芦屋大学論叢　第79号
（令和5年7月29日）抜刷

技術科のプログラミング学習における授業形態の一考察

小　澤　雄　生
安　東　茂　樹

技術科のプログラミング学習における授業形態の一考察

小　澤　雄　生（1）
安　東　茂　樹（2）
（1）芦屋大学大学院教育学研究科博士後期課程
（2）芦屋大学経営教育学部特任教授

1．はじめに

世界情勢は，2023 年度を境に Covid-19 新型コロナウィルス（以下，コロナ）対応の社会から元の生活に移行し始めている。我が国では，同年 5 月 8 日にコロナの感染症分類が「5 類」に引き下げられた。そして，コロナの前まで当然であった，世界の国々や地域間の人々の移動が，国内でも可能になった。

コロナの感染の流行は，人の往来ができない状態であった。その間は，情報通信の重要性が増し，オンラインでのやりとりや在宅ワークなど，これまでにない変化を遂げた。この情報通信等の社会は，様々な要因で急激に深化している。例えば，2023 年 5 月 11 日，米国 Google 社が次世代 LLM「PaLM 2（Pathways Language Model）」を発表した。Google は，AI の分野で遅れをとっていたが，世の中の流れなどから力を入れ始めた。そして，Google の対話型 AI「Bard」も PaLM 2 ベースとなり，日本語も対応するようになった。この大規模言語モデル「PaLM」は，プログラミングに関わる新たな技術として，指示に対してコードの生成やコードのバグ修正も簡単な点があげられている。また，2022 年 11 月に公開された Chat GPT は，高度な AI 技術によって，人間のように自然な会話ができる AI チャットサービスであり，無料で利用できる革新的なサービスとして注目を集め，生成した文章の程度や回答の内容が大きな話題となっている。さらに，2023 年 3 月に GPT-4（Generative Pre-trained Transformer 4）が発表された。これは，米国の非営利団体の Open AI が開発した AI 言語モデル（マルチモーダル）の大規模言語モデルである。以前の GPT-3 と比べて，幅広い一般知識と問題解決能力，画像を読み取りやその内容を説明できる次世代の AI 言語モデルである。これは，Chat GPT と比べて，日本語に対する精度も正確性が高いと言われている。さらに，米国 Microsoft 社は，2023 年 5 月に Windows 11 に対話型人工知能（AI）「Copilot」を搭載すると発表した。このように，AI や GPT などの情報技術は，日々加速度的に進化している。

一方，日本のプログラミング教育は，小学校で 2020 年度に開始され，2021 年度に中学校でも「ネットワークを利用した双方向性のあるコンテンツのプログラミング」という新しい内容を加えて開始された。しかし，他国と比較すると開始年度の遅れや学習内容の程度など，情報技術の進歩に対応できる十分な教育内容であるか判断に戸惑う現状がある。そのため，子どもの情報活用能力の育成は，情報技術の進化に伴い日々進化していく必要がある。

2．研究の目的

2.1 これまでの実践内容

プログラミング教育の実践は，A 義務教育学校において，小学校 3 年生からプログラミング学習を開始し，小学校から中学校へ系統的なカリキュラムを構築し改良を重ねてきた。実践を開始した当初の頃は，様々なライントレースカーや scratch などのビジュアルプログラミングを実践していた。

現在では，小学校の中学年でロボホン（SHARP 社のプログラミングロボット）を活用して，ビジュアルプログラミングでプログラミングの基礎を学ばせている。学習内容は，順次・分岐・反復などのプログラミングの基礎から変数までを取り扱い，総合的な学習や社会科などとの教科横断学習を推し進めながら，日常生活と関連付けて学習を進めている。小学校の高学年では，ビジュアルプログラミングから簡易言語である coffee script 言語を使用して，for ループや関数などプログラミングで必要な技能的な部分を繰り返し学習している。中学生では，これらの学習を踏まえて，Python 言語に移行している。Python 言語を用いて，高学年時に学習した内容を復習する内容とフローチャートを利用して簡単なアルゴリズムを学習している。これらの取り組みは，年度を要するごとに指導者側の指導技術の向上によって，教える進度が速くなり中学校の卒業までにどのレベルまで学習できるかなど，ねらいを向上しつつ具現化している。今後は，Python 言語を使用して，自分で作りたい簡単なプログラムを作成するところまでを目的としている。

2.2 背景と目的

文部科学省（以下，文科省）は，小学校プログラミング教育の手引き[1)]で小学校段階のプログラミングに関する A〜F の学習活動に分類を示した。これまでの研究では，それら様々な実践方法等を特設科目の“技術科”（文部科学省の研究開発指定内容）としてまとめ，プログラミング的思考の育成と本来のプログラミングを学習しながら，系統的なカリキュラムの開発を行ってきた。しかし，プログラミング学習は，積み上げ型の学習であり，学年が上がるごとに理解や進度の差が生じた。加えて，論理的な考え方ができる年齢は，個々人によって差がある。それは，小学生にプログラミングを教えていた担任や技術科教員，プログラミングに長けている教員等が指導した過程で，質問内容や見取り，できあがったプログラムや授業の振り返りなどから，小学校中学年で論理的思考ができている児童もいれば，中学生になってもできない生徒も見られた。この問題は，小学生からプログラミングを学習してきたことで，中学生になり論理的思考ができない生徒の数は年々減少しその解決に向かっている。しかし，プログラミング学習は，論理的思考を効率よく使えるか，その考え方に慣れているかなどによって進度に差が生じた。そのため，プログラミングが得意な生徒は，授業で様々な方法を考えたり，過去の問題に取り組んだりと理解を深める学習をしているが，次の課題に進む前に，どうしても学習者全員の理解度の具合いによって学びを待たせる時間が生じた。

本研究では，それら児童・生徒がどのような学び方を求めているかを調査し，「個別最適な学び」となるにはどのような授業の進め方が最適か模索することが主たる目的である。

3．調査の結果と分析

3.1 調査について

調査は，プログラミングの基礎を学んだ小学校 6 年生（男子 45 名　女子 42 名）計 87 名と，中学校 1 年生（男子 39 名　女子 41 名）計 80 名を対象としてアンケートを行った。実施時期は，2023 年 3 月である。授業実践は，プログラミング的思考の重点化ではなく，特設科目の“技術科”としての本質的なプログラミング学習を行っている児童・生徒を対象として，プログラミングの学習活動を問題・課題解決的な学習活動として行った。プログラミング学習は，ある程度自分で進められるようになってきた状況で，内容を試案して少しずつ改良しながら実施してきた。

調査項目は，1 年間の授業の振り返りを記述式で行った。また，次の質問項目も作成した。まず，授業では，どの程度考えながら進めているかの指標とするために他教科と比較する質問を実施した。次に，説明する時間，自分で考える時間，周りの人と協力する時間に分けて進めた。その中で，それぞれの活動をどの程度の割合で進めた方が良いかの質問項目とその理由を記述式で作成した。

調査の分析は，テキストマイニングの KH Coder を用いた。KH Coder は，樋口[2)3)]が作成し，公開している日本語テキスト型データ分析のシステムである。これは，単語などの選択にあたり恣意的となり得る「手作業」をなくし，多変量解析によってデータ全体を要約して提示することができる。また，コーディング規則を公開するという手順を踏むことで，操作化における自由と客観性の両立が可能となる。

3.2 小学校 6 年生の分析と結果

3.2.1　振り返りの自由記述分析

本研究では，1 年間の授業の振り返りとして自由記述を行い，頻出語を確認した上で，それらの語の共起関係を探った。

児童から得られた 87 件の自由記述データを分析対象とした。KH Coder を用いて前処理を実行し，文章の単純集計を行った結果，112 の段落，233 の文が確認された。また，総抽出語数は（分析対象ファイルに含まれているすべての語の延べ数）は 5,989，異なり語数（何種類の語が含まれていたかを示す数）は，739 であった。さらに，助詞や助動詞などどのような文章にでもあらわれる一般的な語が除外され，分析に使用される語として 2,198（異なり語数 532）が抽出された。これらの頻出語の内の上位 20 語とその出現頻度を表 1（但し，未知語の意味をなさない語は除いた）に示す。

表 1　1年間の振り返りの自由記述における頻出語（小学校 6 年生）

順位	語	頻度	順位	語	頻度
1	思う	70	11	解ける	21
2	考える	68	12	少し	20
3	解く	50	13	コード	19
4	問題	50	14	理解	18
5	プログラミング	46	15	教える	17
6	力	43	16	多い	17
7	自分	32	17	意味	16
8	難しい	26	18	スキル	15
9	最初	23	19	人	15
10	使う	22	20	分かる	15

3.2.2 語の共起関係と自由記述の要約

KH Coder の「共起ネットワーク」のコマンドを用い，自由記述の出現パターンの似通った語を線で結んだネットワークを描いた（図 1）。また，分析にあたっては，出現数による語の取捨選択に関して最小出現数を 5 に設定し，描画する共起関係の絞り込みにおいて描画数を 60 に設定した。

図 1 の左側から，記述が多い点を相対的に見ると「考える力と問題を解く力は 5 年生の頃よりも伸びた」，「難しくなっていくけど、それに合わせて頭を使って考えていくので考える力がついたと思った。」などの考える力や問題を解く力，プログラミングに関わる力がついたという記述の多いことが見てとれた。また，「友達が分からないところとかも，教えたりすることが出来た。解く力は 1 年の最初に比べたら、1 つ 1 つすぐ出来るようになった。」などのように，最初と比較して伸びた・できるようになったという記述や教えることで自分もわかったなどの記述も数多く見られた。図 1 の中央下に，苦手と得意という言葉が線で結ばれているが，「はじめは苦手だった」から「得意になった」などの肯定的な記述に変わっていることも見られた。教え合いや協力する場面でも肯定的な意見がほとんどであり，全体的にプラスに変わり，嫌いになったや苦手になったという記述は見られなかった。なお，振り返り「　」内は，生徒記述の原文（以下，同様）である。

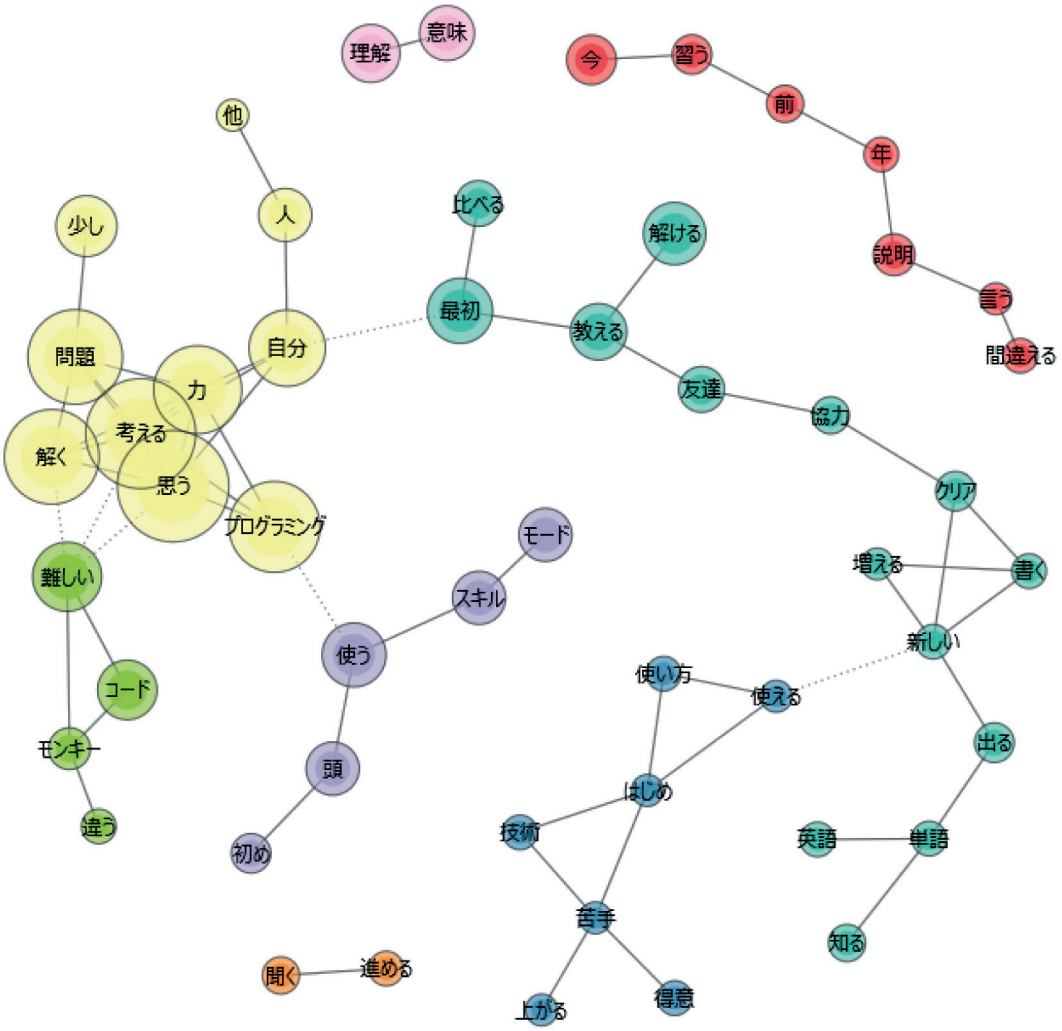

図 1　1年間の振り返りの自由記述の共起ネットワーク（小学校 6 年生）

3. 2. 3 他教科との比較

プログラミング学習では，パソコンを使用しているため，理解していなくても偶然正解することや何度か間違えても繰り返し入力していて正解になることがある。そのため，理解した気になり，あとになって理解できていない状況が多々あった。さらに，他者との教え合いの中で，答えを聞いてしまい考えない状態や理解できていない状態で進んでしまうなど，他教科にはあまり見られない問題が数多くあった。これらから，「プログラミング学習は，自分で考えるや考えなければいけないという行為が他教科と比べてどうか」について調査を行った。表 2 は，自分で考える・考えなければいけない時間が多い順に選択させた結果である。特設科目の技術は，プログラミングのみを行っている。結果は，他教科と比べて，比較的「考える・考えなければいけない」教科であることがわかる。図 2 は，上位 3 教科の順位をつけた児童の人数分布である。算数は, 1 番に選んだ児童が多いが，比較的分散している。理科と技術（プログラミング）は，同じ傾向を示している。選択理由は，記述がないため判別できないが，算数で下位に選択しているものに，得意な児童も含まれていると推察される。

表 2　1 時間の授業で考える時間が多い順の調査結果（小学校 6 年生）

	国語	算数	理科	社会	英語	音楽	美術	体育	技術	家庭科
平均値	4.40	3.66	2.98	5.26	6.13	7.85	6.63	7.53	3.29	7.28
最頻値	4	1	2	5	6	9	8	10	2	7

※順位選択であるため数値が低い方が上位である(以下同様)

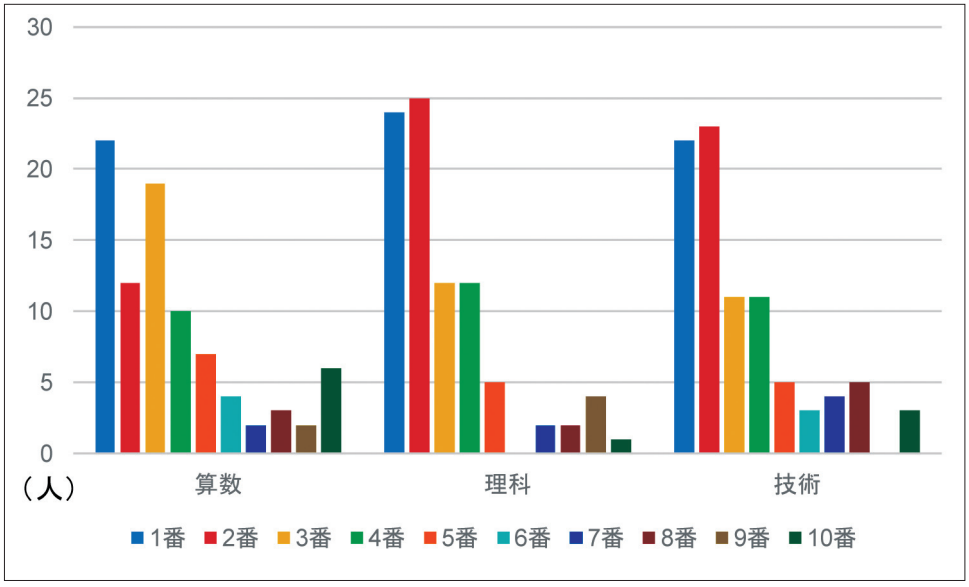

図 2　表 2 の上位分布比較

3. 2. 4　毎時間の授業形態の調査

授業の方法は，これまで様々な授業形態を試してきた。小学校のプログラミング教育が開始される以前は，中学校学習指導要領の指導内容としてプログラミングを行ってきた。内容と方法は，ライントレースカーやブレッドボードを使い，紙に黒く引いた線の上を走らせるプログラムや信号機のプログラムなどを学習させてきた。そして，授業方法は，教えて試す（行う）方法が基本であった。この方法は，時間数の問題もあり，系統的な内容というより一通り経験するというもので，自分で考える範囲もわずかであった。また，ビジュ

アルプログラミングからプログラミング言語に移行しても教えるということが大半で，児童がプログラミング的思考やプログラミングの基礎を学ぶということにほど遠い現状であった。

小学校プログラミング教育が開始されてからは，時間の幅が広がり，様々な方法を試すことができた。それにより，小学校 6 年生までに，ある程度の基礎的な内容が身に付けられるようになった。授業の形態としては，小学校の中学年で，できるだけ日常生活や他教科の知識と絡めながらプログラミングに触れてきた。そして，実際に，①説明を聞く時間，②自分で進める（考える）時間，③周りと協力する時間に分けて進めている。しかし，４年間学習を積み上げてきた児童にとって，それぞれの課題や理解の差から「個別最適な学び」になっていないと概観された。そのため，児童にどのように授業を進める方が良いかの調査を行った。調査項目は，「授業で①説明を聞く・②自分で進める・③周りと協力するの進め方の割合はどれくらいが良いか」で，その結果を図 3 に示す。

調査の結果，指導者の説明時間が 50%以上欲しいという児童は，一人もいなかった。また，友達と協力して進める時間も 30%未満を希望した児童が大半を占めた。結果から明らかになったことは，半分の時間を自分で進めることを希望している。また，残りの時間を指導者の説明と周りとの協力を希望している。若干だが，指導者の説明割合が少ない。その理由を書く記述では，「まずは自分でしっかり考えること」を書いている児童が多くを占めている。その中で，「どうしてもの時だけ周りの人の意見を聞く」が多くあった。ただ，「解答や解き方は教えて欲しくない。考え方だけ聞きたい。」という記述も多くあった。逆の意見では，「先生が教えてくれないことには，考えて時間が終わるだけでは生産性がない」という「まずは教えて欲しい」という記述もあった。

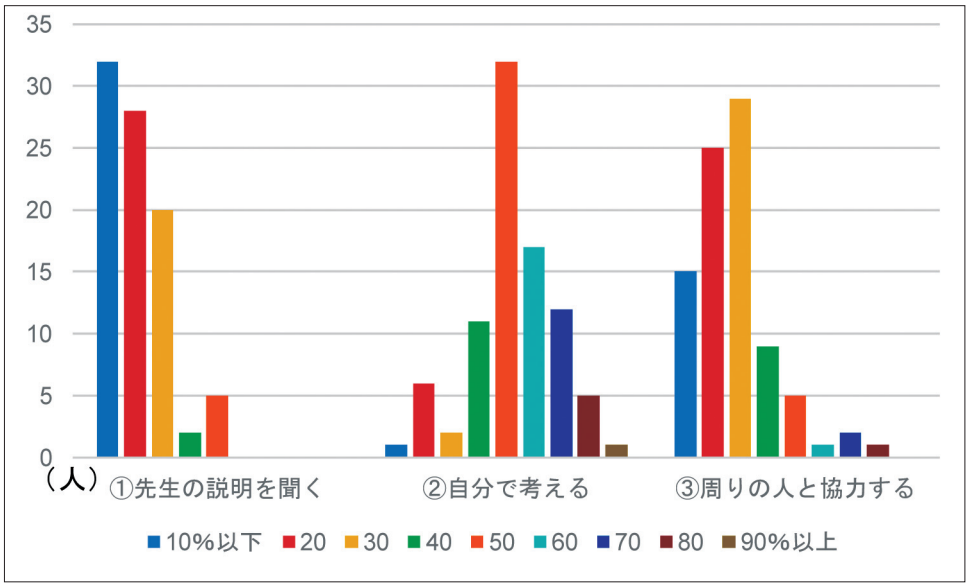

図 3　授業の進め方の割合

3.3 中学校1年生の分析と結果

3.3.1 振り返りの自由記述分析

小学校6年生と同様に中学1年生を対象に，1年間の授業の振り返りとして自由記述を行い，頻出語を確認した上で，それらの語の共起関係を探った。

生徒から得られた80件の自由記述データを分析対象とした。KH Coderを用いて前処理を実行し，文章の単純集計を行った結果，96の段落，177の文が確認された。また，総抽出語数は（分析対象ファイルに含まれているすべての語の延べ数）は5,233，異なり語数（何種類の語が含まれていたかを示す数）は，699であった。さらに，助詞や助動詞などどのような文章にでもあらわれる一般的な語が除外され，分析に使用される語として1,933（異なり語数503）が抽出された。これらの頻出語の内の上位20語とその出現頻度を表3（但し,未知語の意味をなさない語は除いた）に示す。

表3 1年間の振り返りの自由記述における頻出語（中学1年生）

順位	語	頻度	順位	語	頻度
1	思う	57	11	前	20
2	プログラミング	54	12	難しい	20
3	考える	50	13	理解	20
4	自分	42	14	意味	15
5	問題	39	15	聞く	15
6	解く	28	16	見る	13
7	使う	25	17	最初	13
8	力	23	18	出る	13
9	コード	21	19	ステージ	12
10	解ける	20	20	覚える	12

3.3.2 語の共起関係と自由記述の要約

KH Coderの「共起ネットワーク」のコマンドを用い，自由記述の出現パターンの似通った語を線で結んだネットワークを描いた（図4）。また，分析にあたっては，出現数による語の取捨選択に関して最小出現数を5に設定し，描画する共起関係の絞り込みにおいて描画数を60に設定した。

図4の中央の記述が多い点を相対的に見ると「プログラミングの基礎的なことや，ここをこうするとうまくいくか？という考える力がついたと思います。」，「どれだけ簡潔にプログラムを作って動かせるかと考えるのが楽しかった。」などの考える力がついたという記述が数多く書かれていた。また，「最初はforやindexを記号のように捉えていて意味を理解せず使っていたけれど，その意味を調べたり過去の問題を見て使い方を確認することによって，本質を理解して解くことができた。」など，プログラミングの基礎的な理解が深まったという記述も数多くあった。さらに，「自分でやり抜いたことで，その他の数学の問題でも諦めないで，問題を自分で解くことができた。」，「プログラミングをしたことによって，今まで諦めて，他の人に教えてもらっていたところが自分で諦めないで，解けることができるようになった。」などの記述も見られた。

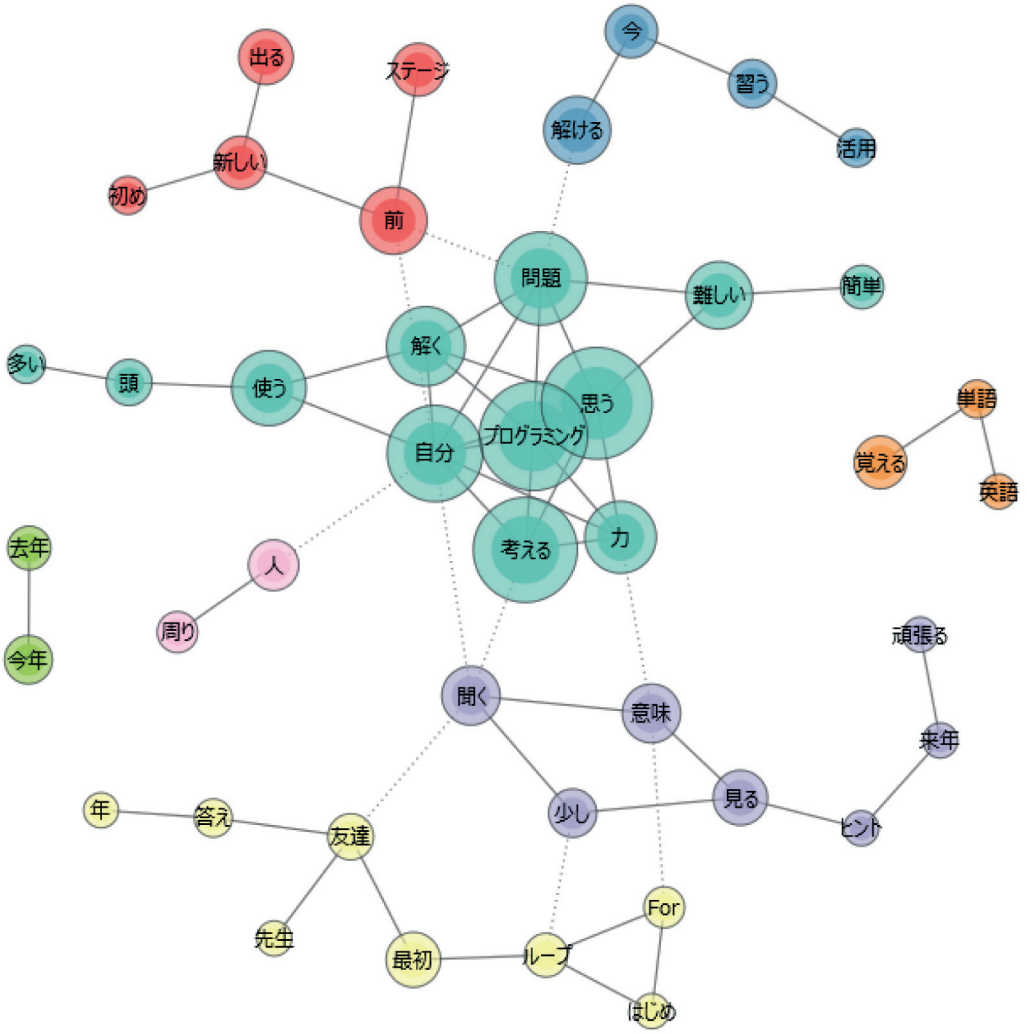

図 4　1年間の振り返りの自由記述の共起ネットワーク（中学校 1 年生）

3. 3. 3 他教科との比較

中学生になると，小学生とは違い理解できていない場合，きちんと理解しようとする生徒が多くなった。そのため，できていても何度かやり直すことや以前のコードを見て理解しようとする姿が見られた。この点から，自分が納得して問題を解きたいという姿から，6 年生より一段と「自分で考えたい」という内容の深みや重みが備わり，違う傾向が概観された。そのため，中学 1 年生は，6 年生よりさらに「個別最適な学び」から離れているように考えられた。他者との教え合いでは，自分の疑問を周りの人に聞くという傾向があり，「教え合いの中で双方の力が上がる」という記述が見られた。6 年生時の「教える人の力がつき，教えてもらう側は答えがわかるだけで損をする。」という記述から，考え方が変化した回答が複数あった。

中学生では，授業形態をどのように考えているか，6 年生と同様に他教科と比較した。技術科は，プログラミングの授業のみで回答するように伝えた。結果は，他教科と比べて，比較的考えなければいけない教科であることがわかった（表 4）。また，図 5 は，社会科を除いた上位 3 教科の順位をつけた児童の人数分布である。社会科を除いた理由は，考えることを重視した取り組みを，年間を通して意図的に行っていたためである。傾向として，技術科を 1 番に選んだ生徒も多いが，6 年生よりは比較的分散している。数学科と理

科は，中学生になると新しい単元や学習内容の量が増え，授業に参加していないと理解できないことが多くなる。対して，技術科（プログラミング学習）は，中学 1 年生には簡易言語で行ってきた基礎を Python 言語で理解するという内容自体が重複していることが多いためと推察できる。これは，振り返り記述でも「かなり去年よりプログラミングについて詳しくなったと思う」という記述が見られた。そして，「最初は分からなかったプログラミングの問題も回数を重ねていくうちにやり方が分かって、聞かなくても自分でできるようになったので良かったです。」などの記述からできるようになったためと考えられる。

表 4　1 時間の授業で考える時間が多い順の調査結果（中学校 6 年生）

国語	算数	理科	社会	英語	音楽	美術	体育	技術	家庭科
5.11	2.81	2.83	3.69	5.40	7.99	6.99	7.28	4.09	8.29
5	1	3	3	4	9	8	10	1	10

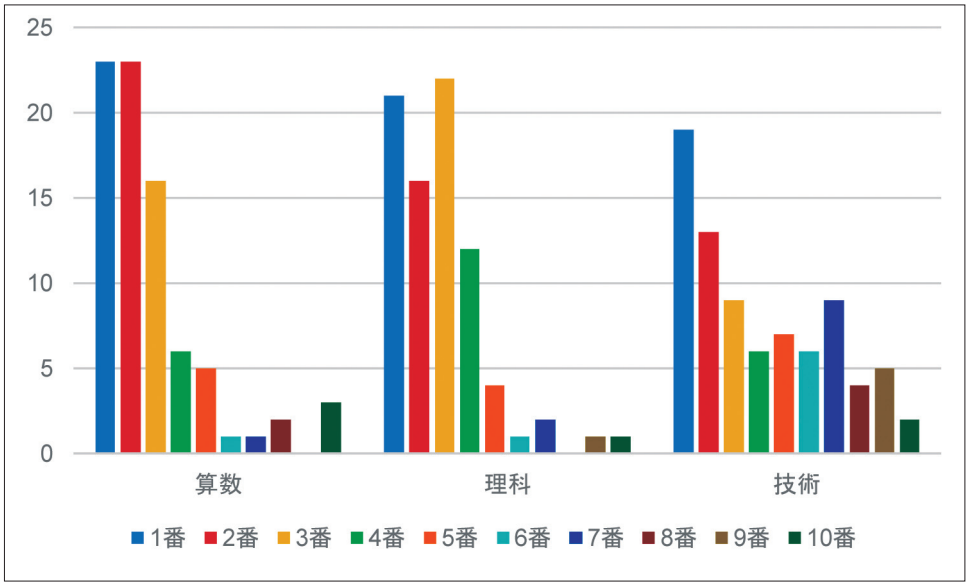

図 5　表 4 の上位分布比較

3. 3. 4　毎時間の授業形態の調査

授業の方法として，中学 1 年生になると小学校 6 年生時より授業実施の合計時間や回数は少ないが，毎時間説明する時間を大幅に少なくした。授業は，簡易言語から Python 言語に移行した。例えば，簡易言語では，配列（array）を学習していたが，Python ではリスト（list）になる。両方使用できるが，数値型と文字列型が混在するリストの方が使い勝手が良いなど，最初は戸惑うことがあり，説明を必要とすることがあった。

調査は，中学 1 年生に 6 年生と同様に「授業で①説明を聞く・②自分で進める・③周りと協力するの進め方の割合はどれくらいが良いか」について調査した。その結果を図 6 に示す。

調査の結果，6 年生とほぼ同じ傾向になった。違いとしては，①の指導者に説明をしてもらう時間が 50%以上欲しいという生徒と，③友達と協力するに 90%以上と答える生徒がいた。結果から見られることは，「先生や周りの人に聞いて，自分の頭の中を整理する」，「自分や周りの人が分からない部分を協力して解決

する」など重複する記述が見られた。その他は，6 年生とあまり変わらない「自分が考えなければあとで困るため」といった理由などが数多くあった。中には，「自分でプログラミングのコードを作って成功すればあのときのようにすればいいんだとなれるため自分でやるのが最優先だと思う。また先生の説明は筋が通っている可能性が高いけど、周りの人と協力するとその人がいまいちわかっていない可能性があるので先生の説明優先。」や「協力するのはいまいちである」という意見もあり，協力する人によるという考え方は中学生の特徴と考えられる。

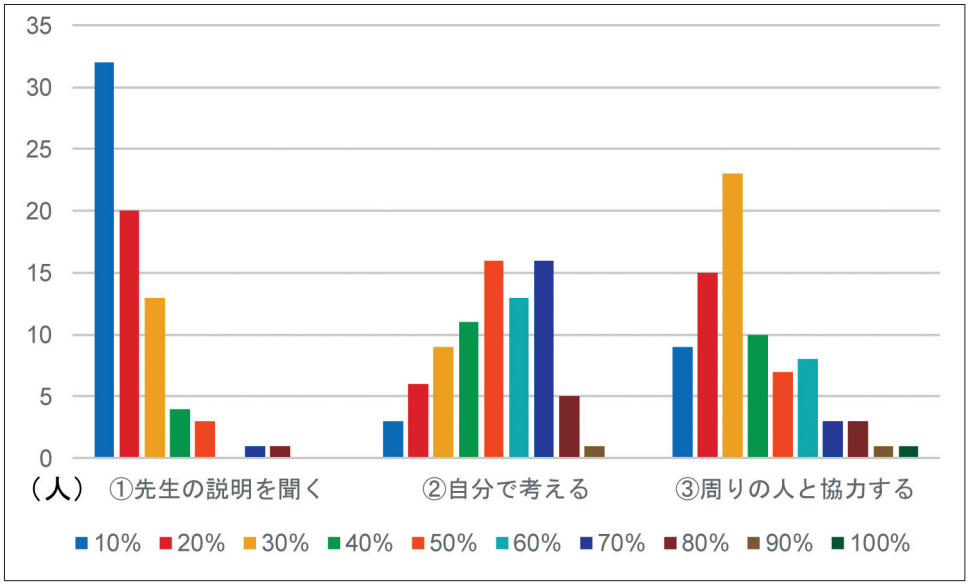

図 6　授業の進め方の割合

4．考察とまとめ

4. 1 考察

結果から明らかになったことは，小学校 6 年生も中学校 1 年生も「指導者に教えてもらいたい」という意見が少なかった。このことは，以前までの授業形態や内容が変化したこと，プログラミングの開始時期が小学校 3 年生からと開始年齢が下がったとことが大きい要因と考えられる。

中学校では，プログラミング学習をこれまでのゼロからのスタートとは違い，小学校で基礎を学んでいることがとても大きい。それは，中学生からプログラミングを開始した生徒は，言語能力が格段にあがるため，ビジュアルプログラミングを行うと確実に理解して考え方を習得する生徒と言語能力が高いために言葉の並び替えの形でできるという二つの傾向があった。そして，後者の傾向が圧倒的に多いため，ビジュアルプログラミングから言語のコードになった時に「分からないや難しい」といった主張が多くあった。その点，小学校の中学年は，言語能力で解決することが難しいため，正確に理解しようとする姿が見られたためである。

もう一つの理由は，これまでの中学生の躓きで多かったことがタイピングの遅さである。これは，説明をしている時に，準備に手間取っていることや打つことが遅いため追いついけず説明の断片しか聞けていないことに起因する。しかし，1 人 1 台端末の導入により，タイピング速度やパソコンの使い方を指導する必要

がなくなり，純粋にプログラミングの内容だけに特化できるようになったことが大きい。さらに，以前はOffice（Windows ソフト）などの文書処理や表計算のソフトの使い方も行っていたが，今では，ソフトは自分で触って使い方を知るものという概念が生まれていることも要因と考えられる。例えば，プレゼンソフトなど指導者が教えると，生徒の活動時間が不足すると共に，教えてもらうまで待っていた。それが，自分で使い方を覚えると，新しいソフトでの学習に抵抗が無くなったことも大きい。この傾向は，「先生の教え方が下手だから分からない」と人のせいにしていたものから，「先生に教えてもらうと自分でプログラミングする時間がなくなる」と記述に変化があり，生徒の振り返りに顕著な成長が見られるようになった。

4. 2 まとめ

プログラミングは，小学校から開始されたことで，あらゆる可能性が見えるようになった。それは，プログラミング的思考のみならず本来のプログラミング学習まで習得が可能と考える。しかし，コロナ感染対応の学校閉鎖期間による中断時期のあった学年は，再度，最初から教えないと忘れている状況があったことから，継続した学習が必要と考える。

今回の調査は，継続的に学習を行った 2 つの学年での調査であった。この調査から，ある程度基礎を学んだあとの授業は，個別進度もしくは協働的な学習が妥当と確認できた。しかし，前者の個別進度の学習は，指導者の力量や学習者の実態把握，準備の大変さから合理的ではない。そのため，後者の協働的な学習が有効と言える。特に中学 1 年生は，調べ学習や探究学習，問題解決学習などを協働的に行ってきた。そのため，その次の段階としてプロジェクト学習（Project based learning）でプログラミング学習を行えば，様々な問題を解決できるのではないかと考えた。

本研究において，児童・生徒のプログラミングを学ぶ上で希望する授業形態を見出すことができた。しかし，プロジェクト学習で実施するとしても，グループの分け方や人数，習熟度別にするかなどの問題がある。さらに，評価や学習の終着点をどこに設定するかなどが課題としてあげられる。今後は，これらの問題を一つずつ解決しながらプログラミング教育の目指すべき方向を検証していきたいと考える。

参考・引用文献

1） 文部科学省，小学校プログラミング教育の手引き（第一版），（第二版），（第三版），
https://www.mext.go.jp/content/20200214-mxt_jogai 02-000004962_004.pdf（第一版）
https://www.mext.go.jp/content/20200214-mxt_jogai 02-000004962_002.pdf（第二版）
https://www.mext.go.jp/content/20200218-mxt_jogai 02-100003171_002.pdf（第三版）
（最終閲覧日 2023 .3 .11）.
2） 樋口耕一: テキスト型データの計量的分析－2 つのアプローチの峻別と統合－，理論と方法，数理社会学会，vol.19（1），pp.101-115，2004.
3） 樋口耕一：KH Coder 3，（3.Beta.07 c - 2023 05/10 版），https://khcoder.net/，2023 年 5 月 10 日取得.
4） 玉瀬耕治：教科の好き嫌いの原因帰属、学習動機の関係，奈良教育大学教育研究所紀要，vol.21，pp.105-113，1985.