものであるとともに，それを考えるのに時間がかかっており，ベン図による分類(課題 1)を全て完成させることができていなかった．また Yes/No などを用いた分類(課題 2)では，「アメーバのような形」と書いたにとどまり，他に記述は見られなかった．そしてフェーズ II でシステムを経験後，フェーズ III のベン図に対して混乱している様子が見られ，フェーズ I より得点が低くなっていた．そしてフェーズ III では，ベン図の条件文として「きれいな丸」と書いているだけであった．よって，フェーズ I の得点が低く，かつベン図の条件文を考えることが困難，もしくは条件文を考えるためにかなり時間がかかってしまうような状況である場合，フェーズ II の経験が Computational Thinking 育成にそれほど寄与しない可能性がある．つまり，グループ F の学習者には，Yes/No チャート学習システムはさほど有効に働かない可能性が示唆された．

§4 グループ G

グループ G では，フェーズ I と III の得点の中央値には変化が見られず，フェーズ I，III の得点はいずれも 2.50 以下であった．そしてフェーズ I の課題 2 とフェーズ III の課題 6 における分類では，数学的に別の条件を用いてなんとか解決しようとしていたが，段数が増えず手続化をうまく行うことができなかった．よって，フェーズ I の得点が 2.50 以下であった場合，フェーズ II の経験が Computational Thinking 育成にそれほど寄与しない可能性がある．つまり，グループ G の学習者には，Yes/No チャート学習システムはさほど有効に働かない可能性が示唆された．

§5 グループ H

グループ H では，フェーズ I と III の得点の中央値に変化が見られなかった．そしてフェーズ I，III の得点の中央値はいずれも 5.00 と満点であった．Yes/No チャート学習システムで正解は出せなかったものの，フェーズ III の課題 6 ではフェーズ I の課題 2 とは違った数学的に適切な分類条件を用いており(フェーズ I では「直角がある」,「対辺が平行」，フェーズ III では「全てが鋭角である」,「曲線を持たない」など)，解答時間も短くなっていた．このことから，フェーズ II の学習によって，数学的に別の分類条件を用いてフェーズ I も III も正解していたという観点で，Computational Thinking が育成されている可能性が示唆された．よって，グループ H の学習者には，Yes/No チャート学習システムが効果的であること

が示された.

§6 システム不正解群の特徴のまとめ

システム不正解群の得点の中央値には有意な向上が見られた(T=1.50, p < .05). そして 14 の学習者を除いて回答時間が短くなっており，回答時間の中央値も有意に減少していた(T =64.50, p < .01). そしてフェーズ I の得点が 2.00 未満の場合，フェーズ II の経験が Computational Thinking の育成にそれほど寄与しない可能性がある. 一方，フェーズ I の得点が 2.00 以上 2.50 以下の場合，フェーズ II の経験が Computational Thinking の育成に寄与する場合とそうでない場合のいずれの場合も想定された. また，フェーズ I の得点が 2.75 以上の場合，Computational Thinking がより高まっていることが示唆された.

よって，Yes/No チャート学習システムで正解を出せなかった場合，フェーズ I の得点によって学習者の反応およびシステムの効果が異なることが示唆された. そしてフェーズ I の得点によっては，フェーズ II の経験が有効に働いているということができ，Yes/No チャート学習システムの有効性が概ね示された.

4.6.5. 考察のまとめ

フェーズ III とフェーズ I の得点を比較すると，得点の中央値が 1%水準で有意に向上しており，システムの有用性は認められた. そして全体およびシステム不正解群では，フェーズ III はフェーズ I と比べて回答時間が有意に短くなっていた.

ここで，システム正解群の得点の中央値には有意な向上が見られなかったが，大半の学習者はフェーズ I より III の方が数学的により正確な分類を行っていた. Yes/No チャート学習システムで正解した学習者の大半は，フェーズ I の段階で手続化について理解していた. そしてフェーズ I において手続化を理解していない学習者も Yes/No チャート学習システムの経験によって，分類条件がより数学的に具体的・高度になったという観点において Computational Thinking が高まっていることが示唆された.

一方，システム不正解群の得点の中央値には有意な向上が見られた. そして 14 の学習者を除いて回答時間が短くなっており，回答時間の中央値は有意に減少していた. 大半の学習者は，システム II の経験が生かされ Computational Thinking がより高まっていることが示唆された. しかし，フェーズ I でベン図の理解ができていない学習者は，フェーズ III で手続化を理解しているという基準に到達できず，フェーズ II の経験が Computational Thinking の育成にそれほど寄与しない可能性が示唆された. そして，フェーズ I でベン図が理解できているものの手続化が理解できていない学習者は，システムの経験が Computational Thinking の育成に有効である場合とそうでない場合のいずれも想定され，個別に対応する必要性が示唆された. そしてフェーズ I で手続化に少しでも気づいている学習者は，システムの経験が有効に働き, Computational Thinking がより高まっていることが示唆された. よって, Yes/No チャート学習システムで正解を出せなかった場合，フェーズ I の得点，つまりベン図と

Yes/No チャートの理解度によって学習者の反応が異なることが示唆された.

そしてシステム正解群，システム不正解群ともに，フェーズ II の経験がなければフェーズ III の分類条件を記述できなかったと想定される学習者が多く見られた．よって，フェーズ II の経験を通して Computational Thinking が向上したことが認められた．つまり，フェーズ I からフェーズ III での得点向上もしくは回答の質が向上したことは，直接的にフェーズ II で経験した方法が習得できたためであると言える．よって，フェーズ II の活動は Computational Thinking の経験となっている．また経験学習の理論は，経験することがその能力を身につける有力な方法であることから，今回の結果は，Computational Thinking を育成するための方法として有望であることを示していると考えられる.

また，システムによって学習を明示的に可視化して，個別にフィードバックを与えることにより，頭の中だけで行うよりも経験学習の具体的経験と内省の一部を促していると想定される．そして，フェーズ II のシステムで扱う図形とフェーズ I と III で扱う図形は異なることから，フェーズ II の経験がフェーズ I,III の実践結果に近転移していると考えられ，概念化の一部を促進していると考えられる．よって，Computational Thinking を経験させることができていると考えられ，実際に学習者の Computational Thinking を向上させる結果になったのではないかと考えられる.

以上のことから，Computational Thinking を育成するために，ベン図，Yes/No チャート学習システムの活用が多くの学習者にとって有効であることが認められた．特に，フェーズ I の段階で課題 2 の Yes/No などを用いた手続化に少しでも気づいている学習者に対し，本システムは有効に働くことが認められた.

4.6.6. システムを用いた学習への示唆

システムを用いた学習への示唆としては，フェーズ I で課題 2 の Yes/No などを用いた手続化に全く気づかなかった場合，システムで Yes/No チャートなどを経験するよりも，まずベン図の分類条件を考えさせることや様々なベン図を作らせるといった学習を行わせることが必要となる可能性がある．またベン図を用いた分類において，分類条件を提示して分類させる経験を通して自身で分類条件を考えさせるといった段階的な学習を行うことが有効となる可能性がある．つまり，課題 2 の Yes/No などを用いた手続化の前に，まずベン図による分類を適切に行えるような学習が重要であると想定される．その上で，課題 2 の Yes/No などを用いた手続化において，ベン図で用いた分類条件を使うことができるといった手立てを講じることが一つの有効な手立てではないかと考えられる.

4.7. 実験 2:プログラミング経験がシステムの学習効果に与える影響

実験 1 では，ベン図，Yes/No チャート学習システムが Computational Thinking 育成に有効であることが示された．Yes/No チャートは分岐を視覚的に表現したものであり，プログラ

ミングで言えば if-else による二分，2 択のネスト構造で表現される．よってそれまでにプログラミング学習経験の中で if-else に関して実際にプログラミングを組んだことがあれば，すでに Yes/No チャートのアルゴリズムを学んでいることになる．その一方で，Yes/No チャートは構造が可視化されており，初学者にもその内容は理解されやすい．つまりプログラミング経験者と未経験者では Yes/No チャートを用いた学習効果に違いがないことが想定される．しかしそのことについては未検討であるため，実験 2 では，それまでのプログラミング経験が Yes/No チャート学習システムによる学習効果に影響を与えるかどうかについて検討する．

なお，今回の実験対象者は，教員養成系の大学生もしくは高校生であり，理系学部でプログラミングを専門的に学んだ者や開発経験者がいなかったことから，プログラミング経験を「ブロック型プログラミング経験」と「(簡易な)コーディング経験(ブロック型プログラミング経験以外の JavaScript や Python などのテキストベースのプログラミング)」とした．

4.7.1. 実施日および実践対象者の状況

2020 年 2 月に高校 9 名，大学 6 名の計 15 名の学生を対象に実践を行った．ここで 4 名の学習者がシステム利用前, 利用後のいずれのテストにおいても満点(5.00)であった．ここで，システムの効果を検証するために，これらの学習者を除外した．よって，有効なデータの対象となる参加者は 11 名(高校 7 名，大学 4 名)であった．有効なデータの取得率は 73.33％であった．

4.7.2. 実践計画

実験 1 と同様に，表 15 に示す実践モデルに基づき，課題 1 から順に実施した．課題 1,2,5,6 では，解答を紙に書かせた．また，課題 3 ではベン図学習システム，課題 4 では Yes/No チャート学習システムを用いた．実験 1 と同様に，課題 1,2,5,6 と課題 3,4 では，使用した図や条件が異なっている．そして課題 1,2 および 5,6 は 20 分，課題 3,4 は 30 分で実施した．ここで，参加者は全員がすでにベン図について学習済みであり，Yes/No チャート学習システムの効果について測定するため，ベン図については理解していることを前提とし，ベン図について理解できていない参加者がいた場合は，その参加者の結果は除外することとした．

4.7.3. 評価の手続き

実験 1 と同様に，表 18 の評価基準を用いて評価を行った．ここで，3 名の採点者それぞれが個別に採点し，その結果の平均点を得点とした．表 18 の Yes/No やそれに類する表現の使用に関する評価では，独自の分類が提示された場合にも，それが Yes/No チャートの考え方に類するならば，採点対象とした．ここで，手続化が行えているかについて評価するために，得点が 3.00 以上であることを基準として設ける．

4.7.4. 結果

実験の結果，課題 1 では参加者全員がベン図を理解していたことが確認され，実験の前提条件を満たしていた. そして Yes/No チャート学習システムの正解者は 3 名, 正解率は 27.27% であった.

表 25 に，Yes/No チャート学習システムの正解/不正解，Computational Thinking の得点を示す．Experience 1 は Scratch などのブロックプログラミング経験，Experience 2 はプログラミング言語でのコーディング経験のことである. また,「1」はそれらの経験があること,「0」はそれらのプログラミングの経験がないことを示す.

表 25 から，フェーズ I では，3 名の参加者の得点が 3.00 以上であった．一方，フェーズ III では，参加者全員が 3.00 以上の得点を獲得していた.

表 25 システムによる正解/不正解と得点の状況

No.	経験1	経験2	正解/不正解	得点	
				フェーズI	フェーズIII
1	1	0	×	2.33	4.67
2	1	1	○	2.33	3.00
3	1	1	×	3.00	5.00
4	1	1	×	4.67	5.00
5	1	0	×	2.33	4.33
6	0	1	○	4.67	5.00
7	0	1	×	2.33	4.33
8	0	0	×	2.00	5.00
9	0	0	×	2.67	4.00
10	0	0	×	2.00	4.67
11	0	0	○	2.00	5.00

1: 経験あり, 0: 経験なし　　(*N*=11)

ここで，Shapiro-Wilk 検定を用いてフェーズ I，III における得点が正規性を有するかについて評価した．その結果，得点に正規性は認められなかった (フェーズ I: W=0.72，p <.01, フェーズ III: W=0.77，p<.01)．このことから，得点をノンパラメトリックデータとして扱う必要があるが, 分散分析はロバスト性があることが報告されている[81]. よって本研究では, 分散分析を用いることとした.

まず, 本システムを用いた学習過程において, 正解した学習者と不正解した学習者の間で Computational Thinking 得点に差があるかどうかを評価した．そこで，システムにおける正解/不正解(水準は正解・不正解)とシステム利用(水準はシステム利用前・利用後)を要因とし Computational Thinking の得点を目的変数とする二元配置分散分析を行った．その結果を表

26 に示す.

表 26 から，得点に対しては，フェーズ II のシステム利用の主効果が有意であり，フェーズ III の得点はフェーズの得点よりも高かった．一方，システムによる正解/不正解の主効果と交互作用は有意ではなかった．したがって，システムによる正解/不正解で群を分けずに以下の分析を行うこととした.

表 26 システムによる正解/不正解と得点の関連性

		得点		得点		
		フェーズI	フェーズIII	正解/不正解	フェーズ	交互作用
システム	*Mean.*	3.00	4.33	0.00	23.68 **	0.85
正解群	*S.D.*	1.19	0.94			
システム	*Mean.*	2.67	4.63			
不正解群	*S.D.*	0.82	0.35			

** p <.01 (N=11, df=1,9)

次に，ブロック型プログラミング経験とコーディング経験に応じてグループに分け，グループごとの記述統計量を求めた．この結果を表 27 に示す.

表 27 から，フェーズ I では，ブロック型プログラミング経験およびコーディング経験いずれも有するグループが最も得点が高く，ブロック型プログラミング経験およびコーディング経験いずれも有さないグループが最も得点の低いことが把握された.

表 27 プログラミング経験の有無別に分けた群における記述統計量

			経験2			
			フェーズI		フェーズIII	
			Yes	No	Yes	No
経験1	Yes	*Mean*	3.08	2.33	4.42	4.33
		S.D.	0.95	0.00	0.83	0.00
	No	*Mean*	3.50	2.17	4.67	4.67
		S.D.	1.17	0.29	0.33	0.41

(N=11)

次に，それまでのプログラミング経験とコーディング経験が Computational Thinking の得点にどのような影響を与えるかについて評価するために，混合計画分散分析を行った．その結果を表 28 に示す.

表 28 から，システム利用 (事前事後) の主効果が有意であったが，システムによる正解

不正解の主効果とおよび交互作用は有意ではなかった.

表 28 Computational Thinking の得点とプログラミング経験，システム経験との関連性

	SS	*df*	*MS*	*F*	
A	0.17	1	0.17	0.17	*n.s.*
B	1.17	1	1.17	1.13	*n.s.*
AB	0.06	1	0.06	0.06	*n.s.*
error [S(AB)]	7.25	7	1.04		
C	12.25	1	12.25	27.81	**
AC	0.03	1	0.03	0.06	*n.s.*
BC	1.00	1	1.00	2.27	*n.s.*
ABC	0.11	1	0.11	0.25	*n.s.*
error [CS(AB)]	3.08	7	0.44		

**p <.01 (N=11)

注:

A: ブロック型プログラミング経験

B: コーディング経験

C: 事前/事後

4.7.5. 考察

本章では，次のような結果が得られた.

(i) システムの正解･不正解に関わらず, Computational Thinking の得点が有意に向上した.

(ii) ブロック型プログラミング経験，コーディング経験の有無にかかわらず，Computational Thinking の得点が有意に向上した.

(i), (ii)の結果から，プログラミング経験の有無は，Computational Thinking を育成する Yes/No チャートを用いた学習効果には影響を与えないことが示唆された．Yes/No チャートを用いて Computational Thinking を育成する際には，それまでのプログラミング経験を考慮する必要はないと想定される.

Yes/No チャートを用いた Computational Thinking 育成に対してプログラミング経験の有無が関与しなかった理由として，Yes/No チャートは初学者を含めた多くの学習者にとって理解しやすい構造，見てすぐ把握できる構造を持っているため，Computational Thinking の得点に影響しなかった可能性がある．そしてそれまでにブロック型プログラミングやコーディング経験があったとしても，今回の課題は分岐の入れ子構造となり，これをブロック型プログラミングやプログラミング言語で表記すれば，可読性が低下してしまう．よって，分類を題材とする Computational Thinking 育成においては，Yes/No チャートを経験すれば十分で

あるとも言える可能性がある．

そのため，プログラミングの経験がなくても，分類課題を実装した Yes/No チャート学習システムを用いることは，一定の効果が期待できる．これは，プログラミングを学習していない小学生に対してもその有用性が期待できる．この学習システムを小学生に利用する場合には，条件や図形の数を減らしたり，表現を既に学習したものに変更したりする必要があると想定されることから，本システムを学習段階に応じた適切な設定を施すことが求められる．

5. 結論

5.1. 創造性とプログラミングに対する意識に関する調査研究

第 3 章に示した調査研究の結果，以下の 6 点の結果が得られた．

(i)　男女間におけるプログラミングに対する様々な意識の全項目の各平均値には有意差が見られ，いずれも男子が女子より平均値が高かった．

(ii)　男女間における創造的態度尺度の全因子の各平均値には有意差が見られ，いずれも男子が女子より平均値が高かった．

(iii)　相関については，プログラミングに対する様々な意識の全項目間，各創造的態度の関係，柔軟性，分析性，進取性とプログラミングに対する意識の大半の項目間で，相関が認められた．一方，持続性，想像性，協調性とプログラミングに対する様々な意識の項目間での相関は性差が認められる項目が多数見られた．また，プログラミングに対する様々な意識の 5 項目と創造的態度の多くの因子間において，関連性が認められた．

(iv)　柔軟性，分析性，進取性については，プログラミングに対する様々な意識の 5 項目のいずれも有意な関連性が認められ，いずれも男子が女子よりも平均値が高かった．

(v)　柔軟性因子を再検討し，「知識・対外型柔軟性」因子，「臨機応変・問題解決型柔軟性」因子が抽出された．

(vi)　柔軟性 2 因子とプログラミングに対する様々な意識のそれぞれの項目には関連性が認められた．また「プログラミングの応用期待感」において，「知識・対外型柔軟性」因子と「臨機応変・問題解決型柔軟性」因子の相関係数に有意差が見られ，「臨機応変・問題解決型柔軟性」の方が強い相関があることが認められた．

そして，これらの実態があることを踏まえたカリキュラムの策定が重要であることを指摘した．また，性別に応じたカリキュラム策定が重要であること，およびレディネスの様々な段階に対応した題材を複数準備することなども重要であることを指摘した. Computational Thinking の観点から見れば, 違ったものの中にパターンがあるかについて考えさせる機会を与える活動を行うことによって，Computational Thinking の「分解」や「パターン化」を育成することにも繋がり，共通部分を見つけてそこから要素を取り出す「抽象化」と関連性があると考えられる．よって，柔軟性を高める活動が，Computational Thinking 育成を指向した活動に有用となる可能性があることを指摘した．

プログラミングを用いて Computational Thinking の育成が可能であること，プログラミングで創造性の育成が可能であること，Computational Thinking と創造性の関連性については指摘されているが，我が国のプログラミング教育で育成が期待されているプログラミングに対する様々な意識とレディネスとしての創造的態度との関連性を示したことは，今後，プログラミング教育の充実化が求められている中で，重要な知見を提供できたのではないかと考えられる．よって調査研究の新規性・独自性が示されたと考えられる．

5.2. Computational Thinking を育成する学習支援システムの開発とその評価

本研究では，Computational Thinking を育成することを目的とした分類課題に対するベン図，Yes/No チャート学習システムの設計・開発を行った．そして経験学習を取り入れた実践モデルを開発し，学習システムの評価を行った．その結果，開発したシステムは Computational Thinking 育成に有効であることが示された．また，フェーズ I の得点や回答内容によってその学習効果が異なることを指摘した．

各教科内の学習内容を用いて Computational Thinking を育成することに対し，図形の分類課題を用いることが有効であること，そして本システムが有効であることを示すことができた．さらに経験学習を組み合わせることによって，方法論が確立していなかった Computational Thinking の育成に対し，一つのモデルを示すことができたと考えられる．よって，今後 Computational Thinking やプログラミング的思考を育成するための基礎的な知見を提供できたのではないかと考えられ，本研究の新規性が示されたと考えられる．

次に，Yes/No チャート学習システムを用いた Computational Thinking 育成に対し，ブロック型プログラミングやコーディングの経験は Computational Thinking の得点に影響を与ええないことが明らかになった．事前のプログラミング経験が，Yes/No チャート学習システムを用いた Computational Thinking 育成に影響を与えないことが示されたことで，Yes/No チャートによる Computational Thinking 育成が，それまでにブロック型プログラミングやコーディング経験がほとんどない学習者に対しても有効である可能性が示された．

このことによって，Yes/No チャートの条件や分類対象の難易度を調整することで，小学生の Computational Thinking 育成も可能である可能性が示された．我が国のプログラミング教育では，プログラミング的思考を各教科内で育成することが求められている現状の中で，Computational Thinking やプログラミング的思考を育成することが可能であることを示すことができ，本研究の新規性・独自性が示されたと考えられる．

5.3. 今後の課題

以上に示したように，本研究の新規性は十分に示されたと考えられる．しかし本研究には多くの課題が残されている．それらの課題について述べる．

1 点目として，Computational Thinking の尺度を用いた調査を実施する必要があることである．現在 Computational Thinking の尺度開発についての研究がなされており[82]，それらの尺度なども参考にしながら，プログラミングに対する様々な意識，および創造性などとの関連性を把握することで，Computational Thinking がコーディングスキル以上の様々なスキルを身につけることに有用であることがより詳細に示されるのではないかと考えられる．

2 点目として，創造性尺度を再検討する必要がある．本研究では，創造性を発揮するような経験を有していないと想定される学生を対象としたことから，レディネスとしての創造

的態度に着目した．しかし創造性を測定するための尺度は，Guilford の S-A 創造性検査や Munzert の創造性尺度[49]をはじめ，様々なものが存在する．これらの関連性を把握し，1 点目の知見と結びつけることが重要であると考えられる．

3 点目として，システムにさらに Computational Thinking を育成するための機能を実装する必要がある．適切なフィードバックを返すシステムを構築するためには，学習者がどこまでできているか，そしてどのようなプロセスで Yes/No チャートを作成しているかを把握する必要がある．よって，作成している Yes/No チャートの履歴を取得するとともにその差分について評価し，現在の進捗や完成度を把握する機能を実装する必要があると考えられる．そして部分的に正解しているか否かを返す機能を実装し，合っている部分とそうでない部分を判別することで，間違っている箇所に焦点を当てた適切なフィードバックを行うこと，一部分を取り出して，そこだけが合っているかどうかを判定させる機能を実装することで，少しずつ正解に近づけるような作成も支援することが可能となる．

4 点目として，対象者の状況に合わせ，ベン図における条件を提供する必要があることである．今回の対象者は高校生以上であり，ベン図における分類条件を自身で考えることに大きな問題は生じなかった．しかし小・中学生に対して実践する場合や，問題の難易度によっては，条件が全く思い浮かばない対象者がいることが想定される．その場合，それ以降の活動に支障が生じてしまう．よって，対象者の学習段階や学習内容との関連性や難易度を把握した上で，フェーズ I のベン図を用いた分類においても条件ブロックを準備する必要があると想定される．

5 点目として，図形の分類課題以外の課題を準備する必要があることである．図形の分類課題以外にも，既存の教科内容には植物や動物の分類など，多くの分類課題が存在する．それらを実装するだけではなく，教員がカスタマイズできる機能を実装し，システムの充実化を図る必要がある．

6 点目として，統制条件を設定しシステムを用いた場合と用いない場合とを比較することである．この検討により，システムのより詳細な評価のみならず，提案した実践モデルの有効性を検討することが可能であると想定される．

7 点目として，Yes/No チャートの学習効果をブロック型プログラミング学習システムと比較検討する必要があることである．Yes/No チャートの学習効果にそれまでのプログラミング経験が影響しないことは示されたが，実際に Yes/No チャート学習システムと同一課題を実装したブロック型プログラミング学習システムの学習効果と比較は行っていない．よって，Yes/No チャート学習システムと同一課題を実装したブロック型プログラミング学習システムの学習効果を比較し，その有用性を検討する必要がある．

8 点目として，Computational Thinking を育成する様々なアプローチを検討し，その機能を実装することである．今回は学習者それぞれの学習を個別最適化し，適切なフィードバックを返すようなシステムを開発したが，これまでに相互評価機能や同時編集を可能とする機能などの実装を別のシステムに対して行っている．それらを本システムにも実装し，多様な

学習を提供するシステムとして改善していく必要がある．

9 点目として，Yes/No チャート以外を用いた方法について検討することである．Yes/No チャート以外にもアクティビティ図やフローチャートをはじめとした表現方法が存在する．学習者の発達段階に合わせた Computational Thinking 学習法についても検討し，定量的に評価する必要がある．

さらに，本研究の結果は，参加者数が少ないことや，参加者が大学生や高校生であったことから，限られたものとなっている．したがって，今後はさらに参加者を増やし，様々な年齢層での実験を行う必要がある．そして，今回の評価方法の妥当性についての検討や遅延テストによる定着率の評価，転移がどのように起こっているかについての検討，UI の改善なども行い，各教科内で Computational Thinking を育成する実践に接続していく必要があろう．それらについては今後の課題とする．

参考文献

[1] レイ・カーツワイル, シンギュラリティは近い [エッセンス版] 人類が生命を超越するとき, NTT出版 (2016)

[2] レイ・カーツワイル (井上健, 小野木明恵, 野中香方子,福田実 (訳)), ポスト・ヒューマン誕生 コンピュータが人類の知性を超えるとき, NTT出版 (2007)

[3] 内閣府, Society 5.0とは (n.d.) https://www8.cao.go.jp/cstp/society5_0/society5_0-1.pdf

[4] ATC21s (Assessment and Teaching of 21st Century Skills), 21 Century Skills (n.d.) http://www.atc21s.org/

[5] 利根川裕太, 佐藤智, 先生のための小学校プログラミング教育がよくわかる本, 翔泳社, p.24 (2017)

[6] 首相官邸, ニッポン一億総活躍プラン (2018) https://www.kantei.go.jp/jp/singi/ichiokusoukatsuyaku/pdf/gaiyou1.pdf

[7] 首相官邸, 日本再興戦略 (2018) http://www.mext.go.jp/component/a_menu/education/detail/__icsFiles/afieldfile/2016/08/09/1375321_02_1.pdf

[8] 文部科学省, GIGA スクール 構想の実現へ (2020) https://www.mext.go.jp/content/20200625-mxt_syoto01-000003278_1.pdf

[9] Wing, J. M, Computational Thinking, *Communications of the ACM*, **49**(3), pp.33–35 (2006) (日本語訳: 中島秀之: 計算論的思考, 情報処理, **56**(6), pp.584–587 (2015))

[10] 中植正剛, 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, 児童教育学研究, **36**, pp.109‐122 (2017)

[11] Tabesh, Y., Computational Thinking: A 21st Century Skill, *Olympiads in Informatics 2017*, **11**, Special Issue, pp.65–70 (2017)

[12] Wing, J. M., Computational Thinking Benefits Society. 40th Anniversary Blog of Social Issues in Computing (2014) http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

[13] 文部科学省 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について (2016) http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm

[14] Fagerlund, J., Häkkinen, P, Vesisenaho, M., and Viiri, J., Computational Thinking in Programming with Scratch in Primary Schools: A Systematic Review, *Computer Applications in Engineering Education*, pp.1–17. (2020) https://doi.org/10.1002/cae.22255

[15] Wong, G.K-W., and Cheung, H.-Y., Exploring Children's Perceptions of Developing Twenty-first Century Skills through Computational Thinking and Programming, *Interactive Learning Environments*, **28**(4), pp.438–450 (2020)

[16] Rotem, I-F, Hershkovitz, A., Eguíluz, A., Garaizar, P., and Guenaga, M., The Associations Between Computational Thinking and Creativity: The Role of Personal Characteristics, *Journal*

of Educational Computing Research, **58**(8), pp.1415–1447 (2020). https://doi.org/10.1177/0735633120940954.

[17] Doleck, T., Bazelais, P., Lemay, D. J., Saxena, A., and Basnet, R. B., Algorithmic Thinking, Cooperativity, Creativity, Critical Thinking, and Problem Solving: Exploring the Relationship between Computational Thinking Skills and Academic Performance. *Journal of Computers in Education*, **4**(4), pp.355–369 (2017) https://doi.org/10.1007/s40692-017-0090-9

[18] Resnick, M., *Lifelong Kindergarten: Cultivating Creativity through Projects, Passion, Peers, and Play*, The MIT Press (2017) (ミッチェル・レズニック, 村井裕実子, 阿部和広, 伊藤穣一, ケンロビンソン, 酒匂寛(翻訳), ライフロング・キンダーガーテン 創造的思考力を育む4つの原則, 日経BP社 (2018))

[19] 阪東哲也, 藤原伸彦, 曽根直人, 長野仁志, 山田哲也, 伊藤陽介, 情報活用能力育成を基盤とした小学校プログラミング教育カリキュラム・マネジメントの提案, 鳴門教育大学情報教育ジャーナル, 16, pp. 27–36. (2019)

[20] Rodríguez-Martínez, J. A., González-Calero, J. A., and Sáez-López, J. M., Computational Thinking and Mathematics using Scratch: an Experiment with Sixth-Grade Students, *Interactive Learning Environments*, **28**(3), pp.316–327. (2020)

[21] Papert, S., *Mindstorms: Children, Computers, and Powerful Ideas*, Basic Books (1980) (奥村喜世子訳: マインドストーム -子供, コンピューター, そして強力なアイデア, 未来社 (1982))

[22] 阪東哲也, 黒田昌克, 福井昌則, 森山潤: 我が国の初等中等教育におけるプログラミング教育の制度化に関する批判的検討, 兵庫教育大学学校教育学研究, **30**, pp.185–196 (2017)

[23] Denning, P. J., The Profession of IT, Beyond Computational Thinking, *Communications of the ACM*, **52**(6), pp.28–30 (2009) https://core.ac.uk/download/pdf/36728065.pdf

[24] BBC Bitesize, Introduction to Computational Thinking (2020) https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1

[25] Google for Education, Exploring Computational Thinking: CT Overview (n.d.) https://edu.google.com/resources/programs/exploring-Computational Thinking/#!ct-overview

[26] International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA): Operational Definition of Computational Thinking for K–12 Education (2011) http://www.iste.org/docs/ct-documents/Computational Thinking-operationaldefinition-flyer.pdf

[27] Shute, V. J., Sun, C., and Clarke, J. A, Demystifying Computational Thinking, *Educational Research Review*, **22**, pp.142–158 (2017)

[28] 太田剛, 森本容介, 加藤浩, 諸外国のプログラミング教育を含む情報教育カリキュラムに関する調査 –英国, オーストラリア, 米国を中心として-, 日本教育工学会論文誌,

40(3), pp.197–208 (2016)

[29] Gov.UK: Department for Education, National Curriculum in England, Computing Programmes of Study (2013) https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

[30] Harvard University, Computational Thinking with Scratch (n.d.) http://scratched.gse.harvard.edu/ct/index.html

[31] Brennan, K. and Resnick, M., New Frameworks for Studying and Assessing the Development of Computational Thinking, *Annual American Educational Research Association Meeting*, (2012) http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

[32] 文部科学省, 小学校プログラミング教育の手引 (第三版). (2020) https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1403162.htm

[33] 岡山県総合教育センター, 小学校プログラミング教育に関する研究―プログラミング的思考を育成する授業づくり―, 岡山県総合教育センター研究紀要, **13**, 19-05. (2020)

[34] リンダ・リウカス (著), 鳥井雪 (翻訳), ルビィのぼうけん こんにちは！プログラミング, 翔泳社 (2016)

[35] 林雄介, 平嶋宗, プログラミング的思考の経験を通じた学習内容の理解・応用・批評的評価 −ベン図, Yes/No チャート, ビジュアルプログラミング言語, 対話型ロボットを用いた小学校における授業実践−, 日本教育工学会研究報告集, JSET19-1, pp.31–38 (2019)

[36] 平嶋宗, 福井昌則, 林雄介, 分類課題に対する Yes/No チャートの組み立てとそのプログラム化としての思考経験を通したプログラミング的思考の育成 ～ 対話型ロボットを用いた教えることによる学習の適用～, 電子情報通信学会技術研究報告, **118**(214), ET2018‑31, pp.19–24 (2018)

[37] 平嶋宗, 福井昌則, 林雄介, 異種リソースから作られた指示の対比によるプログラミング的思考の振返り, 教育システム情報学会第 43 回全国大会, pp.133–134 (2018)

[38] 文部科学省, 諸外国におけるプログラミング教育に関する調査研究報告書 (2015) http://jouhouka.mext.go.jp/school/pdf/programming_syogaikoku_houkokusyo.pdf

[39] 小学校学習指導要領 (平成 29 年告示) (2017) http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/03/18/1413522_001.pdf

[40] 高等学校学習指導要領 (平成 30 年公示) (2018) http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/07/11/1384661_6_1_2.pdf

[41] 文部科学省, 高等学校学習指導要領解説 情報編 (2018) http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/07/13/1407073_11.pdf

[42] 総務省, プログラミング人材育成の在り方に関する調査研究 (2016) http://www.soumu.go.jp/main_content/000361430.pdf

[43] 赤堀侃司, プログラミング教育における論理的な思考とは何か, 学習情報研究論文誌, **261**(4), pp.56‐61 (2018)

[44] 谷田親彦, 足立泰彦, 臼坂高司, 上田邦夫, 情報教育の学習内容への興味・関心と自己評価に基づく教科「情報」学習者の類型化, コンピュータ＆エデュケーション, **22**, pp.100‐105 (2007)

[45] 本郷健, 松崎寛幸, 「情報基礎」のプログラミング学習における教材が学習者の情意的側面に与える影響, 日本産業技術教育学会誌, **40**(2), pp.61–69 (1998)

[46] 森山潤, 青木淑香, 鬼藤明仁, 「情報とコンピュータ」における学習の有用性に対する生徒の意識, 教育システム情報学会誌, **23**(1), pp.33–39 (2006)

[47] Guilford, J.P., *Intelligence, Creativity, and Their Educational Implications*, Knapp, San Diego. (1968)

[48] Wallas, G., *The art of Thought*, London: Jonathan Cape. (1926)

[49] ムンツァート, A.W., (松野武(訳)), 右脳左脳の IQ テスト -能力パワーアップ!!, 東京図書 (1982) (Munzert, A.W., *Test of your I.Q.*, New York, USA: H/U Publications (1980))

[50] Csikszentmihalyi, M., *Creativity, In R.J.Sternberg*, Encyclopedia of human intelligence, Macmillan,pp.298–306 (1994)

[51] 恩田彰, 創造性教育の展開, 恒星社厚生閣, pp.110–114 (1994)

[52] Plucker, J. A., Beghetto, R. A., and Dow, G. T., Why Isn't Creativity More Important to Educational Psychologists? Potentials, Pitfalls, and Future Directions in Creativity Research, *Educational Psychologist*, **39**(2), pp.83–96 (2004)

[53] Schank, R., and Childers, P., *The Creative Attitude*, Macmillan Publishing Company, New York (1988)

[54] 久保薗悦子, 風景構成法における創造性:語りと構成の観点から, 神戸大学大学院人間発達環境学研究科研究紀要, **7**(1), pp.33–42 (2013)

[55] 林文俊, 創造的態度の測定尺度に関する研究 –理工系男子大学生を対象とした予備的検討–, 愛知工業大学総合技術研究所研究報告, 創刊号, pp.133–136 (1999)

[56] 林文俊, 大学生の創造的態度に関する研究 :文系女子大学生を対象とした検討, 日本性格心理学会発表論文集, **9**, pp.90–91 (2000)

[57] 豊島禎廣, 庭瀬敬右, 中学生の創造的態度についての研究 -「原体験」と学力との関連を通して-, 理科教育学研究, 41(2), pp.1–7 (2000)

[58] 繁桝算男, 横山明, サム=スターン, 駒崎久明, 日米学生の創造的態度の因子分析による比較研究, 心理学研究, **64**(3), pp.181–190 (1993)

[59] 国立教育政策研究所, OECD生徒の学習到達度調査 PISA2012年 問題解決能力調査 –国際結果の概要–, (2014) https://www.nier.go.jp/kokusai/pisa/pdf/pisa2012_result_ps.pdf

[60] Star, J. R., and Johnson, B. R., Flexibility in Problem Solving: The Case of Equation Solving, *Learning and Instruction*, **18**(6), pp.565–579 (2008)

[61] Boekelder, A. W. B. M., Boekelder, A., and Steehouder, M. F., Selecting and Switching: Some Advantages of Diagrams for Presenting Instructions, *IEEE Transactions on Professional Communication*, 41(4), pp.229–241 (1998)

[62] 穐山貞登, 創造性, 培風館 (1975)

[63] 青柳肇, 創造的構えテスト作成の試み, 大和学園女子短期大学紀要, **5**, pp.1‐7 (1980)

[64] 田中祥司, 髙橋広行, ブランドの「本物感」を構成する要素の測定,流通研究, **19**(1), pp.39–52 (2016)

[65] Nunnally, J. C., and Bernstein,I. H., *Psychometric Theory* (3rd ed.), McGraw-Hill, New Yorks (1994)

[66] Tabachnick, B. G., and Fidell, L. S., *Using Multivariate Statistics* (5th ed.), Pearson Education, Boston (2007)

[67] Nunnally, J. C., *Psychometric Theory*, McGraw-Hill, New York (1978)

[68] Hair, J. F., Black, W. C., Babin, B. J., and Anderson, R. E., *Multivariate Data Analysis* (7th ed.), Pearson Education, Boston (2014)

[69] 久保田進彦, 同一化アプローチによるブランド・リレーションシップの測定, 消費者行動研究, **16**(2), pp.1–26. (2010)

[70] Kolb, D. A., *Experiential Learning: Experience as the Source of Learning and Development*, Prentice Hall (1984)

[71] Lindsey, L., Berger, N., *Experiential Approach to Instruction, In Instructional-Design Theories and Models*, Volume III, Chapter 7 (2009)

[72] 中原淳, 経験学習の理論的系譜と研究動向, 日本労働研究雑誌, **55**(10), pp.4‐14 (2013)

[73] Hirashima, T., Yamasaki, K., Fukuda, H., and Funaoi, H., Kit-build Concept Map for Automatic Diagnosis International Conference on Artificial Intelligence in Education, pp.466–468 (2011)

[74] Hirashima, T., Yamasaki, K., Fukuda, H., and Funaoi, H., Framework of Kit-build Concept Map for Automatic Diagnosis and its Preliminary Use. RPTEL, **10**(17), pp.1–18 (2015)

[75] Skemp, R. R., Relational Understanding and Instrumental Understanding, *Mathematics Teaching*, **77**(1), pp.20–26 (1976)

[76] Anderson, L. W., and Krathwohl, D. R., *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*, Allyn & Bacon. Boston, MA (2001)

[77] 石井英真, 「改訂版タキソノミー」によるブルーム・タキソノミーの再構築:知識と認知過程の二次元構成の検討を中心に, 教育方法学研究, **28**, pp.47‐58 (2003)

[78] 小林祐紀, 兼宗進, 白井詩沙香, 臼井英成, これで大丈夫! 小学校プログラミングの授業 3+αの授業パターンを意識する[授業実践39], 翔泳社 (2018)

[79] Horiguchi, T., Imai, I., Toumoto, T., and Hirashima, T., Error-Based Simulation for Error-

Awareness in Learning Mechanics: An Evaluation, *Journal of Educational Technology and Society*, **17**(3) (2014)

[80] Collins, A., *Cognitive Apprenticeship: The Cambridge Handbook of the Learning Sciences*, R.Keith Sawyer (Ed.), Cambridge University Press, pp.47–60 (2006)

[81] Khan, A., and Rayner, G.D., Robustness to Non-Normality of Common Tests for the Many-Sample Location Problem, *Journal of Applied Mathematics and Decision Sciences*, **7**(4), 187–206 (2003)

[82] Korkmaz, Ö., Cakir, R., and Özden, M. Y., A Validity and Reliability Study of the Computational Thinking Scales (CTS), *Computers in Human Behavior*, **72**, pp.558–569. (2017)

本研究に関連する主要な成果

第 3 章

1. 福井 昌則, 黒田 昌克, 森山 潤, 平嶋 宗, 高校生のプログラミングに対する意識と創造的態度との関連性, 日本教育情報学会論文誌, **34**(3), pp.19–28, 2019.
2. 福井 昌則, 石川 岳史, 森山 潤, 平嶋 宗, 創造的態度における柔軟性とプログラミングに対する様々な意識との関連性 -高校生を対象とした実証研究-, 日本教育情報学会論文誌, **35**(1), pp.25–36, 2019.
3. Masanori Fukui, Masakatsu Kuroda, Jun Moriyama and Tsukasa Hirashima, The Relationship between Students' Creative Attitudes and Consciousness of Computer Programming, *Proceedings of the International Conference on Creativity and Innovation 2018 (ICCI2018)*, pp.58–72, 2018.

第 4 章

1. 林 雄介, 福井 昌則, 平嶋 宗, 対話型ロボットを利用したプログラミング的思考の「教えることによる学習」, コンピュータ利用教育学会論文誌, 46, pp.38‐45, 2019.
2. Masanori Fukui, Yuji Sasaki, Jo Hagikura, Jun Moriyama and Tsukasa Hirashima, The Effect of Prior Programming Experience on the Use of Binary Tree Learning System for the Promotion of Computational Thinking, *Solid State Technology*, **63**(1s), pp.746–754, 2020.
3. 福井 昌則, 佐々木 雄司, 萩倉 丈, 林 雄介, 平嶋 宗, Computational Thinking を育成する組み立て式ベン図・Yes/No チャート学習システム環境の設計・開発および評価, 人工知能学会論文誌, **35**(6), pp.1–13, 2020.
4. Masanori Fukui, Jo Hagikura, Tomoya Bansho, Yuji Sasaki, Masakatsu Kuroda, Jun Moriyama and Tsukasa Hirashima, Block Type Programming Environment Enabling Online Peer Assessment for Promoting Collaborative Learning, *Proceedings of the 26th International Conference on Computers in Education (ICCE2018)*, pp.343–345, 2018.
5. Masanori Fukui, Jo Hagikura, Yuji Sasaki, Jun Moriyama, Yusuke Hayashi and Tsukasa Hirashima, Development of a Visual Programming Environment that Enables Simultaneous Editing to Promote Collaborative Learning, *Business Innovation and Engineering Conference (BIEC 2020)*, p.48, 2020.
6. Masanori Fukui, Yuji Sasaki, Jo Hagikura, Jun Moriyama and Tsukasa Hirashima, The Effect of Prior Programming Experience on the Use of Binary Tree Learning Systems to Promote Algorithmic Thinking in Computational Thinking, *Proceedings of the 5th International Conference on Management, Engineering, Science, Social Science and Humanities (iCon–MESSSH 20)*, p.54, 2020.

謝辞

本研究の遂行にあたり，指導教員である平嶋宗教授，林雄介准教授には，本当に懇切丁寧なご指導をいただきました．遠方に住んでいる私を快く受け入れて下さり，様々な点についてご配慮いただきましたことにつきましても，感謝申し上げます．

修士課程時代の指導教員であり，現在の上司でもある兵庫教育大学学校教育研究科 森山潤教授には，教育の観点から様々なご指導をいただき，業務遂行においても多くのご支援を賜りました．

共同研究者である慶應義塾大学環境情報学部学生で孫正義育英財団正財団生の佐々木雄司さん，関西学院大学理工学研究科院生の萩倉丈さんには，様々な分野の研究活動において多大なるご協力をいただきました．海外で一緒に発表したのもよい思い出です．

大阪電気通信大学総合情報学部デジタルゲーム学科 高見友幸教授には，研究の機会を多く提供していただきました．修士時代の出身研究室の同期である兵庫教育大学連合教育学研究科 黒田昌克先生には，多くの共同研究を通して様々な知見をご提供いただきました．出身研究室の後輩である神奈川県立津久井浜高等学校情報科教諭 野村新平先生，兵庫教育大学大学院生 下地勇也先生，国領未来先生，山下義史先生には，公私ともにお世話になりました．

関西学院高等部数学科宮寺良平先生には，研究活動に興味を持つきっかけを与えてくださり，大学教員になろうという目標を持たせていただきました．先生との出会いがなければ，今の私はありません．

本研究は，日本学術振興会特別研究員奨励費 18J20489，日本学術振興会スタート支援 20K22187，第 6 回未来教育研究所研究助成，平成 28 年度科学技術融合振興財団 補助金助成，中山隼雄科学技術文化財団 2018 年度第 2 回国際交流助成，兵庫教育大学 兵庫教育大学同窓会研究助成金によって実施されたものです．様々な研究費をいただくことができ，本研究を遂行することができました．

30 歳半ばから研究を開始し，大学に勤務するという状況に至ることは，当時思ってもいなかったことでした．このようなことが実現できたのは，素晴らしい先生，研究仲間，同僚，後輩に恵まれた結果によるものです．皆様に心より感謝申し上げます．

最後になりましたが，このような私の研究活動について理解を示し，様々な点において協力・配慮してくれた妻に感謝の意を記します．

2021 年 1 月吉日

福井 昌則