

# プログラミング教室の運営と手引の作成

廣瀬研究室 4年  
C1160497 亀谷千香子



## 概要

普段の生活の中で身近な家電、自動車にはコンピュータが内蔵されており、コンピュータを動かしているのはプログラムである。そのため、プログラム仕組みを理解し情報を適切に活用、選択し問題を解決していくことが重要と考えられているため、2020年度から小学生プログラミング教育が必修化になる。プログラミング教育を通して身につけたい力は、「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」である。また、小学校だけのプログラミング教育だけではなく、外部のプログラミング教室も年々増加している。しかし、プログラミング教室を企画・運営するにはどの人を対象とするのか、道具は何を準備するのか、教える体制、準備期間など決め方が明らかでない。また、そのような情報について詳しくまとめたものがWeb上で公開されていない。そのため、プログラミング教室を運営するための手引が必要であると考えた。

本研究では、プログラミング教室をはじめて運営する人のために手引を作成する。手引は、小学生向けプログラミング教室「Rubyてらこった」という東北公益文科大学の私立大学ブランディング事業の一環として行った活動から作成する。Rubyてらこったでは、大学生がプログラミング教室を企画・運営し、小学生にプログラミングを教えた。プログラミング教室に参加した小学生にプログラミング教育を通して、「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」、「工夫する力」、「伝える力」が身に付けることができた。そのことから、Rubyてらこったの活動を踏まえて手引を作成した。(690文字)



# 目次

<b>第1章</b>	<b>はじめに</b>	<b>7</b>
1.1	小学生プログラミング教育の必修化	7
1.2	小学生の発達段階	7
1.3	プログラミング教室の開催状況	8
1.4	プログラミング教室の事例	9
1.5	プログラミング言語	10
1.6	プログラミング教室運営の課題と本研究の目的	12
<b>第2章</b>	<b>プログラミング教室</b>	<b>13</b>
2.1	私立大学ブランディング事業	13
2.2	Ruby てらこったの概要	13
2.2.1	言語の設定	14
2.2.2	プログラミング教育で小学生に身につけたい力	15
2.3	各メソッドの指導内容	16
2.3.1	指導体制	17
2.3.2	授業について	17
2.4	使用した道具	18
2.4.1	PC	18
2.4.2	USB メモリ	20
2.4.3	教科書	20
2.4.4	ノート	22
2.4.5	ホワイトボード	22
2.4.6	マグネットシート (小学生用)	22
2.4.7	マグネットシート (教室のホワイトボード)	22
2.4.8	アルファベット対応表	22
<b>第3章</b>	<b>プログラミングで身につく力についての仮説</b>	<b>25</b>
3.1	プログラミング教室実施の仮説	25
3.2	身につく力の判断基準	25
3.3	身につけたい力の判断基準	25
3.3.1	アンケートの種類	26
3.3.2	作成したプログラムより判断	26

<b>第4章</b>	<b>プログラミング教室の実施結果</b>	<b>27</b>
4.1	アンケートの実施	27
4.2	授業前	27
4.2.1	機械操作について	27
4.2.2	プログラミングについて	28
4.2.3	学習について	28
4.3	毎回の授業	29
4.3.1	楽しかったこと	29
4.3.2	難しかったこと	29
4.4	最後の授業後	30
4.4.1	授業について	30
4.4.2	プログラミングについて	31
4.4.3	学習について	31
4.4.4	その他	32
4.5	アンケートの分析	32
4.5.1	アンケートの分析方法	32
4.5.2	アンケートの分析結果	32
4.6	小学生が作成したプログラム	34
<b>第5章</b>	<b>プログラミング教育の考察</b>	<b>39</b>
5.1	プログラミングについて	39
5.2	小学生に身につけてほしい力	39
5.3	論理的思考について	40
5.4	波及効果について	40
5.5	t検定の結果	40
<b>第6章</b>	<b>手引作成について</b>	<b>41</b>
6.1	手引の作成の概要	41
6.2	手引の構成	41
<b>第7章</b>	<b>結論</b>	<b>43</b>
7.1	結論	43
7.2	今後の展望	43
7.3	手引	43

# 第1章 はじめに

2020年に小学生のプログラミング教育が必修化となる。プログラミング教育で身につけたい力は、「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」である。また、小学校内だけのプログラミング教育だけではなく、外部のプログラミング教室の開催も年々増加している。そのことから、小学生のプログラミング教育の必修化、プログラミング教室の開催状況から述べていく。

## 1.1 小学生プログラミング教育の必修化

普段の生活の中で身近な家電、自動車にはコンピュータが内蔵されており、コンピュータを動かしているのはプログラムである。そのため、プログラム仕組みを理解し情報を適切に活用、選択し問題を解決していくことが重要と考えられているため、2020年度から小学生プログラミング教育が必修化になる。プログラミング教育を通して身につけたい力は、「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」である [1]。

### 知識及び技能

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

### 思考力、判断力、表現力等

発達の段階に即してプログラミング的思考力を育成すること。

### 学びに向かう力、人間性等

発達段階に即して、コンピュータの働き、よりよい人生や社会づくりに生かそうとする態度を涵養すること。

## 1.2 小学生の発達段階

小学校の学習指導要領に書いてあるプログラミングで学ぶ内容は、3～6年生の音楽の授業で様々なリズム・パターンを組み合わせて音楽をつくることなどである(表 1.1)。

ピアジェが仮説化する各発達段階での子どもの思考特徴(表 1.2)では7～12歳(小学生)は、具体的操作段階である。その段階では、具体物を扱う限りにおいては論理的操作が可能だが、科学的な問題や論理の変換のようにあらゆる可能な組み合わせを考えねばならぬ問題には困難を示すとある。しかし、12歳以上になると形式的操作段階となり、経験的事実に基づくだけでなく、仮説による論理的操作や、命題間の論理的関係の理解が可能である。そのため、具体的操作段階である小学生は、プログラミングが難しく感じ、プログラミング教育で身につけたい力が身につ

かないのではないかと考える。

表 1.1: 科目とプログラミング内容 [1] より作成

学年	科目	内容
3～6年生	音楽	様々なリズム・パターンを組み合わせて音楽をつくることをプログラミングを通して学習する場面
4年生	社会	都道府県の特徴を組み合わせて47都道府県を見付けるプログラムの活用を通して、その名称と位置を学習する場面
5年生	算数	プログラミングを通して、正多角形の意味を基に正多角形をかく場面
6年生	理科	身の回りには電気の性質や働きを利用した道具があること等をプログラミングを通して学習する場面
6年生	家庭科	自動炊飯器に組み込まれているプログラムを考える活動を通して、炊飯について学習する場面

表 1.2: ピアジェが仮説化する各発達段階での子どもの思考特徴 [2] より引用

発達段階	年齢の範囲	達成可能な典型と限界
具体的操作段階	7～12歳	具体物を扱う限りにおいては論理的操作が可能になる。ものや事象の静的な状態だけでなく変換の状態をも表象可能、外見上の見えに左右されず保存問題や系列化やクラス化の問題解決が可能、だが科学的な問題や論理変換のようにあらゆる可能な組み合わせを考えねばならぬ問題には困難を示す。
形式的操作段階	12歳～	経験的事実に基づくだけでなく、仮説による論理的操作や、命題間の論理的関係の理解が可能である。より抽象的で複雑な世界についての理解が進み、たとえば、エネルギーの保存や化学的合成に関するような抽象的概念やエネルギーの保存や化学的合成に関するような抽象的概念や知識が獲得される。

### 1.3 プログラミング教室の開催状況

小学校以外にもプログラミングを学ぶことのできる教室がある。外部のプログラミング教室は1999年から2014年以降までを比べてみると、年々増加していることが分かる(図1.1)。2013年以降からはプログラミング教室を開催している団体が急増している。



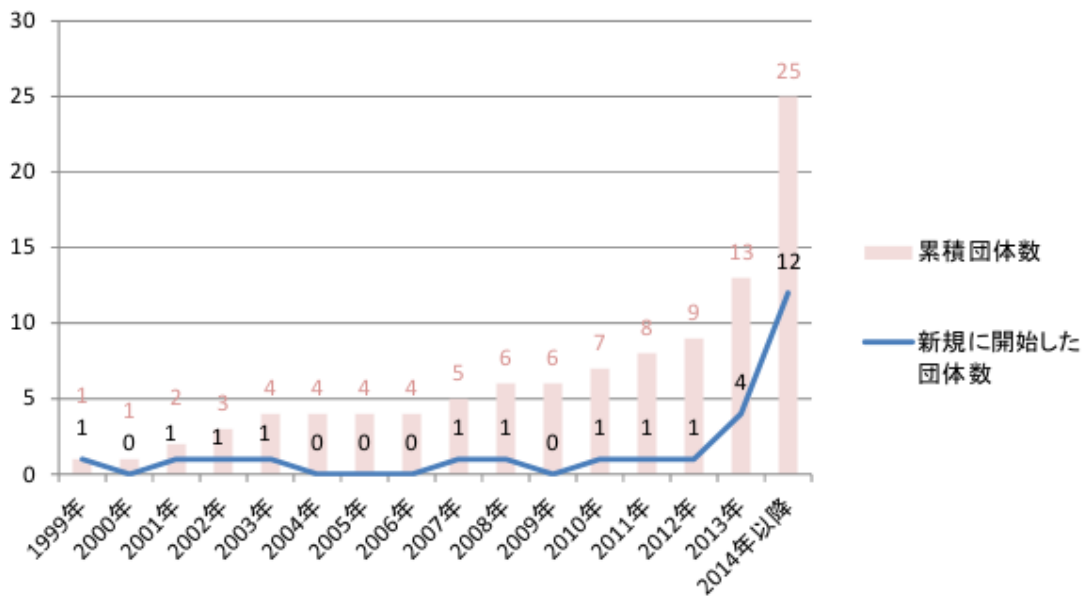


図 1.1: プログラミング教室始時期 [3] 図 5-3 より引用

## 1.4 プログラミング教室の事例

1.3 節で述べたように年々プログラミング教室は増加している。プログラミング教室の対象は、子どもから大人まであり、学べる言語は、Scratch、JavaScript などがある。その中でも小学生を対象にしたプログラミング教室の事例について述べる。

### 松本大学キッズプログラミング教室 [4]

まつもと広域ものづくりフェアで開催した。講習対象を小学校高学年以上である。Scratch を使用。

### 宇城市プログラミング教室 [5]

KidsVenture、さくらインターネット株式会社、宇城市と崇城大学が連携し教室を開催した。対象は小学 4～6 年生である。内容は、電子工作で自作したコンピュータ上にテキスト型言語によるプログラミングである。

### JavaScript による 1 日プログラミング教室 [6]

まつもと広域ものづくりフェアで開催した。対象は中高生である。ゲームを簡単に作成できるように enchant.js を使用。enchant.js とは HTML5 と JavaScript を利用することにし、ゲームを簡単に作れるライブラリである。

### 地方大学における学生主体の子ども向けプログラミング教室 [7]

秋田県立大学本荘キャンパスで開催した。対象は小学生である。低学年、中学年、高学年の 3 つのクラスに分けて実施。プログラミンを使用。プログラミンは 1.5 で詳しく説明する。

### 小学生向けプログラミング体験講座 [8]

新潟国際情報大学中央キャンパスにおいて開催した。対象はプログラミングを体験したことがない小学校4年生から6年生である。Scratchを使用。

## 1.5 プログラミング言語

プログラミング教室で使われているプログラミング言語で高い比率を占めるのがScratchについてJavaScript、Javaである(図1.2)。

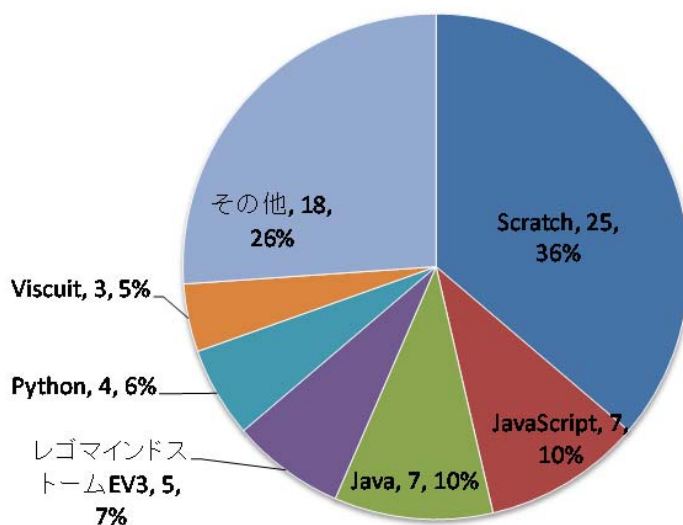


図 1.2: 教室・講座で利用されているプログラミング言語 [3] 図 5-11 より引用

主なプログラミング言語について説明する。

#### 1. LOGO

Seymour A. Papert が児童の思考能力向上を目的として 1960 年代に開発されたもので、命令文によって画面上の「タートル」を動かし、タートルの軌跡で線画を描くようになっている。

#### 2. ドリトル

筑波大学久野、大阪電気通信大学兼宗が開発したものであり、LOGO 同様にタートルを動かし、図形などを描く機能がある。<https://dolittle.eplang.jp/download>

#### 3. Viscuit

原田康徳氏によって開発された手書きイラストを用いたアニメーション作成機能に特化した

プログラミング言語であり、タブレットで利用可能である。<https://www.viscuit.com/>

#### 4. Scratch

MIT メディアラボが開発したプログラミング言語学習環境であり、ブロックの組み合わせによってプログラミングするオブジェクト指向言語である。<https://scratch.mit.edu/>

#### 5. Blockly

Google が作成したビジュアルプログラミング開発ライブラリであり、ブラウザ上で動作するオープンソースのブロック型言語である。タブレットで利用可能である。<https://developers.google.com/blockly/>

#### 6. Smalruby

高尾宏治（ネットワーク応用通信研究所）によって開発された Ruby をもとにしたビジュアル言語である。<https://smalruby.jp/blog/2017/01/14/smalruby-editor-0-4-1-has-been-released.html>

#### 7. プログラミン

文部科学省が Scratch を参考に開発したものでブラウザで動作するプログラミング学習用サービスである。<http://www.mext.go.jp/programin/>

#### 8. JavaScript

Netscape Communications 社によって開発されプログラミング教室・講座では Web アプリケーションの作成に用いられることが多い言語である。<http://www.ecmascript.org/>

#### 9. Java

Sun Microsystems によって開発され、プログラミング教室・講座では Android アプリ作成に用いられるケースが多い言語である。<https://www.oracle.com/technetwork/java/javase/overview/index.html>

#### 10. Python

Guido van Rossum によって開発され、全米の大学では初心者にはプログラミングを教育する教材として最もカリキュラムに取り入れられている言語である。<https://www.python.org/>

#### 11. C

ブライアン・カーニハンとデニス・リッチーによって開発され、現在もっとも普及しているプログラミング言語である。国際標準化機構 (ISO) や日本工業規格 (JIS) にも標準として採用されている。

#### 12. HTML

CERN（欧州合同素粒子原子核研究機構）の研究者であったティム・バーナーズ・リーによって開発・公開され OS X や iOS 向けのアプリケーションの開発に利用できる言語である。現在は改定版である HTML5 が多く利用されており、HTML5 では高度な Web アプ

りも作成可能である。<https://www.w3.org/html/>

## 1.6 プログラミング教室運営の課題と本研究の目的

東北公益文科大学(以下本学)ではRubyを用いたプログラミング教室「Rubyてらこった」を開催した。実際に大学生がプログラミング教室を企画・運営をする時に、どのような準備、指導する内容、指導体制など決める基準が明確に分からなかった。また、プログラミング教室の企画の立て方、運営の方法、準備する道具、準備期間について詳しくまとめているものがWeb上で公開されているものがなかった。そのため、プログラミング教室を行うための手引が必要と考えた。

本研究では、小学生向けのプログラミング教室を企画・運営し、はじめてプログラミング教室を運営する人を対象にしたRuby等を用いた小学生向けのプログラミング教室の手引を作成することを目的とする。

## 第2章 プログラミング教室

本学では平成30年度から私立大学ブランディング事業の一環として、小学5、6年生向けのRubyを用いたプログラミング教室を行っている。プログラミング教室の内容については以下の通りである。

。

### 2.1 私立大学ブランディング事業

本学では平成30年度から私立大学ブランディング事業を行っている。私立大学ブランディング事業とは、大学の特色ある研究を基軸として全学的な独自色を大きく打ち出す取組を行う私立大学の機能強化を促進する取り組みである[9]。本学のブランディング事業では、5つの軸がある[10]。

1. 地域資源の掘り起こし研究
2. 庄内地域の文化の保存・集積（アーカイブ化）
3. 庄内地域の無形文化財をIT技術により伝承する研究
4. 地域文化をIT技術により他の地域の人にもわかりやすく伝え、庄内地域の魅力を発信するための手法の研究
5. 地域資源を産業振興やIT技術により活用できる人材の育成

その中の「5. 地域資源を産業振興やIT技術により活用できる人材の育成」としてRubyてらこったの企画・運営を行ってきている。

### 2.2 Rubyてらこったの概要

本学の私立大学ブランディング事業の一環としてプログラミング教室「Rubyてらこった」を開催した。Rubyてらこったの目的は、庄内地方を中心とした地域の若者達に情報技術を教え、情報社会を生き抜くために必要な力を身につけていくことである。Rubyてらこったは、キーボード入力、操作の関係からアルファベットを習い終わっている方が良いと考えたため小学5、6年生を対象としている。

### 2.2.1 言語の設定

プログラミング教室を開催するにあたり、まず教える対象となる言語を決定した。様々なプログラミング言語がある中で Ruby を選択した理由は二つあり、一つは、Scratch のようなグラフィック型ではなく、小学生のうちからテキスト型の本格的なプログラミングを学ぶ機会を提供したいと考えたからで、もうひとつは、本学では2年次に1年間必修科目の基礎プログラミングで Ruby を学ぶため教えやすいと考えたからである。2年次に学ぶ内容は以下の通りである [11]。この2つの理由より、Ruby を教えることに決めた。

#### 1. 計算システムの基本操作と概念

なぜプログラミングが必要であるのかを知り、基本的なコマンド(コンピュータに特定の機能の実行を指示する命令)操作、Unix(OSの一つ)について、ファイル、ディレクトリの概念などを学ぶ

#### 2. プログラミングの基礎

インタプリタ、プログラミングについて学ぶ

#### 3. 変数と値

変数の概念と値処理の方法を学ぶ

#### 4. 演算子

プログラム上での四則演算の方法を学ぶ

#### 5. 制御構造

条件分岐や場合分けの方法を学ぶ

#### 6. 出力処理を行うメソッド

画面出力の基本的な内容を学ぶ

#### 7. 配列

番号をつけて一度に多くのデータを処理する方法を学ぶ

#### 8. パターンマッチング(正規表現)

データに含まれる文字列のパターンを指定し、そこから必要な部分を取り出す方法を学ぶ

#### 9. ファイルの入出力

直接ファイルに入出力をする方法を学ぶ

#### 10. 計算機の内部構造

デジタルで数字や文字などを表現する方法について学ぶ

#### 11. ハッシュ

順番、個数にとらわれず任意の属性と値を結びつけ管理する方法を学ぶ

#### 12. 再帰

大きな問題を類似した小さな問題に分割して考える方法を学ぶ

#### 13. CGI

自分が作成したシステムを Web 上に効果的な形で視覚化させる方法を学ぶ

### 2.2.2 プログラミング教育で小学生に身につけたい力

文部科学省は、小学校プログラミング教育の手引 (第二版) において小学生にプログラミング教育で身につけたい力として「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」が述べられている (表 2.1)。この項目に加え Ruby てらこったでプログラミングを学んだ小学生に身につけたい力は「工夫する力」、「伝える力」であり、内容は、以下の通りである。

表 2.1: 身につけたい力 [1] より作成

#### 知識及び技能

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

#### 思考力、判断力、表現力等

発達の段階に即してプログラミング的思考力を育成すること。

#### 学びに向かう力、人間性等

発達段階に即して、コンピュータの働き、よりよい人生や社会づくりに生かそうとする態度を涵養すること。

#### 工夫する力

授業で使うサンプルのプログラムを自分で考えて、工夫をして、他の人とは違うプログラムを作ることができるようになること。

#### 伝える力

伝える力には、2つの意味がある。1つ目は、自分の作成したプログラムの発表を通して工夫した点、頑張った点など伝えられるようになること。

2つ目は、周りの人が作成したプログラムの発表を聞いて感想などを伝えられるようになること。

## 2.3 各メソッドの指導内容

授業内容は、全5回の限られた回数の中で理解でき、プログラムを作成、実行する時に複雑にならないようにし、教える順番は、基本的な操作から徐々に難しい内容になるよう設定している。5回目で自由に作成し、発表できるようなプログラムを作成できるように組み立てた(表2.2)。

表 2.2: 授業内容

回数	内容	作成プログラム
1回目	基本操作	画面出力 自己紹介プログラム
2回目	ループ	繰り返し処理のプログラム
3回目	配列	データ処理のプログラム
4回目	条件分岐	条件のあるプログラム
5回目	まとめ	習ったことを応用してプログラムを作成

1回目から5回目までの学習する内容は、出力メソッド、文字列入力処理に必要なメソッド、制御構造、配列及び乱数、sleep 関数である。内容は、以下の通りである。

### 出力メソッド (`print`, `printf`, `puts`)

アスキーアートのように入力したものをそのまま出力させるために必要である。`print` と `puts` も同じように画面出力のためのものであるが、`puts` は末尾に改行した形で出力される。`printf` は書式付の `print` 文であるため数字などを出力させるために利用した。

### 文字列入力処理に必要なメソッド (`gets`, `chomp`)

キーボードに打ち込んだ値を文字列として取得するゲームプログラムを作るときに必要なある。

### 制御構造 (`while`, `if`, `elsif`, `else`)

制御構造とはプログラムで実行される流れを定めたものであり、レジスタープログラムなどの入力したものを繰り返しの処理を行うときやクイズの結果の判定や条件で繰り返しの処理を行うために必要である。繰り返しの処理を行うために `while`、条件分岐には `if`, `elsif`, `else` を使用した。使い方の例は以下の通りである。

```

if a
  hogehoge1
elsif b
  hogehoge2
else
  hogehoge3
end

```

条件が `a` であれば `if` の `hogehoge1` の部分が実行される。条件が `a` ではなく 2つ目の `b` であれば `hogehoge2` の部分が実行される。条件が `a` でも `b` でもない時は `else` の `hogehoge3` の部分を実行する。



### 配列及び乱数 (srand, rand)

2つを組み合わせてクイズ問題を用意して乱数で選ばせたり、ジャンケンの手の内をランダムに出したりするために使うので必要である。

### sleep 関数 (sleep)

プログラムを時間指定して一時停止することができるため大学生の課題でも反応が良いため小学生も楽しむことができると思ったので必要である。

## 2.3.1 指導体制

授業でただ説明するだけでは、教えにくいとため、授業の担当者1人、タイムキーパーが1人、授業アシスタントは参加する小学生の人数と同じようになるような体制にした。1対1で教える体制にしたのは、授業に追いつけない部分をサポートできるようにするためである。授業が進む速度に対して、小学生が理解する速度に差が発生することがあった。その際には、随時授業の合間に時間をとり、小学生が理解する速度に合わせて授業を展開していく形を心がけた。

## 2.3.2 授業について

授業は、小学生でも内容が分かりやすく、楽しく学ぶことができるようにすることを意識した。1回の授業時間を2時間に設定し、50分ほど授業をしたら約15分ほどの長めの休憩を取り、小学生が集中しやすいようにした。授業の流れは以下の通りである。

### 1. 前回の授業の復習

授業で学んだ内容のポイントを復習する。

### 2. タイピング練習

授業で使う英単語を3回ずつ練習する時間を取る。

### 3. 内容の説明

授業で新しく学ぶことを説明し、重要なポイントは、メモをとる時間を作る。

### 4. サンプルプログラムの実行

最初にどのようなプログラムなのか実行させる時間にする。

### 5. サンプルプログラムの説明

4のサンプルプログラムがどうしてそのような動きをするのか説明する時間をつくる。

### 6. 休憩

長めの休憩をとる。

### 7. プログラム作成

プログラムの作成では、学んだ内容を使い自分で考えて作成してもらおう。作成の時は、ホ

ホワイトボードにマグネットシートを貼ったり、ペンで書いたりしてからパソコンに入力する。イメージ通りにプログラムが作れない時は、アシスタントがサポート、説明をする。サポート、説明で重要なのは、どうしたらプログラムがうまく動くのか考えてもらい理解してもらおうことである。

#### 8. 作成したプログラム発表

作成したプログラムをみんなの前で発表する。発表方法は、二つあり、一つは、発表方法はみんなの前で作成したプログラムを実行し発表するでもうひとつは、他の人が作成したプログラムを実行してみる方法である。小学生が作成したプログラムについては 4.4.4 で詳しく述べる。

## 2.4 使用した道具

小学生に分かりやすく教えるための道具として、教科書、マグネットシートなどを作成した(表 2.3)。

表 2.3: 必要な道具一覧

アイテム	用途
PC	小学生の操作用、授業用、スライド用のために使用
USB メモリ	作成したプログラムを記録するために使用また、教室閉講後にも利用できるよう無料で配布可能なシステムをインストールしておく
教科書	授業の内容を確認するために使用
ノート	必要なことを書くために使用
ホワイトボード	プログラムを PC に入力する前に作成したいプログラムを書くために使用
マグネットシート	プログラムを作成するとき何を組み合わせると良いのか分かりやすくするために使用
アルファベット対応表	文字を入力するとき分かりやすくするために使用

### 2.4.1 PC

小学生用、授業用の PC の OS は NetBSD である。NetBSD とはオープンソースのオペレーティングシステムであり、本学の計算機と同じ OS だと学生が教えやすいからである。また、授業以外でも使ってほしいため無料で配布できるオープンソースが良いと考えたからである。PC は、小学生用、授業用の他にスライド用の PC も用意した。

- 小学生用 PC

授業中のプログラムを作成、Web ページからプログラムをダウンロードするために使用した(図 2.1)。

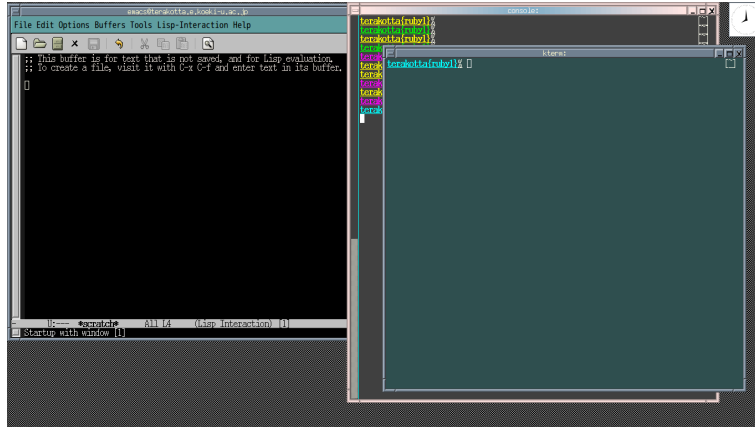


図 2.1: PC 画面

- 講師用 PC

小学生に分かりやすくするため小学生の使用している計算機と同じものを使用した。

- スライド用 PC

教室では2つのプロジェクターが利用できるため、教卓 PC から実際の操作画面と同時に、教材のスライドを映すための PC を別途用意した。教室に2つプロジェクターが無かった場合は大型液晶ディスプレイに映して授業をした。

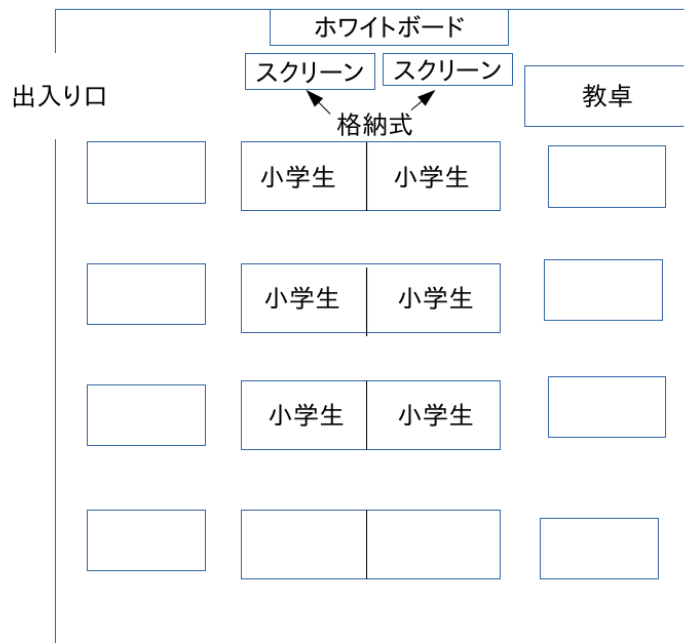


図 2.2: 教室見取り図

### 2.4.2 USB メモリ

今回は 16GB 以上の容量を持つ高速な USB フラッシュメモリに OS と教材に使用するソフトウェア一式をインストールし、USB メモリから起動して授業用環境として使用できるようにした (23 ページ図 (a))。USB メモリは、OS のインストールした後、参加者一人のアカウントを作成し、名前を記入して配布をした。

### 2.4.3 教科書

授業内容に合わせて、学生らで作成した (23 ページ図 (b))。オリジナルのイラストや分かりやすい図などを取り入れ、興味が湧くような内容にすることを心がけた。また、語句の説明なども小学生でも分かるようにした。内容は以下のようなものである。

#### プログラミングとは

Ruby とは何か、プログラムの作成方法を説明する。

#### ファイル、ディレクトリ

小学生でも分かりやすいようにディレクトリは、プログラムをしまっておく場所、ファイルはプログラムをつくる場所とイメージ図を使い説明した (図 2.3)。

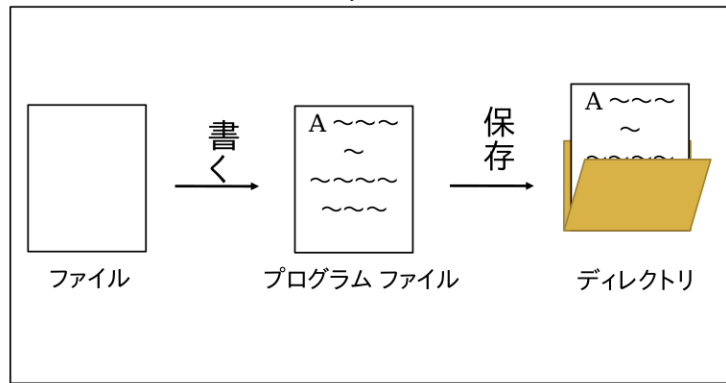


図 2.3: ファイル、ディレクトリ

**print, puts, gets** について

画面出力の方法、キーボードから入力させる方法をまとめた。

**while** について

ループについて学ぶ。

**配列、乱数**

配列を使ってランダムに取り出す方法をまとめた。

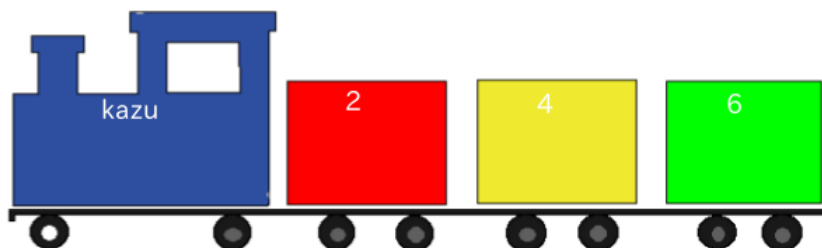


図 2.4: 配列のイメージ図

**if, elsif, else** について

条件分岐をさせる方法、if, elsif, else の違いについてまとめた。

**プログラムを工夫するためのメソッド**について

プログラムを出力させるときに文字が変化したり、時間をおいて表示させたりするものをまとめた。

**コマンド集**

プログラムの作成時などに覚えておくと便利なコマンドをまとめて書いておき、分からない時にすぐに確認できるようにした。

#### 2.4.4 ノート

初回の授業で配布し、小学生が学んだことや気づいたこと、ポイントなどをこまめに書き込めるように用意した(23ページ図(c))。授業内では、ノートに書く場面があれば講師が指示を促し小学生がノートを書く時間を設けた。

#### 2.4.5 ホワイトボード

A3程の大きさのホワイトボードを参加者全員に配布した。ホワイトボードにプログラムを書いてからPCにプログラムを打ち込んだ方が間違えることが少なくなり、分かりやすいと考えたため用意した(23ページ図(d))。また、小学生がホワイトボードに書きながらプログラムを組んで行くことで理解を深める目的で使用した。

#### 2.4.6 マグネットシート(小学生用)

授業の勉強する新しいことや大切なものをマグネットシートにした(23ページ図(e))。これを貼るだけで各メソッドを記述できるようにし、記憶しやすくするために作った。例を挙げると、「while」と「end」などのセットで扱うものに関しては、セットで先にマグネットをホワイトボードに貼り付けてからプログラムを組み立てるなどの工夫をした。あらかじめ用意されているものを使用したため、間違いの減少や作業時間の削減にも繋がった。

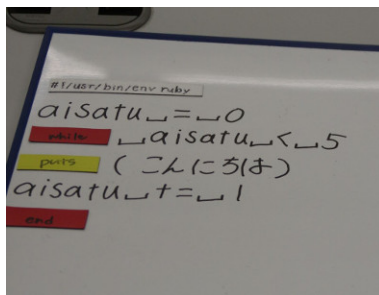


図 2.5: while と end のマグネットシートを使用

#### 2.4.7 マグネットシート(教室のホワイトボード)

小学生にメソッドの部分の部分を分かりやすくするために、教室のホワイトボードに貼るマグネットシートを用意した(23ページ図(f))。

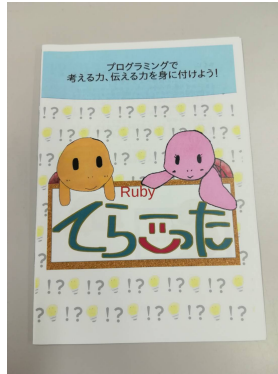
#### 2.4.8 アルファベット対応表

小学5年生は、ローマ字の大文字だけ習っているため、小文字で書かれたプログラムリストの小文字はわからない。特に、大文字と小文字が違う「l(エル)」と「i(アイ)」や「h(エイチ)

)」と「n( エヌ )」と「r( アール )」などが小学生にとって分かりにくい。そのため、大文字小文字を理解するために作成し配布した(23 ページ図 (g))。また、アルファベット対応表の配布だけでなく、小文字で書かれた Ruby の単語の文字ごとに大文字のルビを振るようにした。



(a) USB メモリ



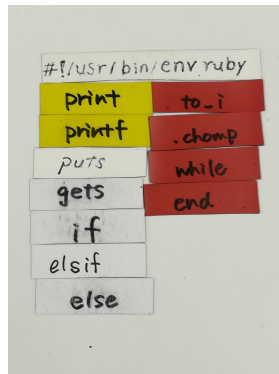
(b) 教科書



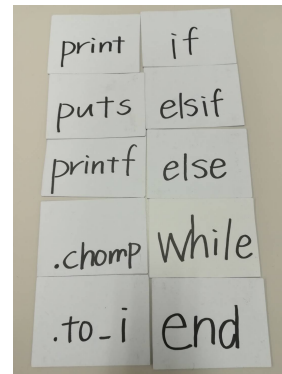
(c) ノート



(d) ホワイトボード



(e) マグネットシート



(f) マグネットシート (教室のホワイトボード)

— アルファベット表 —

A a	B b	C c	D d
E e	F f	G g	H h
I i	J j	K k	L l
N n	M m	O o	P p
Q q	R r	S s	T t
U u	V v	W w	X x
Y y	Z z		

(g) アルファベット対応表



## 第3章 プログラミングで身につく力についての仮説

本章では参加した小学生がプログラミングの授業を通してどのような変化があるか仮説を立てる。仮説を判断するため基準を述べる。

### 3.1 プログラミング教室実施の仮説

参加する小学生はある程度プログラミングに興味があり、難しいという印象よりも楽しそうという印象があると考えられる。また、授業を通してプログラムが難しいというイメージが減少するのではないかと考えた。さらに、授業を通して波及効果として学習が好きな方に変化し、別のことにも挑戦してみようという意欲的になると仮説を立てる。

### 3.2 身につく力の判断基準

実際に 3.1 節の仮説を立てたものを検証をする方法としてアンケートを 3 種作成した。アンケートの種類と結果については以下の通りである。また、アンケートで判断できないものは、授業の様子や小学生が作成したプログラムより判断する。

### 3.3 身につけたい力の判断基準

2.2.2 節で述べた力を Ruby で作ったものの具体的に判断する基準を述べる。

- 知識及び技能  
プログラムを作成できる。  
様々なメソッドを覚え、使うことができる。
- 思考力、判断力、表現力等  
自分が作成したいプログラムに対してどのメソッドを使うとできるか考えることができる。
- 学びに向かう力、人間性等  
サンプルプログラムをどのような工夫できるか自分で教科書を見てたり、大学生に聞いたりして取り組むことができる。
- 工夫する力  
サンプルプログラムを自分が作りたいものに作り変えることができる。

- 伝える力  
作成したものに対して発表ができる。  
他の人の発表を聞いた後に感想を伝えることができる。

### 3.3.1 アンケートの種類

プログラミングで身につく力についての仮説を検証するために、アンケートは1回目の授業の開始前、毎回の授業後、最後の授業後に行った。

授業開始前アンケート

- 普段の機械操作について
- プログラミング教室参加理由
- プログラミングの経験
- プログラミングのイメージ
- 作成したいプログラム
- 学習について

毎回の授業後アンケート

- 授業で楽しかったこと (何でも可)
- 授業中難しかったこと (何でも可)

最後の授業後アンケート

- 授業について
- プログラミングについて
- 論理的思考力について
- 学習について

### 3.3.2 作成したプログラムより判断

小学生が授業をした後にサンプルプログラムの改造、5回目の授業で作成したオリジナルプログラムから知識が身についたか判断する。

## 第4章 プログラミング教室の実施結果

作成したアンケートを1回目の授業前、毎回の授業後、最後の授業後に行った。本章では、そのアンケート結果を述べていく。

### 4.1 アンケートの実施

2019年度のRubyてらこっちは、6月開始の教室には6名、7月開始の教室には6名、合計で12名参加した。

### 4.2 授業前

普段の機械操作とプログラミングのイメージ、経験、学校の学習についてのアンケートを実施した。

#### 4.2.1 機械操作について

普段どんな機器を使っているのか調査した結果は以下の通りである(表4.1)。選択肢の後につく()は点数化したものである。機械操作の部分については、使用頻度の高いものから高い点数をつけた。

表 4.1: 機械操作について

パソコンは使いますか			
週3回以上(3)	週2回以上(2)	週1回以下(1)	使わない(0)
0	2	6	4
スマートフォンは使いますか			
週3回以上(3)	週2回以上(2)	週1回以下(1)	使わない(0)
9	0	1	2
テレビゲームはしますか (iPad, Switchも含めて)			
週3回以上(3)	週2回以上(2)	週1回以下(1)	しない(0)
7	2	1	2
パソコンゲームはしますか			
週3回以上(3)	週2回以上(2)	週1回以下(1)	しない(0)
1	1	1	9
携帯ゲームはしますか			
週3回以上(3)	週2回以上(2)	週1回以下(1)	しない(0)
0	5	2	5

機械操作については、パソコンよりもスマートフォンを使用している参加者が多かった。また、ゲームをするときも、パソコンよりもテレビゲーム (iPad, Switch も含め) のほうが使用頻度が多いという結果がでた。

#### 4.2.2 プログラミングについて

プログラミング教室に参加した理由とプログラミングのイメージについて調査した。結果は以下の通りである (表 4.2)。選択肢の後ろにつく ( ) は点数化したものである。

表 4.2: プログラミング教室に参加した理由とプログラミングのイメージについて

なぜプログラミング教室に参加しましたか			
親のすすめ	友達と一緒にだから	プログラミングに興味があるから	その他
4	0	7	1
プログラミングをしたことがありますか			
したことがある		したことがない	
5		7	
プログラミングは楽しそう			
とても楽しそう (4)	楽しそう (3)	楽しそうではない (2)	全く楽しそうではない (1)
6	6	0	0
プログラミングは難しそう			
とても難しそう (4)	難しそう (3)	難しそうではない (2)	全く難しそうではない (1)
2	7	3	0
作ってみたいプログラムはありますか (自由記述)			
ゲームプログラム (3名回答)、計算させるプログラム、学習のプログラム、字を書くプログラム 占いプログラム			

プログラミング教室の参加理由は、親の勧めや友達と一緒にだからという理由よりも、プログラミングに興味があるからという理由が一番多かった。プログラミング経験については、半数以上が経験がないという結果になった。プログラミングのイメージでは、楽しそうというイメージもあるが、難しそうというイメージがあることが分かる。作成してみたいプログラムについては、ゲームプログラムを作成してみたいというのが多かった。

#### 4.2.3 学習について

小学校での学習について調査した。結果は以下の通りである (表 4.3)。選択肢の後ろにつく ( ) は点数化したものである。

学習については、学校の勉強が好きであると回答した参加者が半数以上いた。また、学校の勉強が難しいと感じている人が少なかった。

表 4.3: 小学校での学習について

学校の勉強が好き			
とても好き (4)	好き (3)	好きではない (2)	全く好きではない (1)
4	6	2	0
学校の勉強が難しい			
とても難しい (4)	難しい (3)	難しくはない (2)	全く難しくもない (1)
0	2	6	4

### 4.3 毎回の授業

授業の振り返りとして授業中に楽しかったこと、難しかったことを書いてもらった。

#### 4.3.1 楽しかったこと

1回目の授業から5回目の授業で楽しかったことは以下の通りである(表4.4)。

表 4.4: 授業で楽しかったこと

回数	内容
1回目	文字を打つこと(4名)、プログラム作成(3名)、作成したプログラム発表(3名) コマンド入力(2名)、大学生と話したこと、学校で学習できないことを学べた
2回目	プログラムの改造(6名)、作成したプログラムの実行(2名)、文字を打つこと(2名) 他の人の作成したプログラムを実行すること、大学生と話したこと
3回目	プログラムを改造したこと(5名)、他の人が作成したプログラムを実行すること(4名)
4回目	プログラムの改造(9名)、プログラムを実行すること(2名)、大学生と話したこと
5回目	プログラムを作成したこと(10名)、プログラムを実行したこと、大学生と勉強したこと

授業で楽しかったことは、プログラミングを改造したことや文字を入力するという作業だけではなく、発表したり、他の人のプログラムを実行したことなどであった。

#### 4.3.2 難しかったこと

初回から5回目までの全体を通して難しいと感じたことは以下の通りである(表4.5)。なかでも、プログラムの改造、文字を打つこと、プログラムを作成するためのメソッドについての意見が多かった。文字を打つことでは、4.2.1の結果から、PCを使用する機会が少ないため、キーボード操作が難しく感じたのではないかと考える。

表 4.5: 授業で難しかったこと

回数	内容
1回目	文字を打つこと (6名)、print と puts の違い (3名)、発表すること
2回目	プログラムの改造 (4名)、文字を打つこと (3名)、while の考え方 (2名) break について
3回目	プログラムを改造したこと (3名)、文字を打つこと (2名)、配列、乱数
4回目	プログラムの改造 (5名)、文字を打つこと、条件分岐
5回目	プログラム作成 (4名)、文字を打つこと (3名)、変数 (2名)、覚えることが多い 発表したこと

## 4.4 最後の授業後

5回目の授業後に1回目から5回目までの授業について、プログラミングを実際に行ってみてどうだったか、学校の学習について、今後挑戦したいことについてアンケートを行った。

### 4.4.1 授業について

プログラミングの授業を実際に受講してみてどうだったか調査した。結果は以下の通りである(表 4.6)。選択肢の後ろにつく () は点数化したものである。

表 4.6: 授業について

授業時間はどうでしたか			
長い (1)	ちょうどよい (2)	短い (3)	
1	11	0	
内容は分かりやすかった			
とても分かりやすい (4)	分かりやすい (3)	分かりにくい (2)	とても分かりにくい (1)
7	5	0	0
大学生は楽しかったですか			
とても優しい (4)	優しい (3)	優しくない (2)	全く優しくない (1)
12	0	0	0
授業に参加してみて楽しかったことを書いてください (自由記述)			
プログラムを作成・改造すること (8名)、プログラムを実行できたこと、文字を打つこと 発表を聞くこと、大学生と話すこと、全て			
授業に参加してみて難しかったことを書いてください (自由記述)			
覚えることが多いこと (3名)、文字を打つこと (3名)、プログラムを作成・改造すること (2名)、変数 (2名)、配列、print			

授業については、授業時間はちょうどよく、内容は分かりやすかったという結果がでた。参加し

てみて楽しかったことは、プログラミングを作成・改造すること、プログラムの実行だけではなく、大学生と話したことなどがあつた。難しく感じたこととして、覚えることが多い、文字を打つことという回答があつた。

#### 4.4.2 プログラミングについて

プログラミングをしてみてどうだったのか調査してみた。結果は(表 4.7)の通りである。選択肢の後につく () は点数化したものである。

表 4.7: プログラミングについて

プログラミングは楽しかった			
とても楽しかった (4)	楽しかった (3)	楽しくなかった (2)	全く楽しくなかった (1)
8	4	0	0
プログラミングは難しかった			
とても難しかった (4)	難しかった (3)	難しくなかった (2)	全く難しくなかった (1)
1	7	2	2
またプログラミングをしたい			
とてもしたいと思う (4)	したいと思う (3)	したくないと思う (2)	全然したくないと思う (1)
5	7	0	0
筋道をたてて考える力が身についたと思う			
とても思う (4)	思う (3)	思わない (2)	全く思わない (1)
3	9	0	0

プログラミングを実際にしてみて、楽しかったという回答もあつたが、難しく感じていた参加者がいた。しかし、またプログラミングをしたいと思つていることが分かつた。また、プログラミングを通して論理的思考力が身についたと参加者が実感できたと回答していた。

#### 4.4.3 学習について

小学校での学習について調査した。結果は以下の通りである(表 4.8)。選択肢の後につく () は点数化したものである。学習については、授業開始前の結果 4.2.3 節と比較してあまり変化が現れなかつた。

表 4.8: 小学校での学習について

学校の勉強が好き			
とても好き (4)	好き (3)	好きではない (2)	全く好きではない (1)
5	4	3	0
学校の勉強が難しい			
とても難しい (4)	難しい (3)	難しくはない (2)	全く難しくはない (1)
0	3	4	5

#### 4.4.4 その他

自由に今後挑戦したいことを書いてもらった。内容は、以下の通りである (表 4.9)。

表 4.9: 挑戦したいこと

挑戦してみたいことを書いてください (何でも可)
もっと難しいプログラムを作る (3名)、家でもプログラミングをする (2名)、自分でゲームを作りたい (2名)、みんなにプログラミングを教える、50m クロールで泳ぐ、看護師になっていろいろな人を救いたい

挑戦してみたいこととして、もっと難しいプログラムを作る、家でもプログラミングをする、自分でゲームを作りたいなどプログラミングに関することや、50m クロールで泳ぐ、看護師になっていろいろな人を救いたいなどプログラミング以外のことにも挑戦したいこと回答した参加者がいた。

## 4.5 アンケートの分析

プログラミングをしてみる前としてみた後では、波及効果が現れるのか、数値に変化があるのかを比較するために調べる方法として t 検定を用いた。

### 4.5.1 アンケートの分析方法

同じ人を 2 回測定するので対応のある t 検定を用いた。アンケートの結果を数値化し分析を行った [13]。アンケート結果を数値に直した値については、4.4 節のアンケート結果にある通りである。R に与えた実際の数値や関数のリストは、付録に添付する。

### 4.5.2 アンケートの分析結果

t 検定で比較する質問項目は、プログラミングについて楽しそう、難しそうが実際にプログラミングを体験してみて楽しかった難しかったの経験、学習 (勉強が好き、勉強が難しい) についてである。結果は以下の通りである。



- 楽しそうと楽しかったを比較

水準1は授業前の「プログラミングは楽しそう」というイメージ(とても楽しそう(4)、楽しそう(3)、楽しそうではない(2)全く楽しそうではない(1))の数値である。水準2は授業後の「プログラミングは楽しかった」という(とても楽しかった(4)、楽しかった(3)、楽しくなかった(2)全く楽しくなかった(1))数値である。

```
== Mean & S.D. ( SD=sqrt(Vtotal/N) ) ==
```

	N	Mean	S.D.
水準1	12	3.75	0.43
水準2	12	3.5	0.5
差	12	0.25	0.72

t(11)= 1.1516 , ns (.10<p)

事前のアンケートの「プログラミングは楽しそう」であるの結果と5回目の授業の「プログラミングは楽しかった」の結果の平均を比較したところ有意ではなかった。つまり、イメージと実際の差はなかった。

- 難しそうと難しかったを比較

水準1は、授業前の「プログラミングは難しそう」というイメージ(とても難しそう(4)、難しそう(3)、難しそうではない(2)、全く難しそうではない(1))数値である。水準2は、授業後の「プログラミングは難しかった」という(とても難しかった(4)、難しかった(3)、難しくなかった(2)、全く難しくなかった(1))数値である。

```
== Mean & S.D. ( SD=sqrt(Vtotal/N) ) ==
```

	N	Mean	S.D.
水準1	12	2.58	0.76
水準2	12	3	0.58
差	12	-0.42	0.95

t(11)= 1.4663 , ns (.10<p)

事前のアンケートの「プログラミングは難しそう」の結果と5回目の授業の「プログラミングは難しかった」の結果の平均を比較したところ有意ではなかった。つまり、イメージと実際の差はなかった。

- 勉強が好き

水準1は、授業前の「学校の勉強が好き」の値(とても好き(4)、好き(3)、好きではない(2)、全く好きではない(1))である。水準2は、授業後の学校の「勉強が好き」の値(とても好き(4)、好き(3)、好きではない(2)、全く好きではない(1))である。

```
== Mean & S.D. ( SD=sqrt(Vtotal/N) ) ==
```

	N	Mean	S.D.
水準1	12	3.42	0.76
水準2	12	3	0.58
差	12	0.42	1.26

t(11)= 1.1055 , ns (.10<p)

事前のアンケートの「プログラミングは楽しそうである」の結果と5回目の授業の「プログラミングは楽しかった」の結果の平均を比較したところ有意ではなかった。つまり、イメージと実際の差はなかった。

- 勉強が難しい

水準1は、授業前の学校の勉強が難しいの値(とても難しい(4)、難しい(3)、難しくはない(2)全く難しくない(1))である。水準2は、授業後の学校の勉強が難しいの値(とても難しい(4)、難しい(3)、難しくはない(2)全く難しくない(1))である。

```
== Mean & S.D. ( SD=sqrt(Vtotal/N) ) ==
```

	N	Mean	S.D.
水準1	12	1.5	0.5
水準2	12	2.17	0.8
差	12	-0.67	1.03

t(11)= 2.1574 , + (.05<p<.10)

事前のアンケートの「学校の勉強が難しい」の結果と5回目の授業の「学校の勉強が難しい」の結果の平均を比較したところ差があり有意傾向である。この数値の差は、5回目の授業のアンケートの数値の方が学校の授業が難しく感じたということである。

考察については次の第5章で述べる。

## 4.6 小学生が作成したプログラム

授業を通して様々なメソッドを学んだ小学生が5回目のオリジナルプログラムで作成たものは、ゲームプログラム、クイズプログラム、タイピングプログラムなどである。

このプログラムはクイズプログラムで学校の授業で習ったことをクイズにし、回答時間を出力したものである。ポイントは、条件分岐である。それぞれの問題に4つの選択肢があり、回答を

キーボード入力から受け取り、選択したものによって出力するものを変えている。ただ、出力させるのではなく、sleep を使い工夫をしている。

```
# coding: utf-8
#!/uer/bin/env ruby
# coding: utf-8

puts("(9*7)/(2+1)の答えはいくつでしょう")

puts("1:32")
puts("2:24")
puts("3:21")
puts("4:31")
print("答えを入力:")
  answer = gets.to_i

sleep(1)
puts("結果は……")
sleep(1)

if answer == 1 then
  puts("ザンネン!ハズレ!もう1度やってみよう!")
elsif answer == 2 then
  puts("ハズレだよ!おいしい!")
elsif answer == 3 then
  puts("正解だよ!すごいね!!")
elsif answer == 4 then
  puts("残念!はずれだよ!")
else
  puts("その番号はないよ!!")
end

puts("")
puts("4と9の最小公倍数はいくつでしょう")

puts("1:24")
puts("2:27")
puts("3:32")
puts("4:36")
print("答えを入力:")
  answer = gets.to_i

sleep(1)
puts("結果は……")
sleep(1)

if answer == 1 then
  puts("ザンネン!ハズレ!もう1度やってみよう!")
elsif answer == 2 then
  puts("ハズレだよ!おいしい!")
elsif answer == 3 then
  puts("残念!次がんばろう!")
elsif answer == 4 then
  puts("正解だよ!すばらしい!!!")
else
  puts("その番号はないよ!!")
end
```

```
end

puts("")
puts("日本列島の春の季節のだいたいの雲の動きは")

puts("1:南から北")
puts("2:西から東")
puts("3:東から西")
puts("4:北から南")
print("答えを入力:")
  answer = gets.to_i

sleep(1)
puts("結果は……")
sleep(1)

if answer == 1 then
  puts("ザンネン!ハズレ!もう1度やってみよう!")
elsif answer == 2 then
  puts("正解!絶好調!!!!")
elsif answer == 3 then
  puts("残念!次がんばろう!")
elsif answer == 4 then
  puts("ザンネン!ちがうよ!")
else
  puts("その番号はないよ!!")
end

puts("")
puts("日本列島は何個の島が集まって出来ているでしょうか")

puts("1:4こ")
puts("2:6こ")
puts("3:7こ")
puts("4:5こ")
print("答えを入力:")
  answer = gets.to_i

sleep(1)
puts("結果は……")
sleep(1)

if answer == 1 then
  puts("正解だよ!完璧!!!")
elsif answer == 2 then
  puts("ザンネン!ちがうよ!")
elsif answer == 3 then
  puts("もう1度確認しよう!!")
elsif answer == 4 then
  puts("おしい!!!!もう1度やってみよう!")
else
  puts("その番号はないよ!!")
end
```

このプログラムは latter という配列の中に入っている “グラードン”, “カイオーガ”, “レシラム”, “

キュレム”の4つをどのくらいの時間で入力できるかというプログラムである。ポイントは、繰り返し処理で条件があったときに、配列の中にある問題を順番に入力させるものが変わるようになっている。最後にかかった時間を出力できることである。

```
#!/usr/koeki/bin/ruby
# -*- coding:utf-8 -*-

letter = ["グラードン","カイオーガ","レシラム","キュレム"]

print("どれだけ早くポケモンを打てるかな\n")
printf("%s\n", letter)

start = Time.now.to_i

i = 0
while i < letter.length
  printf("\v%s:", letter[i])
  input = gets.chomp
  if input == letter[i]
    stop = Time.now.to_i
    STDERR.print("\t正解\n")
    i += 1
  end
end

printf("かかった時間 %d 秒!\n", stop - start)
```



## 第5章 プログラミング教育の考察

本章では、第4章のアンケート結果から考察を述べる。

### 5.1 プログラミングについて

プログラミングのイメージは楽しそうというイメージがある。しかし、難しそうというイメージもあった。授業をしてまたプログラミングをしてみたいという回答が多かった。実際にプログラミングを学んでみて難しかったという回答があった。内容が難しく感じたのは、1.2で述べたように発達段階の要因があるのではないかと考える。また、内容が難しく感じたという理由だけではなく、キーボード操作も難しく感じた要因ではないかと考える。

### 5.2 小学生に身につけてほしい力

Ruby teraこっただで参加した小学生に身につけてほしい力についてアンケート結果、授業の様子、作成したプログラムから考察を述べる。

#### 知識及び技能

5回目の授業で自由に作成したプログラムを見ると授業で学んだことを理解し、工夫してプログラムを作成していた。そのことから、知識及び技能が身についたと言える。

#### 思考力、判断力、表現力等

毎回の授業でプログラムを作成するときに、分からないことがあると教科書を見て考えたり、大学生に質問をしたりしていた。また、もっとプログラムを良くするにはどのメソッドを使えばいいのか判断していた。そのことから、思考力、判断力、表現力等の力が身についたと言える。

#### 学びに向かう力、人間性等

プログラムを集中して作成したり、授業中に分からない所があった時に周りの大学生に質問をしたりして解決をしていた。また、難しい内容の部分でも積極的に学ぼうとする姿勢があったことから、学びに向かう力・人間性等が身についたと言える。

#### 工夫する力

毎回の授業でサンプルのプログラムを自分で考えて作成していた。5回目の授業のオリジナルのプログラムを作成する時に前に作成したものを改良して作成ができていたので、身についたと言える。

## 伝える力

自分が作成したプログラムの工夫した点や頑張ったところなど発表することができていた。また、他の人が作成したプログラムに対しての感想やどうやったらこの動きになるのかなど質問をお互いに行っていたので、力が身についたと言える。

### 5.3 論理的思考について

論理的思考についての結果 4.4.2 節では身についたと実感している小学生が多いという結果になった。5回目の授業で今まで学んだものを使い作成するときにもどのようにしたら動くのかを考えて作成している小学生が多かった。しかし、4.4.2 節で述べたようにプログラミングが難しいという結果も出た。

### 5.4 波及効果について

仮説では学校の勉強が好きになり、難しく感じなくなるという変化があるのではないかと考えたが実際には変化がなかった。変化がなかった原因としては以下3つの原因があると考えられる。1つ目は、回数が少ないことである。プログラミング教室開催が全5回と回数が少ないため変化が現れなかったと考える。2つ目は、対象人数が少ないことである。今回の調査対象12人中、勉強が好きな項目で数値が変化しなかったのは7人、数値があがった人は2人、数値が下がった人は3人であった。勉強が嫌いの項目では、変化しなかったのは10人、数値があがったのは1人、数値が下がったのは1人であった。変化しない参加者が多く数値に変化が現れなかった。3つ目は、質問方法に問題があったことである。学習の部分だけでは、波及効果が見られなかった。しかし、自由記述では、4.4.4 で述べたように、もっと難しいプログラムを作る、自分でゲームを作りたい、みんなにプログラミングを教える、50m クロールで泳ぎたいなどの回答があった。そのような回答から波及効果があったのではないかと考える。そのため、学習の項目だけに変化がなかっただけで質問方法に問題があったのではないとも言える。

### 5.5 t 検定の結果

有意傾向の結果がでたのは、勉強が難しいという項目であった。小学校の学習内容よりも今回のプログラミング教室の内容が難しく感じたためこのような結果がでたと考えられる。また、プログラミング教室の開催期間が夏休み中であり、学校で授業をしていない期間であるため、実感をもって答えられなかった可能性がある。



## 第6章 手引作成について

1.3節で述べたようにプログラミング教室は、年々増加している。しかし、プログラミング教室を企画・運営をするには「どの人を対象とするのか」、「道具は何を準備するのか」、「教える体制、準備期間など」を決めなければならない。しかし、そのような情報について詳しくまとめているものがWeb上で公開されていない。そのため、プログラミング教室を運営するための手引が必要であると考えた。Ruby てらこったの授業でのアンケートの結果、アンケート分析、授業の様子、作成したプログラムからプログラミング教育を通して、力を身に付けることができたという結果になった。そのことから、Ruby てらこったの活動を基準としたプログラミング教室の手引を作成した。

### 6.1 手引の作成の概要

Web サイトにてプログラミング教室を開催するための方法が公開されている [14][15][16]。公開されている情報はプログラミング教育の目的、使用教材、所要時間などである。しかし、その情報では足りない部分があると考えた。例えば、小学生にプログラミングを教える体制、教える方法など具体的な内容などである。そのため、実際にプログラミング教室を開催する企画を立ててから授業を行うための手引が必要であると考えた。

本研究で作成する手引は、実際に Ruby などを用いたプログラミング教室を企画し、運営していくのに利用してもらうことを目的としている。そのために、企画するうえでの計画の立て方、プログラミング教室を通して身につけてほしい力、小学生に教える範囲の決め方などを述べている。運営では、教えるために必要な役割、授業の準備、教えるための方法を述べている。

### 6.2 手引の構成

Ruby てらこったの活動の企画、運営方法、授業の進め方をまとめ、はじめてプログラミング教室を運営する人を対象にした手引を作成した。

#### 1章はじめに

プログラミング教育必修化の背景、プログラミング教室、プログラミング教室の課題についてまとめてある。

#### 2章プログラミング教室実施のための必要項目

教える側が必要なこと、プログラミング教室の目的と小学生に身につけてほしい力、授業・日程についてまとめた。

### 3章運営

運営方法、必要な道具(PC、USBメモリ、教科書、ノート、ホワイトボード、マグネットシート、アルファベット対応表)、授業準備、各授業のサンプルプログラム、模擬授業、他の機関との調整についてまとめた。

### 4章授業をするための方法

当日の準備・確認事項、授業の流れ、プログラムの発表方法について、授業後のミーティングについてまとめた。

### 5章授業を受講した小学生の感想

アンケートから、参加した小学生が授業で楽しいと感じた部分、難しいと感じた部分についてまとめた。

### 6章教えるための要点

教える上で注意すること、その他(授業の内容が予定より進まなかった場合、参加者が来れなくなった場合、利用しているPCの調子が悪くなった時についてまとめた。

### 7章実際に行うための一例

Ruby てらこったの活動を踏まえて企画、運営を円滑に行うための提案についてまとめた。内容は、企画、運営方法、広報関係の提案、ミーティング方法、申し込み方法の提案授業、6月に行った準備日程についてまとめた。

### 8章課題の考察

プログラム教室を運営するための課題(プログラミングツールに関する課題、プログラミング環境に関する課題、プログラミング教育を行う教員の課題)の考察についてまとめた。

### 付録

授業で使用した、コマンド集、アルファベット対応表を添付している。

## 第7章 結論

### 7.1 結論

参加した小学生は、プログラミング教育を通して、「知識及び技能」、「思考力、判断力、表現力等」、「学びに向かう力、人間性等」、「工夫する力」、「伝える力」が身に付けることができたという結果になった。

### 7.2 今後の展望

今後の展望については、以下の通りである。

はじめてプログラミング教室を運営をする人に手引を利用してもらう

実際に手引を利用してもらいプログラミング教室を運営してもらう。そこで情報が不足していたり、改善をしたほうがよい場合は手引を修正する。

アンケートの改善

今回はアンケートと作成したプログラムにより身についた力の判断をした。プログラムができてそれぞれのメソッドについてしっかり理解しているのか分からない部分がある。そこでアンケートに加え、学習した用語を実際にテストのように書いてもらう方法で判断し検証をする。

### 7.3 手引

Ruby てらこったでの活動でプログラミング教育の効果があつたという結果になった。そのことから、Ruby てらこったを基準に手引を作成し、付録と添付する。

## 謝辞

本研究は、平成30年度私立大学ブランディング事業「地域資源を産業振興やIT技術により活用できる人材の育成」の助成を受けた成果である。酒田市教育委員会、鶴岡教育委員会、参加してくれた方など Ruby てらこったの活動に協力をしていただいた方に感謝を申し上げる。

## 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 (第二版)”. [http://www.mext.go.jp/component/a\\_menu/education/micro\\_detail/\\_\\_icsFiles/afiedfile/2018/11/06/1403162\\_02\\_1.pdf](http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afiedfile/2018/11/06/1403162_02_1.pdf), (参照日 2019-5-24).
- [2] サトウタツヤ 渡邊芳之. 「心理学・入門ー心理学はこんなに面白い」, 株式会社有斐閣, 2011, p268.
- [3] 総務学省. “プログラミング人材育成の在り方に関する調査研究報告書”. [http://www.soumu.go.jp/main\\_content/000361430.pdf](http://www.soumu.go.jp/main_content/000361430.pdf), (参照日 2019-8-1).
- [4] 室谷心. “小学生にプログラミングを教える”. 松本大学研究紀要 (2013)p269-281.
- [5] 田口雄太 杉浦忠男 中島厚秀. “プログラミング教室を通じた相互成長ー宇城市プログラミング教室ー”. 崇城大学紀要 (2019)p115-122.
- [6] 室谷心 矢野口聡 浅見 (林) 大輔. “楽しさで引っ張るプログラミング入門講座 : JavaScript を使った 1 日プログラミング教室用教材の開発と試用”. 松本大学研究紀要 (2018)p75-82.
- [7] 廣田千明 寺田裕樹 橋浦康一郎 渡邊貫治. “地方大学における学生主体の子ども向けプログラミング教室 ー秋田県における IT 教育の推進ー”. 秋田県立大学ウェブジャーナル A (地域貢献部門) (2017)p71-80.
- [8] 河原和好. “小学生を対象にしたプログラミング教育について”. 新潟国際情報大学情報文化学部紀要 (2017)p27-35.
- [9] 文部科学省. “私立大学研究ブランディング事業”. [http://www.mext.go.jp/a\\_menu/koutou/shinkou/07021403/002/002/1379674.htm](http://www.mext.go.jp/a_menu/koutou/shinkou/07021403/002/002/1379674.htm), (参照日 2019-9-18).
- [10] 東北公益文科大学. [https://www.koeki-u.ac.jp/news\\_topics/branding-saitaku\\_2017.html](https://www.koeki-u.ac.jp/news_topics/branding-saitaku_2017.html), (参照日 2019-9-18).
- [11] 広瀬雄二. 「Ruby プログラミング基礎講座」, 技術評論社, 2006, p312.
- [12] 大石桃菜 佐々木大器 山口円馨. “東北公益文科大学における小学生向けプログラミング教室「Ruby てらこった」の取り組み”. 文部科学省 私立大学研究ブランディング事業 日本遺産を誇る山形県庄内地方を基盤とした地域文化と IT 技術の融合による伝承環境研究の展開 (平成 29 年度～平成 33 年度)p50-54.
- [13] 田中敏 Nappa. “js-STAR”. <http://www.kisnet.or.jp/nappa/software/star/>, (参照日 2019-12-3).

- [14] Progra!. “小・中学生向けプログラミング教室を開業（開校）するための準備と手順のポイント”. <https://progra.org/blog/65/>, (参照日 2019-11-14).
- [15] NPO 法人 Ruby プログラミング少年団. “一日 Ruby プログラミング体験教科書”. <https://github.com/smalruby/smalruby.jp/tree/master/source/files>, (参照日 2019-11-14).
- [16] 総務省. “一日 Ruby プログラミング体験教科書”. [http://www.soumu.go.jp/programming/data/017/017\\_01\\_01.pdf](http://www.soumu.go.jp/programming/data/017/017_01_01.pdf), (参照日 2019-11-14).

## 付録