プラン自動提案機能を有した プログラミング的思考育成支援システムの開発

山本拓人[1]　内田智之[2]　川本佳代[2,3]　鈴木祐介[2]
宮原哲浩[2]　林雄介[3]　平嶋宗[3]

概要：2020 年からプログラミング的思考を育成するプログラミング教育が小学校で必修化されるなど，プログラミング的思考の育成の重要性が強く叫ばれている．そこで，本研究では，プログラミング的思考の育成をめざし，学習者の知識に基づき適切な課題を課す学習プランを設計する．また，その学習プランを自動提案する機能を持った学習システムを開発する．さらに，評価実験によりその有用性を示す．

キーワード：プログラミング的思考，学習システム，学習プラン

Development of a Support System with a Function Automatically Proposing Plans to Foster Computational Thinking

TAKUTO YAMAMOTO[1]　TOMOYUKI UCHIDA[2]　KAYO KAWAMOTO[2,3]　YUSUKE SUZUKI[2]
TETSUHIRO MIYAHARA[2]　YUSUKE HAYASHI[3]　TSUKASA HIRASHIMA[3]

Abstract: From 2020, programming education to foster computational thinking became compulsory in elementary schools, and the importance of fostering computational thinking is strongly emphasized. Thus, in this paper, we design a learning plan that imposes appropriate exercises based on the learner's knowledge with the aim of fostering computational thinking. We also develop a support system that has a function of automatically proposing plans to foster computational thinking. Furthermore, its usefulness is shown by reporting experimental results.

Keywords: Computational Thinking, Learning System, Education Plan

1. はじめに

プログラミング的思考を育成するプログラミング教育が 2020 年から小学校で必修化されるなど，プログラミング的思考の育成の重要性が叫ばれている．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[1]において，プログラミング的思考とは,「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と定義され，コンピュータやプログラミングの概念に基づいた問題解決型の思考のことであり,主に「分解」,「抽象化」,「一般化」,「組合せ」の 4 つの要素に分解される[2]．また，文部科学省[3]は，小学校学習指導要領（2017 年告示）において，プログラミング学習において用いる教材の中に平面図形問題を挙げている．そこで，プログラミング的思考の育成をめざし，三島[4]は平面図形問題を用いたプログラミング的思考育成システムを開発した. さらに，プログラミング的思考の元となる要素に，論理的思考[5]がある．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[1]において,「読解力，論理的思考力，創造性，問題解決能力などは,時代を超えて常にその重要性が指摘されており，これからの時代においてもその重要性が変わることはない」とされているなど，論理的思考についても，注目されているスキルの一つであると言える．科学コミニュケーション研究所[6]は，論理的思考の中でも特に，問題の構造を的確に捉える力，筋道を立てて考える力，論理的整合性のある表現をする力の３つの要素に着目している．これらを元に論理的思考の育成をめざし，フローチャート作成課題を用いた学習システム「ろっつ」[7]，数論問題を用いた学習システム[8]，平面図形問題を用いた論理的思考力育成支援システム[9]などがある．

平面図形問題や数論問題等を用いてプログラミング的思考を育成するには，それらの問題に関する知識が必要となり，必要な知識が不足しているとプログラミング的思考を効果的に育成することが難しい．そこで，本研究では，

1 広島市立大学情報科学部
Faculty of Information Sciences, Hiroshima City University
2 広島市立大学大学院情報科学研究科
Graduate School of Information Sciences, Hiroshima City University
3 広島大学大学院先進理工系科学研究科
Graduate School of Advanced Science and Engineering, Hiroshima University

平面図形問題や数論問題等に関する知識を調べる問題を事前テストとして解いてもらった結果を元に学習者の知識を把握し，効果的にプログラミング的思考を育成する学習プランを自動提案する機能を持った学習システムを開発することを目的とする．

2. 準備

2.1 論理的思考とプログラミング的思考

論理的思考力とは，道理や筋道に沿って思考を巡らせて結論を導いたり，複雑な事象をわかりやすく説明したりすることができる能力のことを指す．この力を身につける事によるメリットが４つ挙げられる．１つ目は，問題解決能力の向上である．論理的思考力の要素である，問題を細分化し，整理し，分析する力を育むことにより，問題の本質の把握に繋がり，同様の問題を起こしづらくなる等，問題解決能力の向上が期待できる．２つ目は，プレゼン力や提案力の向上である．論理的思考力の要素である，筋道を立てて矛盾なく結論を出すことができる力を育むことにより，聞き手に対して説得力のある提案をすることができ，プレゼン力や提案力の向上が期待できる．３つ目は，情報の取捨選択能力の向上である．論理的思考力の要素である，様々な情報の中から自分に必要な情報を選んで，その情報に基づいて物事を考えることができる力を育むことにより，問題の本質や必要な情報のみを把握することができ，情報の取捨選択能力の向上が期待できる．４つ目は，コミュニケーション能力の向上である．論理的思考力の要素である，相手の意見を理解する力や自分の意見を伝える力を育むことにより，相手と自分の意見をすり合わせて適切な結論を導き出すことができ，コミュニケーション能力の向上が期待できる．以上より，論理的思考力を身につけることによって，様々なメリットが期待できる．

論理的思考をベースとした思考の一つに「プログラミング的思考」がある．プログラミング的思考は，小学校段階において 2020 年度から教育が必修化されるなど現代社会において特に注目されている要素の一つである．プログラミング的思考は「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と定義されている．ベネッセのサイト[2]によると，「プログラミング的思考」とは，コンピュータやプログラミングの概念に基づいた問題解決型の思考であり，例えば，コンピュータで処理する対象を細かくして実行しやすくしたり，それらの処理を実行するために繰り返しや条件分岐を用いたりする動きのことである．

ここで，「プログラミング的思考」の定義をもとに，ベネッセのサイト[2]により整理された，「プログラミング的思考」の４つの要素を以下に示す．

分解：大きな動きを解決可能な小さな動きに分けること．特に複雑な問題の場合には，解決できる小さな動きに分解して，問題を解決しやすくする．

抽象化：目的に応じて適切な側面・性質だけを取り出し，他の部分を捨てること．

一般化：物事の類似性や関係性を見出すこと．さらにそれを別の場合でも利用できる内容にすること．一般化することで予想しやすくなり，より汎用的に問題を解くことができる．

組合せ：目的に応じて試行錯誤しながら，明確でよりよい手順を創造すること．組合せ方法には「順次」，「繰り返し」，「条件分岐」などが含まれる．

2.2 平面図形問題を用いた学習システム

三島[4]は，平面図形問題を用いたプログラミング的思考育成システムを開発した．本研究では，図形の性質問題として使用している．この問題では主に角度入力画面，流れ作成画面を用いてプログラミング的思考の育成を図る．

まず，角度入力画面では，緑で表示された部分の全ての角度を求める．求めるために必要な図形の性質を選択することにより，適切な性質を取り出す動きである抽象化の育成を図る．角度入力画面の例として図形の性質問題２（角度入力画面）を図 2.1 に示す．

次に，流れ作成画面では，１つ目の画面で求めた角度等を用いて，必要な図形の性質をフローチャートに当てはめ，正解の角度を求める．回答までの流れを要素ごとに分けながら考える動きにより，大きな動きを解決可能な小さな動きに分ける動きである分解の育成，要素を繋げながら解答を導出する動きにより，物事の類似性や関係性を見出すことができ一般化の育成，この問題は最短路でなければ正解とならないことから，試行錯誤しながら，明確でよりよい手順を創造する組合せの育成が期待でき，角度入力画面と合わせてプログラミング的思考４要素全ての育成が期待できる．流れ作成画面の例として図形の性質問題２（流れ作成画面）を図 2.2 に示す．

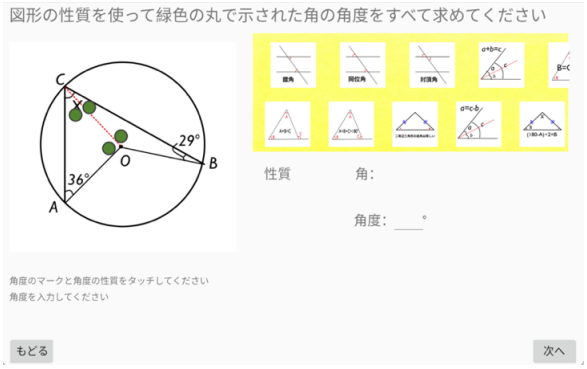

図 2.1：図形の性質問題２の画面（角度入力画面）例

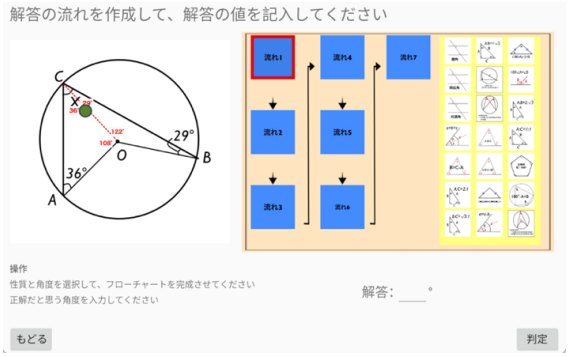

図 2.2：図形の性質問題２の画面（流れ作成画面）例

2.3 フローチャート作成課題を用いた学習システム

川本ら[7]は，フローチャート作成課題を用いた学習システム「ろっつ」を開発した．このシステムは問題の性質によって４つに分類されており，本研究では，文章の並び替え問題１，文章の並び替え問題２，車の動きを考える問題１，車の動きを考える問題２として使用している．

2.3.1 文章の並び替え問題１

文章の並び替え問題１はフローチャートを用いて文章の並び替えを行う問題である．色枠部分を長押ししながら動かすことにより，並び替えを行うことができる．本研究では，難易度を上げるため，文章の接続詞の部分を初めから取り出しておき，接続詞部分のみで判断することができないように改良を行なった．問題例として，文章の並び替え問題１の画面例を図 2.3 に示す．

2.3.2 文章の並び替え問題２

文章の並び替え問題２はフローチャートを用いて文章の要約を完成させる問題である．文章自体を読まずに，パーツだけを読みながら作成してしまうという問題を解消するため，問題提示画面で問題を読んだ後，フローチャート作成画面に遷移し，解答を行う方式に改良を行なった．また，プログラミング的思考育成向けとして考えると抽象化の部分が要素として欠けているという課題が生じていたため，パーツに誤選択肢を設け，適切な要素を取り出す操作を追加することにより，抽象化の要素の追加を行なった．問題例として，文章の並び替え問題２の画面例を図 2.4 に示す．

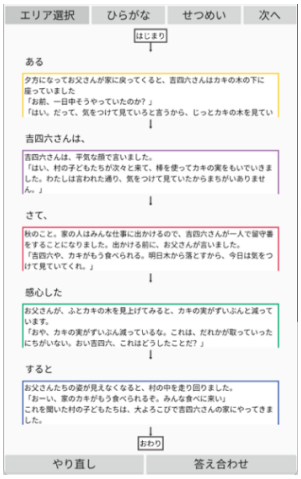

図 2.3：文章の並び替え問題１の画面例

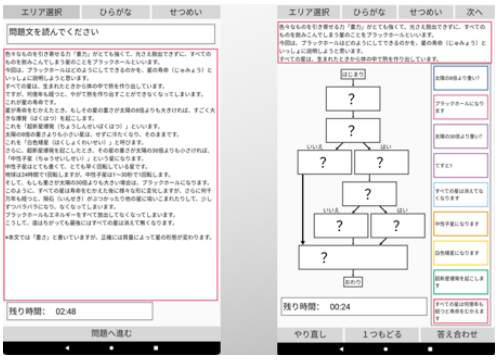

図 2.4：文章の並び替え問題２の画面例

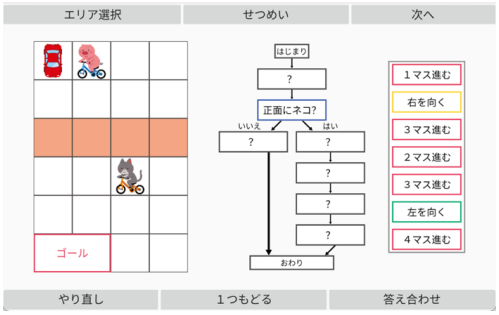

図 2.5：車の動きを考える問題１の画面例

2.3.3 車の動きを考える問題１

車の動きを考える問題１はフローチャートを用いて，スタートからゴールまでの車の動きのルートを考える問題である．この問題も，プログラミング的思考力育成向けとして考えると，抽象化の部分が要素として欠けているという課題が生じていたため，パーツに誤選択肢を設け，適切な要素を取り出す操作を追加することにより，抽象化の要素の追加を行なった．車の動きを考える問題１の画面例を図 2.5 に示す．

2.3.4 車の動きを考える問題２

車の動きを考える問題２は車の動きを考える問題１と逆の動きで，提示されたフローチャートを用いて，車が最後にどこで止まっているのかを考える問題である．問題例として，車の動きを考える問題２の画面例を図 2.6 に示す．

2.4 数論問題を用いた学習システム

川本ら[8]は，数論問題を用いた論理的思考力育成システ

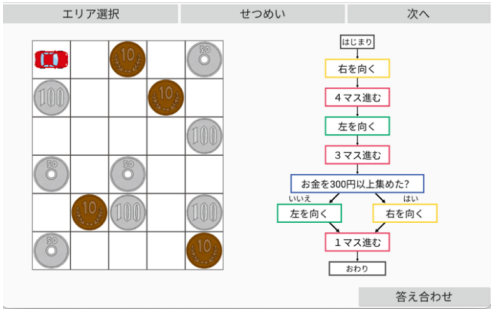

図 2.6：車の動きを考える問題２の画面例

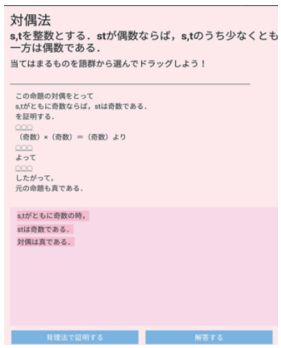

図 2.7：証明問題１の例

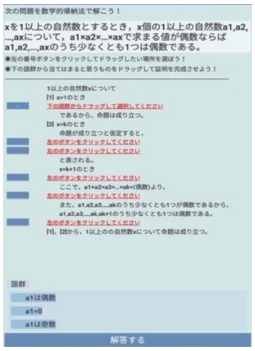

図 2.8：証明問題２の例

ムを開発した．このシステムは問題の性質によって２つに分類されており，本研究では，証明問題１，証明問題２として使用している．

証明問題１は前後の文の繋がりを理解させつつ，証明全体の論理的整合性を考えさせるために，パーツに分解された証明文を正しく組み立て直して再構成する方式を用いた証明問題である．具体的には，複数の空行がある証明が提示され，画面下部にある語群から適切な文を空行にドラッグ&ドロップして証明を作成する．完成された証明文をパーツに分解するという性質上誤選択肢は存在しない．証明問題１の画面の例を図 2.7 に示す．

証明問題２は前後の文の繋がりを理解させることに重点があるため，初めから証明の論理構造を与える方式を用いた証明問題である．具体的には，証明問題１と同様複数の空行がある証明が提示されるが，空行の先頭にある選択ボタンをタップすることで，語群として画面下部にその行に入れるべき文を含む３つの文が現れる．その語群から適切な文を空行にドラッグ&ドロップして証明を作成する．証明問題２の画面の例を図 2.8 に示す．

3. 提案システム

3.1 システム設計

育成効果の高い(劣っている)能力を主として高めるようなプラン自動提案機能を以下のように設計する．平面図形問題や数論問題等を用いてプログラミング的思考を育成するには，それらの問題に関する知識が必要となり，必要な知識が不足しているとプログラミング的思考を効果的に育成することが難しい．よって，平面図形問題や数論問題等に関する知識を調べる問題を事前テストとして解いてもらう．その結果を元に学習者の知識を把握し，効果的にプログラミング的思考を育成するための学習プランを作成する．学習者は提示されたプランに基づき課題に取り組むことで，4 要素に関する能力を偏りなく得ることができ，結果としてプログラミング的思考の能力を高められる．この設計方針の元，提案システムを難易度の設定，システムの改良を行った．

3.2 難易度の設定

これまでに開発した学習システムで使用した課題に関して，プログラミング的思考の要素ごとの難易度を次のように設定した．設定の流れは，２節で挙げたベネッセのサイト[2]における，プログラミング的思考の要素ごとの定義を用いて判定基準を５段階で設定し，課題の性質ごとに，「数論問題（数論）」，「文章の並び替え問題（文章）」，「車の動きを考える問題（車の動き）」，「図形の性質問題（図形）」に分けて設定した難易度を以下に示す．

3.2.1 分解

以下に設定した判定基準を示す．各問題の分解に関する難易度を表したレーダーチャートを図 3.1(左)に示す．

５：実際に小さな動きへの細分化のページがあり，また細分化するために思考を要する．

４：特に分解を行うための動きはないが，小さな動きを繋げていくような思考を要する．

３：小さな動きへの細分化が必要であるが，流れははっきりしておりあまり思考は必要でない．

２：動きを分解するような動作は存在せず，並び替えを行えば回答が導ける．

１：動きを分解するような動作は存在せず，並び替えを行えば回答が導ける．また，動作も短いためあまり思考は必要でない．

3.2.2 抽象化

以下に設定した判定基準を示す．各問題の抽象化に関する難易度を表したレーダーチャートを図 3.1(右)に示す．

５：適切な性質・側面を選択するような操作が存在し，実際に思考が必要である．

４：特に抽象化を行うための動きはないが，適切な性質・側面を選択するような思考を要する．

３：適切な性質・側面を選択するような操作が存在するが，流れははっきりしておりあまり思考は必要でない．

２：適切な性質・側面を選択するような操作が存在せず，並び替えを行えば回答が導ける．

１：性質・側面を選択するような操作は存在しない．また，動作も短いためあまり思考は必要でない．

3.2.3 一般化

以下に設定した判定基準を示す．各問題の一般化に関す

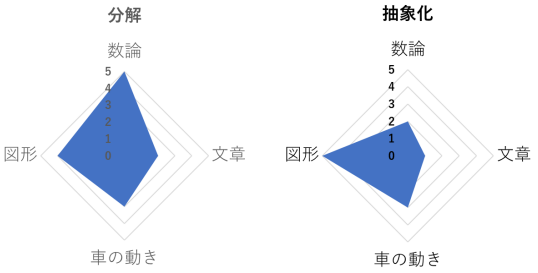

図 3.1：分解と抽象化に関する難易度

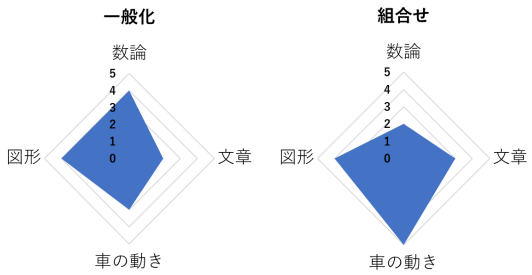

図 3.2：一般化と組合せに関する難易度

る難易度を表したレーダーチャートを図 3.2(左)に示す．

５：類似性や関連性を見出す操作が存在する．また，別の場面でも利用するような操作が存在する．

４：特に一般化を行うための動きはないが，類似性や関連性を見出すような思考を要する．

３：類似性や関連性を見出すような操作が存在するが，流れははっきりしておりあまり思考は必要でない．

２：類似性や関連性を見出すような操作が存在せず，並び替えを行えば回答が導ける．

１：類似性や関連性を見出すような操作は存在しない．また，動作も短いためあまり思考は必要でない．

3.2.4 組合せ

以下に設定した判定基準を示す．各問題の組合せに関する難易度を表したレーダーチャートを図 3.2(右)に示す

５：上記に上げる組み合わせ方法が複数含まれる．また，回答の作成に試行錯誤が必要である．

４：回答作成の流れは順次のみであるが，回答の作成に試行錯誤が必要である．

３：上記に上げる組み合わせ方法が複数含まれるが，流れははっきりしておりあまり試行錯誤は必要でない．

２：回答作成の流れは順次のみであり，並び替えを行えば回答が導ける．

１：回答作成の流れは順次のみであり，動作も短いためあまり試行錯誤は必要でない．

3.3 システムの改良

プログラミング的思考の育成を目的としているが，２節で挙げたシステムは論理的思考力育成向けに作られたシステムであり，プログラミング的思考育成向けとして考えると，４要素を満たしているシステムは少ないのが現状である．そこで，プラン案を作成する前にシステムの改良を行った．文章の並び替え問題２の改良前の問題を図 3.3 に示す．この問題は，誤選択肢がないため，適切な性質を取り出す動きが存在せず，２節で示したプログラミング的思考の要素である，「抽象化」の育成が期待しづらいという課題があった．また，この問題は小学生を対象に作成されているため，本研究での学習者向けに考えると簡単すぎるという問題があった．そこで，これらを解消するためにシステムに誤選択肢を設け，適切な性質を取り出す動きを追加し，

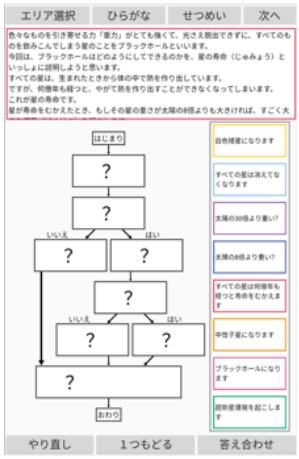

図 3.3：改良前の問題

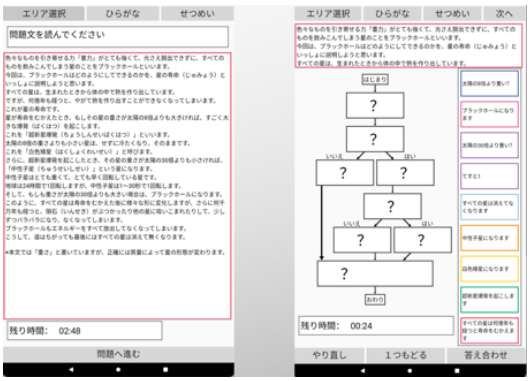

図 3.4：修正後の問題

問題に制限時間を設け，文章を先に読んでフローチャートを作成する方式にレイアウトを変更し，難易度の修正を行った．改良後の問題を図 3.4 に示す．

3.4 システム構成

作成したシステムの流れ（図 3.5 参照）をもとに，各画面の役割や GUI について以下に説明する．

(1) 教授者ログイン画面（図 3.6 参照）は，教授者が ID とパスワードを用いてログインを行う画面である．ログイン時にデータベースを用いて，学習者のステータスが初回のログインなのか，過去にプラン作成を行なったことがあるかを判定し，初回ログインであれば，事前テスト処理画面に遷移する．過去にプラン作成を行なっていた場合，ステータスに応じて，問題作成画面や問題一覧画面に遷移する．

(2) 事前テスト処理画面（図 3.7 参照）では，事前テストの得点を入力することで，3.2 節で決定した難易

図 3.5　提案システムの概要図

度に基づいて，プログラミング的思考の要素ごとの得点に換算され，データベースに換算得点を保持するとともに，問題作成画面に遷移する．

(3) 問題作成画面（図 3.8 参照）では，事前テストや過去のプランの得点によって換算されたプログラミング的思考の要素ごとの得点を用いて学習プランが自動作成される．加えて，学習が期待される力を表すレーダーチャートが同時に作成される．レーダーチャートの青枠部分は学習前の事前テストや過去のプランの換算得点を表し，緑枠部分は，自動作成されたプランの問題について，学習が期待できる力を分析し表示している．赤枠部分は，学習の最終目標を表している．また，難易度ボタンや修正ボタンによって問題の変更を行うことができる．難易度ボタンは，赤文字が現在の問題の難易度を表し，黒文字のボタンを押すことで，その難易度の問題に変更される．修正ボタンでは，手動作成ボタンに遷移し，異なる種類の問題に変更することができる．

(4) 手動作成画面（図 3.9 参照）では，教授者が独自に問題を選定し，プランの作成を行う．選択された問題はプランに反映されると共に，レーダーチャートも修正される仕組みとなっている．

(5) 問題一覧画面（図 3.10 参照）では，学習者が実際に問題を解くための画面である．解き終わった問題はグレー，次の問題は赤文字で表示される仕組みとなっている．

(6) 解答状況画面（図 3.11 参照）では，問題一覧画面で学習した学習データの分析が行われる．まず，過去２回分の学習結果を分析したレーダーチャートが作成される．レーダーチャートの要素には，「分解」，「抽象化」，「一般化」，「組合せ」の要素ごとの換算得点に加えて，「得点」，「難易度」があり，学習者のプログラミング的思考における弱点に加えて，プラン自体の難易度，得点を把握することができる．加えて，前回データにおいては，「文章の並び替え問題」，「数学問題」，「グラフ問題」に分けた正答率，プログラミング的思考における弱点分析が行われ，緑で強調された部分に学習結果として表示される．また，学習データを用いて次回の学習プランが自動作成され，紫で強調された部分に表示される．

4. 評価実験

提案システムを，開発環境 Android Studio version 4.2.1 の言語 Kotlin version1.5.0 を用いて，Android タブレット端末上に実装し，提案システムの有用性を示す評価実験を行った．実験の流れを以下に示す．

4.1 実験協力者

実験協力者は，広島市立大学の情報科学部生・大学院情報科学研究科生 12 名である．まず，協力者に対して事前アンケート・事前テストを行い，事前テストの得点が均等に

図 3.6：教授者ログイン画面

図 3.7：事前テスト処理画面

図 3.8：問題作成画面

図 3.9：手動作成画面

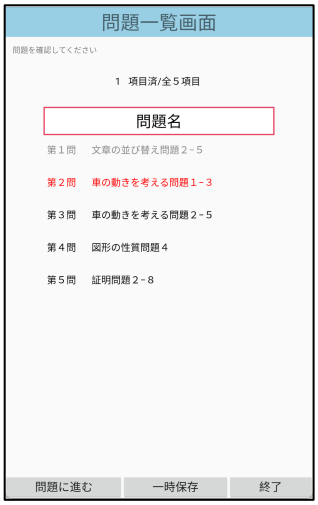

図 3.10：問題一覧画面

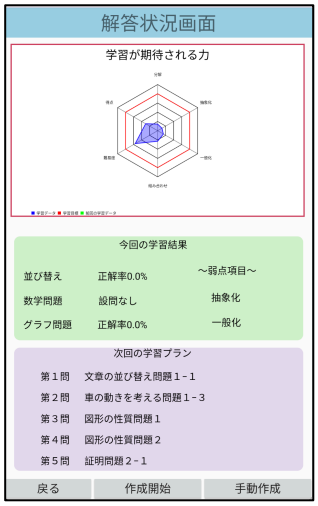

図 3.11：解答状況画面

表 4.1：事前テストの得点の平均(t 検定)

	データ数	平均	標準偏差	t 検定(両側)
提案群	6	30.33	7.91	t(9)=0.1726
比較群	6	31.17	7.47	.10<p

なるように提案群 6 名と比較群 6 名の 2 群に分けた．12 名の事前テストの得点の平均に対して t 検定を行なった結果を表 4.1 に示す．この結果，提案群と比較群は均質であるといえる．

4.2 実施期間・時間

以下の 2 日間で実験を実施した．

第 1 日程：2021 年 12 月 16 日～12 月 20 日のいずれか 1 日
第 2 日程：2021 年 12 月 21 日～12 月 23 日のいずれか 1 日

実験実施時間を以下に示す．なお，以下のように実施目安時間を設定したが，協力者が設定時間内に解答を終え筆記用具を置いた場合は解答を終了したとみなし，解答時間を短縮した．

事前アンケート，事前テスト　　：50 分程度
提案システムまたは画一的なプランによる学習：40 分程度
事後アンケート，事後テスト　　：50 分程度
もう一方の群で用いた学習，比較アンケート　：40 分程度

4.3 学習方法

提案群には，本論文の提案システムを用いて学習してもらった．比較群には，画一的な学習プランを用いて学習してもらった．これは，3.2 章で設定した難易度については普通，問題は「数論問題」，「文章の並び替え問題」，「車の動きを考える問題」，「図形の性質問題」を満遍なく学習できるようなプランを設定した．

4.4 実験手順

実験は第 1 日程，第 2 日程の 2 日間に分けて行なった．まず，第 1 日程で，(1)事前アンケート・事前テストを行った．次に，第 2 日程で，(2)提案群は提案システムを，比較群は画一的なプランを用いた学習，(3)事後アンケート，事後テスト，(4)提案群は画一的なプランを，比較群は提案システムを用いた学習後，比較アンケートの順で行った．

事前アンケートは，協力者の特性を測ることを目的として行った．実施目安時間は 5 分程度とした．また，アンケートは両群同様のものを使用した．

事前テストは，協力者の学習前の能力を測ることを目的として行った．テスト問題として，問題を 5 問出題した．実施目安時間は 50 分程度とした．また，テストは両群同様のものを使用した．得点配分は，文章問題，グラフ問題，数論問題各 20 点ずつである．

事前アンケート・事前テストを行なった後，提案群には提案システムを，比較群には画一的なプランを用いて学習を行ってもらった．なお，提案群の学習に使用したシステムの概要は第 3 章，比較群の学習に使用した画一的なプランの詳細は第 4 章を参照する．実施目安時間は両群とも 40 分程度とした．

事後アンケートは，協力者の学習を評価することを目的として行った．実施目安時間は 10 分程度とした．なお，提案群は提案システムとプランに含まれた問題についての評価，比較群は画一的なプランと含まれた問題についての評価を行なった．

事後テストは，両群の学習効果を比較することを目的として行なった．テスト問題として，事前テストと同様の問題を 5 問出題した．実施目安時間は 50 分程度とした．

比較アンケートは，提案群に画一的なプランを，比較群に提案システムを使用後，提案システムと画一的なプランの比較を目的として行なった．実施目安時間は 5 分程度とした．また，アンケートは両群同様のものを使用した．

5. 結果と考察

5.1 事前テスト・事後テストの点数比較と考察

提案群・比較群(要因 A)に対する事前テスト・事後テスト(要因 B)の結果を表 5.1 に示す．事後テストの総合点の平均について，t 検定を行なった結果を表 5.2 に示す．

表 5.2 より，事後テストの総合点の平均において提案群と比較群の間に有意差は見られなかった．

事後テストは「文章問題」，「数論問題」，「グラフ問題」の 3 つの問題を組み合わせて出題した．文章問題の得点の平均において t 検定を行なった結果を表 5.3 に示す．数論問題の得点の平均において t 検定を行なった結果を表 5.4 に示す．グラフ問題の得点の平均について，t 検定を行なっ

表 5.1：テスト結果

要因 A	要因 B	データ数	平均	標準偏差
提案群	事前テスト	6	30.3333	7.9092
提案群	事後テスト	6	41.6667	5.4975
比較群	事前テスト	6	31.1667	7.4703
比較群	事後テスト	6	35.6667	5.5277

表 5.2：事後テストの総合点の平均(t 検定)

	データ数	平均	標準偏差	t 検定(両側)
提案群	6	41.67	5.5	t(9)=1.7209
比較群	6	35.67	5.53	.10<p

表 5.3：事後テストの文章問題の得点の平均(t 検定)

	データ数	平均	標準偏差	t 検定(両側)
提案群	6	9.83	3.48	t(9)=1.7211
比較群	6	6.17	3.24	.10<p

表 5.4：事後テストの数論問題の得点の平均(t 検定)

	データ数	平均	標準偏差	t 検定(両側)
提案群	6	17	4.28	t(9)=0.2639
比較群	6	17.67	3.73	.10<p

表 5.5：事後テストのグラフ問題の得点の平均(t 検定)

	データ数	平均	標準偏差	t 検定(両側)
提案群	6	14.83	1.07	t(6)=2.4317
比較群	6	11.83	2.54	.05<p<.10

た結果を表 5.5 に示す．表 5.3，表 5.4 より，文章問題，数論問題の得点の平均について提案群と比較群の間に有意差は見られなかった．表 5.5 より，グラフ問題の得点の平均について提案群は比較群と比べて有意に得点が高い傾向があると言える(t(6)=2.4317, .05<p<.10)．グラフ問題は，プログラミング的思考の 4 要素をバランスよく用いながら解答する事が求められる問題である．よって，提案システムはプログラミング的思考の育成に有効である可能性があることが示唆された．

5.2 比較アンケートの結果と考察

比較アンケートは，提案システムを用いた学習(A)と画一的なプランを用いた学習(B)を比較する質問により構成される．1・2・3 を提案システムの方が当てはまるという意見，4・5・6 を画一的なプランの方が当てはまるという意見として該当する人数を集計し，二項検定を行なった．提案群の結果を表 5.6 に，比較群の結果を表 5.7 に示す．なお，アンケートは両群同様のものを使用した．

比較アンケートの結果，質問番号 2, 4, 6 については両群と自動作成されたプランを選択した人が有意に多かった．この結果から，学習者は自動作成されたプランの方がプログラミング的思考が身に付きやすく，学習しやすいと感じていたと考えられる．

6. 結論

本研究では，効果的にプログラミング的思考を育成するための学習プランを自動提案する機能を持った学習システムを開発した．評価実験により，提案システムがプログラミング的思考の育成に有用である可能性があることが示唆された．また，評価アンケートを行い，プランの妥当性を確認した．今後の課題として，システムの弱点提示方法の改善やプラン選定理由の表示，問題部分の GUI の統一，チュートリアル問題の作成などが挙げられる．

謝辞

本研究は科研費(19K03059)の助成を受けたものである．

表 5.6：学習を比較する質問の結果（提案群）

	質 問	A	B	二項検定
1	どちらの方がより深い思考を必要とすると思いますか.	4	2	n.s.
2	どちらの方がよりプログラミング的思考が身につくと思いますか.	6	0	p<.05
3	どちらの方がより勉強したと感じると思いますか.	5	1	n.s.
4	どちらの方がより学習しやすいと思いますか.	6	0	p<.05
5	どちらの方が，内容を理解するのが難しいと思いますか.	3	3	n.s.
6	どちらの方が，アクティブに学習すると思いますか.	6	0	p<.05

表 5.7：学習を比較する質問の結果（比較群）

	質 問	A	B	二項検定
1	どちらの方がより深い思考を必要とすると思いますか.	6	0	p<.05
2	どちらの方がよりプログラミング的思考が身につくと思いますか.	6	0	p<.05
3	どちらの方がより勉強したと感じると思いますか.	6	0	p<.05
4	どちらの方がより学習しやすいと思いますか.	6	0	p<.05
5	どちらの方が，内容を理解するのが難しいと思いますか.	5	1	n.s.
6	どちらの方が，アクティブに学習すると思いますか.	6	0	p<.05

参考文献

［1］ 文部科学省，“小学校プログラミング教育の手引（第一版）”（平成 30 年 3 月）．https://www.mext.go.jp/content/20200214-mxt_jogai02-000004962_004.pdf，(参照 2022-1-27).
[2] ベネッセ教育情報サイト，“図で解説「プログラミング的思考」とは”．https://benesse.jp/programming/beneprog/2018/07/13/computationalthinking/，(参照 2022-1-27).
[3] 文部科学省，“【総則編】小学校学習指導要領（平成 29 年告示）解説”．https://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/03/18/1387017_001.pdf，(参照 2022-1-27).
[4] 三島里佳．平面図形問題を用いたプログラミング的思考に基づく論理的思考力育成支援システム．2020 年度広島市立大学情報科学部卒業論文，2021.
[5] 井上尚美．言語論理教育入門．明治図書，1989.
[6] 科学コミニュケーション研究所．“科学の技術の智プロジェクト　専門部会報告書「数理科学」第 4 章”．http://literacy-report.scri.co.jp/wp-content/uploads/2018/12/01_数理科学専門部会報告書（訂正版）.pdf，(参照 2022-1-27)．
[7] 川本佳代，出口直輝，林雄介，平嶋宗，砂山渡．論理的思考力育成を指向したフローチャート活用学習システムと小学校児童による実験的評価．教育システム情報学会誌，2015，Vol .32，pp. 214-219.
[8] 川本佳代，古谷美夏，宮脇綾子，内田智之，平嶋宗，林雄介．数学証明問題を用いた論理的思考力育成システムの開発．2018 年度人工知能学会全国大会（第 32 回），2018，1L301.
[9] 川本佳代，佐々木崇大，内田智之，林雄介，平嶋宗．平面図形問題を用いた論理的思考力育成支援システムの開発．人工知能学会先進的学習科学と工学研究会（第 86 回），2019，pp. 18-23.