Ruby等を用いたプログラミング教室の手引

亀谷千香子

令和元年9月

目 次

第1章　はじめに

2020 年度に小学生のプログラミング教育が必修化になる。文部科学省の小学校プログラミング教育の手引では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」が挙げてれいる。

小学生のプログラミング教育の必修化前である現在もプログラミング教室は開催されている。しかし、プログラミング教室を企画運営するための方法を手引として公開されているが、企画・運営のための情報が少ない。そのため Ruby 等を用いた小学生向けのプログラミング教室を運営をするための事前の準備、教えるための道具、教える方法などのまとめを手引として提案する。

1.1　プログラミング教育必修化の背景

小学生プログラミングの必修化に向けて文部科学省では小学校プログラミング教育の手引などを公開している。プログラミング教育を導入する理由として、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、生活を便利にしているからである。コンピュータはプログラムで動いているため、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だからである。また、プログラミング教育は子供たちの可能性を広げることにもつながるからである。

1.2　プログラミング教室

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。プログラミング教室で学ぶ内容、プログラミング言語などについては 1.2.1、1.2.2 で説明する。

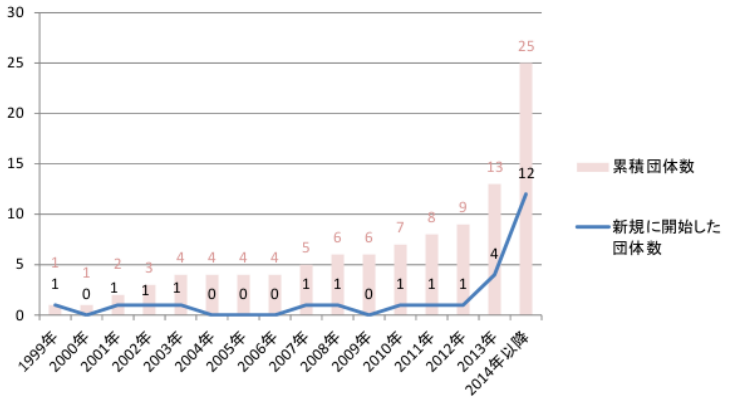

図 1.1: プログラミング教室始時期 [1] 図 5-3 より引用

1.2.1　内容

プログラミング教室で使われているものは大きく分けるとテキスト型、グラフィック型がある [2]。まず、テキスト型、グラフィック型それぞれのメリット、デメリットについて述べる。

テキスト型

文字を並べてプログラムを作成するものである。

(例) ドリル, JavaScript, Ruby, Python, java, C など

表 1.1: テキスト型

○/△	理由
○	概念を表す語を追加することで表現力が大きくなる
○	入力、修正が比較的に容易
○	複雑なプログラムも作成可能
△	書き方を覚える必要がある
△	すぐに見てわかりにくい

グラフィック型

画面上でプログラムを作成する ものである。

(例) スクラッチ, ビスケット, アルゴロジックなど

表 1.2: グラフィック型

○ /△	理由
○	書き方、規則を覚える必要がない
○	文字を使わなくて良いため誰でも簡単に作成できる
△	修正に時間がかかる
△	複雑なプログラムを作成するのに限界がある

1.2.2　プログラミング言語

プログラミング言語とは、コンピュータへ指示を出すためのプログラムを作成するのに使われている言語である。プログラミングを教えるときは、教える人の対象、プログラムの難易度、参加者に作成したいプログラムに合わせてプログラミング言語を決める必要がある。主に利用されている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。

ドリル

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。

Viscuit
原田康徳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。

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。

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。

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。

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。

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。

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。

Python
Guido van Rossum によって開発され、全米の大学では初心者に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。

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。

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 アプリも作成可能である。

1.2.3 プログラミング言語の割合

総務省によるとプログラミング教室で使用されているプログラミング言語の割合は Scrach が最も高い (図 1.2)。その理由は、短期間での学習に適してると考えるためと述べられている [1]。

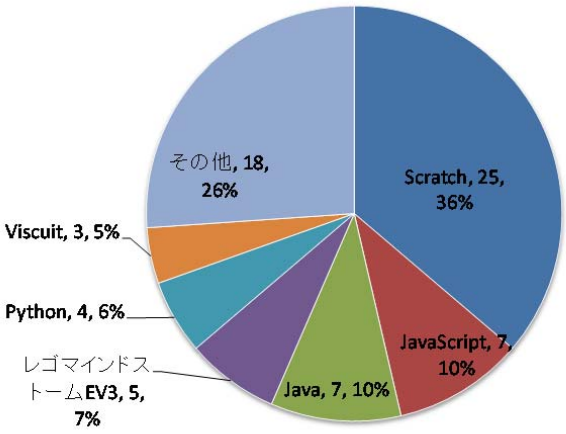

図 1.2: 教室・講座で利用されているプログラミング言語 [1] 図 5-11 より引用

1.2.4　使用されている機器

プログラミングで使用されているものの割合は図 1.3 である。多くのプログラミング教室では PC が多く使用されている。その他にもラズベリーパイという小型のボードコンピュータである 1.4。Linux などの OS を搭載すれば、インターネットアクセスができるものである。

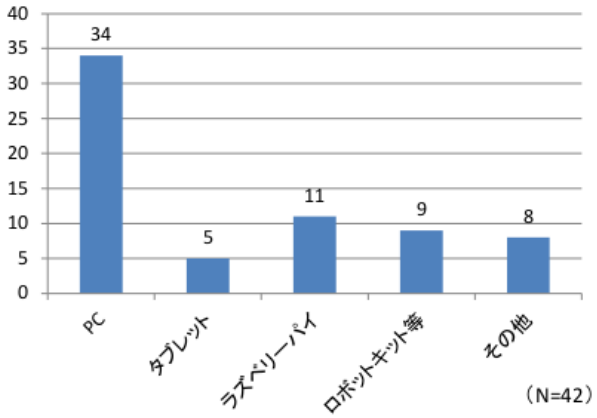

図 1.3: 教室・講座で使用されている機材 [1] 図 5-14 より引用

1.3　プログラミング教育の課題について

プログラミング教育では、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などが報告されている [3]。プログラムを分かりやすく指導するとなると

図 1.4: ラズベリーパイ

道具、教材のだけではなく、行う場所、指導する人などの課題がでてくる。その点をふまえて 8 章では課題とプログラミング教室の企画、運営をした結果の考察をまとめた。

1.3.1 手引作成の背景

多くの 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。公開されている情報としてはとしてはプログラミング教育の視点、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多くあると考えた。例えば、小学生にプログラミングを教える体制、教え方の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を教え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。

1.3.2 対象読者

この手引は、プログラミング教室を企画、運営をしようと考えている人である。1.3.1 で述べたようにプログラミング教室を行うための情報が不十分である。そのため、準備方法、広報、他の機関との連携についてまとめている。

1.3.3 手引の構成

はじめてプログラミング教室を企画、運営する人にも分かりやすいように決める順番で述べている。そのため手引の構成としては以下のようになっている。

- プログラミング教育の実施のための必要項目

  活動する人の必要な知識や能力、プログラミング教室を行う目的、小学生に身につけてほしい力、日程の立て方について

- 運営

  企画したものを運営するための役割、業務内容、授業準備について

- 授業をするための方法

  授業の流れ、当日の準備について

- 参加した小学生の感想

  全 5 回の授業に参加した小学生が楽しかったと感じた部分、難しいと感じた部分について

- 教えるための要点

  小学生に教える時の授業担当者、アシスタントそれぞれの注意点について

- 行うための一例

  Ruby てらこったの活動のように企画、運営をするために何にどのくらいの時間をかけて準備しているのか、円滑の活動を行うための方法について

第2章　プログラミング教室実施のための必要項目

本章では、プログラミング教室実施のために必要な教える人の知識・必要な力と計画の立て方について述べる。教える人の知識・必要な力については、組織の利点を活かしてどのようにプログラミング教室を企画するかを述べていく。計画の立て方については、プログラミング教室を行うための目的に沿って効果的に進めるための方策について述べる。

2.1　教える側が必要なこと

プログラミング教室の活動を行う時に必要で必要なこととして「知識」「教える能力」「発想力」「時間管理能力」「伝える力」4 つが必要であるのではないかと考える。プログラミング教室で教える場合はただプログラミングが得意である人を優先するのではなく、小学生に合わせて楽しく学んでもらえるように工夫できる人を優先した。その理由は以下のような理由である。

2.1.1　プログラミングの知識

プログラミング教室で実際に教える時は小学生の質問に対して分かりやすく教えることができるのが前提になっている。そのために、ただ内容を理解しているだけではなく、自分なりに考えて教えられる程度に理解していなければならない。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という必修科目がある。その科目でプログラミング言語の Ruby を学ぶ。1 年間で学ぶ内容は以下の内容である [4]。本学の授業では 13 の範囲まで学ぶが、Ruby てらこったで教える授業の範囲は 1 から 7 までである。

1. 計算システムの基本操作と概念

   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

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の基礎

3. 変数と値

   変数と値処理の方法

4. 演算子

   プログラミングの加減乗除の手法

5. 制御構造

   プログラムで命令が実行される流れを定める手法

6. 出力処理を行うメソッド

   決めた複雑な処理を行う手法

7. 配列

   多くのデータをまとめて保持する手法

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する手法

9. ファイルの入出力

   数にファイルを指定した場合はそのファイルから 1 行ずつ内容を読み込む手法

10. 計算機の内部構造

    2 進数、16 進数、文字コードについて

11. ハッシュ

    任意の値に任意の値を結び付ける手法

12. 再帰

    ハノイの塔のような再帰処理の仕組みと手法

13. CGI

    HTML 文書内にあるデータ入力窓とそれらを受け渡すスクリプト名を書いておき、ボタンを押すとそのスクリプトに入力値が渡るような手法

2.1.2　教える能力

小学生の一人一人理解している部分、キーボードの操作の速度などが違うため、それぞれに合わせて教える必要がある。小学生とコミュニケーションを取りながら分かりやすく教えるため、Ruby てらこったのメンバー募集は小学生と関わること、教えることが比較的に得意な学生を優先的にした。例えば、教職課程を履修している学生、学習支援サークルとして小中学生に学習を教えている学生、2 年次の必修の基礎プログラミングのティーチングアシスタント (以下 TA) をしている学生などである。

2.1.3　発想力

発想力では、どうしたら小学生がプログラムを楽しく、分かりやすく学んでもらうことができるのかをイメージしてアイディアを出す必要がためある。例えば、誰かが授業で使う資料を見せてきた時にこうしたらもっと良くなるなど意見を 1 つ出すだけでも分かりやすさというのが変わってくる。また、小学生の質問に対してどうしたら分かりやすく伝わるかなど考える必要がある。

2.1.4 時間管理能力

業務で割り振られたものを完璧に締め切りまでで完成させるためには、どのくらいの準備時間がかかるのかということを意識しなければならない。締め切りで提出すればいいという意識で作業をすると良い物ができない。そのために、ただ作業をするのではなく、他のメンバーから確認してもらい改善、修正する時間を作るということが必要になってくる。作成物の修正の時間のことを考えて余裕をもって作成をしなければならなので逆算をして物事を考えてくる必要がある。また、作業以外にも他にしなければならないことがあると考えられるため時間の管理というのを意識していくことが大切になる。

2.1.5 伝える力

ここで使う伝える力とは、正確に相手に伝えるということを意味している。どの業務も自分一人だけで作業できることは少ない。そのため、メンバーで協力、分担しながら作業を進めていくことのほうが多い。そのためどの作業がどこまでできているのかなど具体的にメンバーに伝えるということは大切である。

2.2 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

プログラミング教室を行う目的とプログラミング教室を通して身につけてほしい力について述べていく。

2.2.1 Ruby てらこったと活動の目的

昨年度から大学のブランディング事業の一つの地域資源を活用する人材育成の研究としてプログラミング言語の Ruby を用いたプログラミング教室の企画・運営を行ってきている [6]。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。小学生のうちからプログラミング言語 Ruby を用いた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ているためである。小学生でも対象は小学 5、6 年生にした。その理由としては、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である。

授業は大学生が教える。また、小学生 1 人に対して大学生が 1 人でサポートできるような体制にした。その理由としては、小学生一人一人に合わせて教えることができるようにしたいと考えたからだ。授業内容は比較的に簡単な内容であり、授業を通して簡単なゲームプログラムを作成できるように内容を組み立てた。

2.2.2 授業を通して身につけてほしい力

小学生の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 [5]。その 3 つの力に追加して Ruby てらこったで身につけてほしい力についても述べる。

知識及び技能

生活でコンピュータが活用されていくことや、問題の解決には必要な手順があること

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

この項目に加え Ruby てらこったで身につけたい力は以下の通りである。

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つくれるようになること

伝える力
: 伝える力としては、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点、頑張った点など伝えられるようになれること

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伝えられるようになること

2.2.3　教える範囲

教える範囲としては小学生が比較的に理解しやすいところであり、例えばクイズ、くじ引きのゲームを自分の力で考えられるようにするために必要なものを教える範囲とした。

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と判断した。

文章処理メソッド (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と判断した。

制御構造 (while, if, elsif, else)
: 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に必要と判断した。また、クイズの結果の判定や条件で繰り返しを行うために使うため必要であると判断した。

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と判断した。

sleep 関数 (sleep)
: プログラムを時間を指定して一時停止することができるので小学生が楽しむことができるのではないかと判断した。

2.3　授業・日程

小学生に教える順番、授業時間の設定、開催日時について述べる。

2.3.1　教える順番と教える時間

基本的な操作から徐々に難しい内容になるよう教える順番を設定している。また、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。

教える時間は 1 回の授業で 2 時間としている。小学生が集中して授業ができるように大体 45 分から 50 分ぐらい授業をしたら約 15 分位の長めの休憩を入れるようにした。

表 2.1: 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

2.3.2 各小学校のとの調整

地域の小学生に参加してもらうために各小学校に資料を配布する。資料には、何をするのか、日程、問い合わせ先、申し込み方法などの内容をまとめる。資料配布では教育委員会、小学校に開催するプログラミング教室の内容日程など説明する。

2.3.3 開催日程

全 5 回の授業に参加し学んでほしいことから、小学生がプログラミング教室に参加しやすくするために学校が基本休みである土曜日、日曜日や学校・地区の行事と重ならないように開催日程を設定した。

第3章　運営

本章では、プログラミング教室を運営をするために必要な組織の構築、授業準備について述べる。組織の構築では、役割、業務内容などの運営するための方法について述べる。授業準備では、小学生に楽しくプログラミングを学んでもらうための必要な準備について述べる。

3.1　運営方法

開催に先立ち、それを運営する組織を作る必要がある。ここでは、Ruby てらこったでの構築例をもとに、実際の活動を円滑に進めるための事前準備、主な役割、業務内容、ミーティング方法について述べる。

3.1.1　役割関係の明確化

活動をする時に、スムーズに作業ができるように役割を明確にする必要がある。Ruby てらこったでの主な役割は図 3.1 の中で説明する。

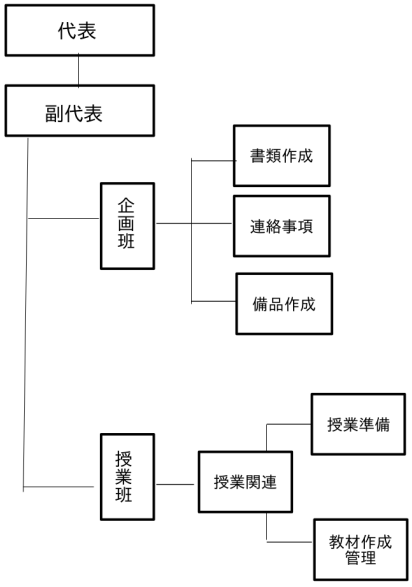

図 3.1: 組織図

リーダー

リーダーの役割は全体的に計画、作業の進行状況などを把握、ミーティングの進行を担うことをする。

全体の計画では、いつまでに何をするのかを明確にする。明確にしたらグループ全員で内容を吟味する時間、内容修正する時間などを含め逆算し、作業の締め切りを決める。その時に、締め切りを決めてそのままにするのではなく作成を頼んだ人に対して、締め切りの何日か前に中間成果物を見せてもらうようにするなどして作業の進み具合を把握する。

ミーティングをするときは何を話し合うのか決めておき、あらかじめメンバーに伝える。作成した成果物に対して全員からどうしたらもっと良くなるのかを意見をもらう時間にする。修正したものをいつまでに共有するか決める。例えば、授業用の教材の場合は、小学生でも分かりやすいか、もっと分かりやすくできないかなど意見をもらう。その後、次のミーティングまで修正した教材を作成するということをした。また、業務の割り振りなどもする。業務の割り振りでは、メンバーの話を聞き、それをふまえて的確に割り振る必要がある。ミーティングの流れは図 3.2 で説明する。

授業担当

業務は以下のようなものがある。

- 教材作成

  担当の授業で使用する教材を作成する。教材としては、スライド、サンプルプログラムなどである。詳しくは 3.3 で述べる。

- 模擬授業

  実際の授業を想定し、タイムテーブルを確認しながら行う。自分では分かりやすく教えているつもりでも相手には分かりにくい部分などは気づきにくいので、多くの人から意見をもらう機会を作る。

- 授業

  小学生の様子 (操作はできているか、指示が伝わっているか) を見ながら分かりやすく丁寧にを意識して授業を行う。

各授業のアシスタント

小学生一人に対して一人のアシスタントをつける。業務は以下の内容である。

- 小学生が分からない所の説明

  授業の内容が難しかったり、分からなかった場合ににサポートしている小学生に合わせて教えることをする。

- 操作の補助

  キーボード操作が遅い場合がある。その時に、キーボードの位置を教えたり、コマンドを教えたりする。

タイムキーパー

タイムキーパーは、各授業のタイムテーブルに合わせて授業担当者に時間を伝える。そのため、授業の進み具合など確認して授業が円滑に進むようにサポートする。時間に余裕がある時は、プログラム作成の時間や発表時間などにするなど休憩中に打ち合わせする。反対に、時間が想定よりも足りない時は、変更する部分など提案してなるべく時間内で授業が終わるように考えて動く必要がある。

連絡係

連絡係は参加する保護者の方にメールで連絡を取ることをする。はじめに送ったメールに対して返信してもらい連絡が取れることを確認する。メールで連絡をするので誰が読んでも正確に伝わる内容を早くお知らせすることを意識して業務をする必要がある。また、メールで質問が来た時も対応をする。

3.2 必要な道具

授業で使う道具は以下のようなものを用意した。

表 3.1: 必要な道具一覧

アイテム	用途
PC	小学生の操作用、授業用、スライド用のため使用
USB メモリ	作成したプログラムを記録するため使用また、教室閉講後にも利用できよう 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ため使用
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	プログラムを作成するときに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に分かりやすくするため使用

3.2.1 PC

PC は小学生用、授業用、スライド用の 3 種類使用した。小学生用と授業用の 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、本学のパソコンと同じ OS だと学生が教えやすいと考えたからである。また、授業以外でも使ってほしいと考えたため無料で配布できるオープンソースがよいからである。

- 小学生用

  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

- 授業用

  小学生の使用しているパソコンと同じであるためどのように操作するのか小学生に分かりやすくするため使用

- スライド用

  スライド用を用意することで画面の切り替えすることができるためスムーズに内容の説明ができる

使用した教室に 2 つ独立したプロジェクタがあったので授業用、スライド用両方を映して授業を行った。2 つプロジェクターがなかった場合はテレビに映して授業をした。

3.2.2 USB メモリ

今回は 16GB 以上の容量を持つ高速なメモリに 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。

3.2.3　教科書

授業内容に合わせて、学生らで作成した。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
: プログラミングとは何か、Ruby とは何かを説明する

ファイル、ディレクトリ
: ファイルとディレクトリの説明をする

  小学生でも分かるようにファイルは、絵を書く時の紙でディレクトリは紙をしまっておく場所というようにした

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

if, elsif, else について
: 条件分岐をさせる方法、if, elsif, else の違いについて図を用いて分かりやすくした

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 (9.1 を参照)

3.2.4　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。授業内では、ノートに書く場面があれば先生が指示を促し小学生がノートを書く時間を設けた。

3.2.5　ホワイトボード

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わかりやすいと考え、用意した。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った。

3.2.6 マグネット

その授業の内容で勉強する新しいことや大切なものをマグネットにした。それらを貼るだけでよいようにし、記憶しやすくするために作った。これらを授業内でホワイトボードに貼りながら作業を進めていった。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。

3.2.7 アルファベット対応表

小学 5、6 年生はアルファベットを習い終えているが大文字小文字を理解するために作成し配布した (9.2 を参照)。キーボードでは大文字表記になっているため「 l(エル) 」と「 i(アイ) 」や「 h(エイチ) 」と「 n(エヌ) 」などが間違いやすかった。

3.3 授業準備

授業の準備では授業を行うまでの流れを述べていく。

3.3.1 内容の吟味

授業での内容をミーティングなどで話し合い内容を確認する。教える内容では、小学生でも分かりやすく学ぶことができるものを中心にしている。そのため、小学生にとって内容が難しすぎないか、変数などのイメージ図はどうするかなど話し合う。1 回分の授業で教える内容が多くなり過ぎないか、教えるプログラムは分かりやすいか、プログラム作成の時間が十分に取れるかなどを考える。教える内容が多くなりすぎないかという基準については模擬授業をしてみたり、各授業での小学生の理解度を見たりして内容を決めた。プログラムについては 3.4 で述べる。

3.3.2 教える体制

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。

3.4 各授業のサンプルプログラム

授業用のサンプルプログラムは授業の内容のポイントをおさえたものにした。授業中に説目を加え理解を深めてもらうようにした。

3.4.1　1 回目のプログラム

画面出力の print,puts を使ったプログラムを作成している。このプログラムでは、print と puts の違いを理解してもらうために両方使用した。画面に出力される ”” 部分の文字を変更する。速く操作ができる小学生には print または put を増やして好きな文字を書き込ませて変えるようにした。

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

print "名前\n"
print "学校名\n"
puts "好きな食べ物"
puts "好きな動物"
```

3.4.2　2 回目のプログラム

変数、while のループを使ったプログラムを作成している。gets.chomp を最初に追加して所持金を入力させたり、最後の合計に 1.08 をかけて消費税込みの金額を出したりなど変えることができる。

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

puts "スーパーのレジ買ったものの値段を入れてね。(100 円→100 と入力しよう)"
memo = 0

while true
  print "値段は? (終わりたい時は q を押してね)"
  nedan = gets.chomp

  if nedan == "q"
    break
  end

  memo += nedan.to_i
  printf("今の合計は %d 円だよ \n",memo)
end
printf("今回の合計は %d 円です。お買い上げありがとうございました! \n",memo)
```

3.4.3　3 回目のプログラム

配列と乱数を組み込んだプログラムをサンプルにした。この配列の部分をグー、チョキ、パーに変えるとジャンケンプログラムに変えることができる。

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

kuji = ["1等","2等","3等","4等","ハズレ"]
print("なにがでるかな～？")

puts("....")

sleep(1)
srand()
nani = rand(5)
printf("あなたは%sでした！\n",kuji[nani])
```

3.4.4 4回目のプログラム

条件分岐の if を使いクイズプログラムをサンプルにした。クイズの問題文を変えるだけではなく選択肢を増やしたり、回答によって違うものを画面出力できるように変えることができる。

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

puts "今の元号はなんでしょうか？"

puts "1:平成"
puts "2:令和"
puts "3:昭和"

print "答えを入力:"
kotae = gets.to_i

sleep(1)

if kotae == 1 then
  puts "惜しい！ハズレだよ！"
elsif kotae == 2 then
  puts "正解だよ！読み方はわかるかな？？"
elsif kotae == 3 then
  puts "ザンネン！ハズレだよ！"
else
  puts "その番号はないよ！！"
end
```

3.4.5 5回目のプログラム

今まで学んだ内容を含めたプログラムをサンプルにした。

```
#!/usr/bin/env ruby

puts "コンピュータとじゃんけんをしよう!"
puts "5回中3回勝てればクリアだよ!"
sleep(3)
puts ""
print "それじゃ GAME START\n"
puts ""

win = 0
kaisuu = 1
janken = ["だしてないよー", "グー", "チョキ", "パー"]

while kaisuu <= 5
  sleep(2)
  puts ""
  printf("%d回目!(あなたの勝利数:%d)\n", kaisuu, win)
  puts ""
  print "じゃんけん!(グーなら「1」チョキなら「2」パーなら「3」を押してね): "
  while true
    you = gets.to_i
    if you >= 4
        you = 0
    end
    com = rand(3) + 1
    printf("ぽん!(あなたは「%s」でコンピュータは「%s」)\n",
           janken[you], janken[com])
    sleep(1)
    if (you==1&&com==2) || (you==2&&com==3) || (you==3&&com==1)
      puts "あなたの勝ち!"
```

```
      win += 1
      kaisuu += 1
      break
    elsif (you==1&&com==3) || (you==2&&com==1) || (you==3&&com==2)
      puts "あなたの負け!"
      kaisuu += 1
      break
    elsif you == com
      print "あいこで(グーなら「1」チョキなら「2」パーなら「3」を押してね): "
      redo
    else
      puts "なにもだしてないからあなたの負け!"
      kaisuu += 1
      break
    end
  end
end

sleep(3)
printf("あなたの勝利数は「%d 回」\n", win)
sleep(3)

if win >= 3
  puts"おめでとう!"
  sleep(2)
  puts "+--------------------------------------------------+"
  system 'banner YOU WIN!!'
  puts "+--------------------------------------------------+"
else
  puts "残念..."
  sleep(2)
  puts "+--------------------------------------------------+\n"
  system 'banner GAME OVER'
  puts "+--------------------------------------------------+\n"
end
```

3.4.6 スライド

授業で使うスライドはあくまでも授業の副教材であることを意識する。スライドに書く内容は図、重要な部分だけにする。するとポイントなどが小学生にも分かりやすく伝えることができる。

3.4.7 模擬授業

模擬授業では授業担当者が実際の授業を想定して行う。主に見るポイントとして以下のようなものを確認する。

- 教え方が分かりやすいか (プログラムの説明など)
- ノートを取る部分の確認
- スライドが見やすいか (スライドの配色、文字が多すぎないか)
- 時間配分 (プログラムの作成時間があるか、時間が無くなったらどこを省いて説明するか)

3.4.8 他の機関との調整

プログラミング教室を運営をするときは単独で行うことが難しいと考えるための他の機関との協力が必要である。例えば、プログラミング教室を行う前に、広報資料 (活動内容、プログラミング教室活動日時、募集条件、締め切り、問い合わせ先などの情報) 配布がである。資料配布のための募集期間、締め切りなどの時間を考えた。日程が決まったら本学から酒田市教育委員会に連絡をとり、教育委員会から各小学校に資料を配布して、その後に各小学校の先生から小学生に資料を配布して参加したい小学生から連絡をもらった。そのため他の機関との調整を行った。Ruby てらこったの場合は、各小学校、酒田市教育委員会と調整をして行った。

プログラミング教室を開催すると決めて参加者を募集する時に、プログラミング教室に参加したいと考える小学生がどのくらいなのかが分からず参加人数予想ができなかった。そのため、酒田市内の全ての小学校に募集をかけるのではなく、本学の近隣の小学校、小学生数が多い小学校という条件のもと小学校を 4 校に絞って募集をすることを決めた。その後小学校の先生にプログラミング教室の説明を行い許可を得て資料配布をした。参加募集期間の締め切りを資料配布から約 2 週間ほどにした。するとある程度の小学生が実際にプログラミング教室に参加申し込みをしてくれた。

7、8 月にまたプログラミング教室をおこなった。最初の 4 校の募集により各小学校からの参加人数が大体予想がついた。また、小学校が夏休み中であるため、参加可能人数を増やしたいと考えた。そのため酒田市の全部の小学校に募集をかけた。その時に全部の小学校を訪問、プログラミング教室の説明、資料を配布するのには時間がかかる。そのため、酒田市内の小学校に配布するのは、酒田市の教育委員に活動の説明を行い協力を得て配布してもらった。

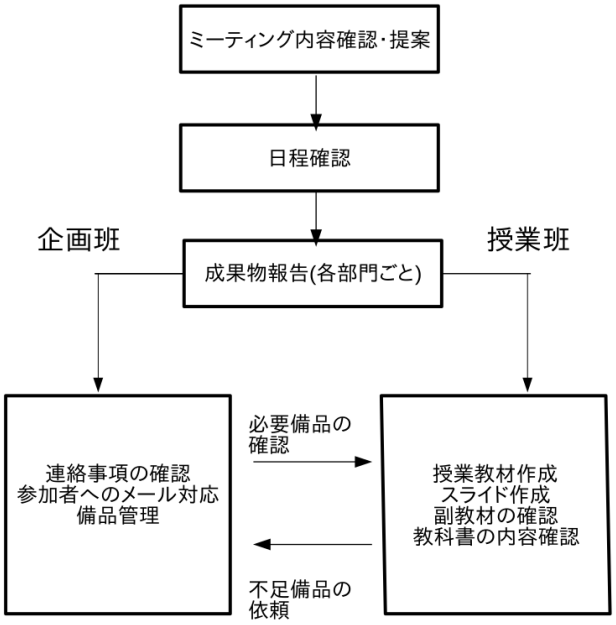

図 3.2: ミーティング方法

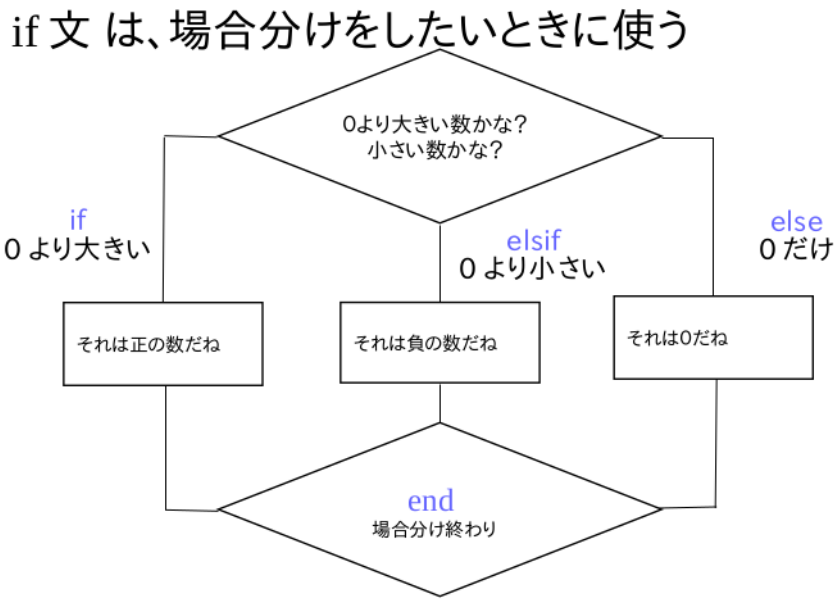

図 3.3: 条件分岐イメージ図

第4章　授業をするための方法

この章では、少し難しいプログラムの内容でも小学生に分かりやすく、楽しく学んでもらうための当日の準備作業、授業の流れ、プログラムの発表方法、授業後のミーティングについて述べていく。授業を進めるときはタイムテーブル通りに教えるだけでは内容を理解してもらうことが難しいと考える。そのために事前準備や確認作業を行う。授業では教えるということも大切でるあるが自分で考えたプログラムを作成するだけではなく発表することを大切にしている。その理由は 2.2.2 で述べたように伝える力を身につけてほしいからである。

4.1　当日の準備・確認事項

当日は、授業がスムーズに進めることができるように授業が始まる 1 時間ぐらい早く集合し準備する。まず、備品のホワイトボード、ペン、マグネット、PC をの設置などをする。PC の設置では、ネットワークが繋がるかを試す。その後、授業のおおまかな流れを確認する。

4.2　授業の流れ

授業の流れは以下の通りである。

1. 前回の授業の復習

   前回の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

   タイピング練習では、授業で使う英単語を 3 回ずつ練習する時間をとる。例えば、1 回目の授業であれば、print, puts などである。速く入力が終わった人には、5 回など入力する回数を増やすようにした。

3. 内容の説明

   内容の説明では、授業で新しく学ぶことを説明する。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

   サンプルプログラムは、新しく学ぶ内容を使う。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。休憩の間には、小学校の話し、習い事、趣味など小学生とコミュニケーションを取る。

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う。作成の時は、ホワイトボードにマグネットを貼ったり、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、考えてもらい理解してもらうということである。

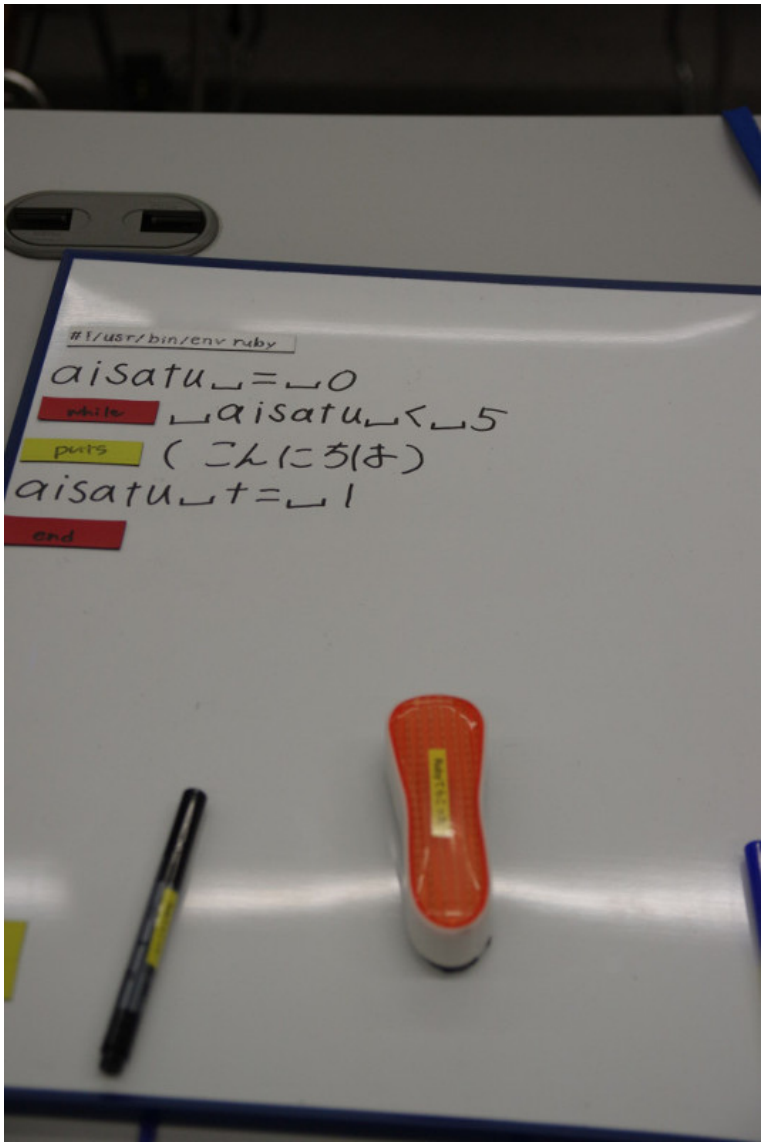

図 4.1: ホワイトボードに作成するプログラムを書き込んでいる様子

8. 作成したプログラム発表

   作成したプログラムは、発表する時間を確保する。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方法と他の人が作成したプログラムを実行してみる方法の 2 つである。次の 4.3 で詳しく述べる。

4.3 プログラム発表について

授業で作成したプログラムを発表する機会を設けた。その理由としては、自分の作成したものを発表し伝える力を身につけること、他の人の発表、プログラムの良さなどを考えてもらうためである。授業で行った発表方法は 2 つある。その方法は以下の通りである。

みんなの前で発表する

最初に作成したプログラムの難しかった点、頑張った所など話してもらう。プロジェクターに映し、作成プログラム実行をして発表する方法である。発表する時に何を話せばいいのか困ってしまう小学生もいるのであらかじめ話してほしい項目を提示すると困った様子が見られなかった。

他の人のプログラムを実行する

全員で座席を順に移動し、他の人の作成したプログラムをお互いに体験させる。この方法では、実際に実行できるので工夫している部分など気づくことが多い。実行してみた感想などお互いに伝えることができる。

4.4 授業後のミーティング

授業後に約 30 分ほどのミーティングをする。まずはじめに、授業の担当者から良かった点、悪かった点を先に話してもらう。悪かった点についてはメンバーから改善策、意見をもらう。そのあとに、授業のアシスタントから参加した小学生の様子、理解度についての意見をもらう。その後に次回までの授業の改善点、対応策、準備しなければならないことなどを話し合う。

第5章　授業を受講した小学生の感想

1回目から5回目まで授業後にアンケート (授業で楽しかったこと、難しかったこと) をとった。以下の表はそのまとめである。

表 5.1: 楽しかったこと

回数	内容
1回目	PCの操作、コマンド操作、自己紹介プログラムを発表したこと、大学生と話しをしたこと
2回目	キーボードの操作、みんなが作成したプログラムを実行すること、大学生と話をしたこと
3回目	みんなが作成したプログラムを実行すること
4回目	プログラムの作成、大学生と話しをしたこと
5回目	オリジナルのプログラムを作成したこと

表 5.2: 難しかったこと

回数	内容
1回目	PCの操作、自己紹介
2回目	break、ループが難しかった、プログラムを改造すること
3回目	乱数でランダムに取り出すこと、自由にプログラムを作ること
4回目	if,elsif,eleseの使い方が難しかった、プログラムを改造すること
5回目	変数が多くなりすぎて難しかった、オリジナルのプログラムを作成すること

5.1　楽しいと感じる部分

楽しいと感じた部分として発表が楽しかったという意見が多かった。発表を聞き、他の参加者のプログラムを動かしてみるのが新たな発見があり楽しかったのだと考える。

また、大学生と話しをするのが楽しかったという意見があった。プログラムを作成する時間や休憩の時間など多くの場面で話す機会があったのは良かったと考える。

5.2　難しいと感じる部分

操作の部分ではキーボードの操作が難しいと感想に書いていた。PCをあまり使わない小学生にとってキーボードを使い入力するには、どこに入力したい文字、数字があるのか探しながら打つので思ったよりも時間がかかるため難しく感じたのだと考える。

プログラムの内容では変数の部分が難しかったという感想が多かった。2 回目から 5 回目にかけて難しいプログラムを作成していくと、変数が徐々に増えていく。そのため、何をどの変数にしたか確認しながらプログラムを作成するため難しく感じたのだと考える。

5.3　参加した小学生が作成したプログラム

全 5 回の授業を通し学んだことを使用して小学生が作成したプログラムは以下のようなものである。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。

```
#!/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)
```

このプログラムは、フラッシュ暗算ゲームで rand で 0 から 10 までの数字をランダムに取り出して掛け算を計算するものである。

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

puts"フラッシュ暗算ゲーム!"
puts"表示された数字をかけた合計を出してね!"
kaisuu = 0
sum = 1
sleep 1
puts"よーい..."
sleep 2
puts"スタート!"
sleep 1

while kaisuu < 3
  srand
  kazu = rand(10)
  print kazu
  sum *= kazu
  sleep 1
  print"\b"
  kaisuu += 1
end
print"さて合計いくら? : "
start = Time.now.to_i
answer = gets.to_i
```

```
finish = Time.now.to_i
time = finish - start

if answer == sum
  puts"おめでとう ! 正解 !"
  printf("正解までのタイム: %d秒\n", time)
else
  puts"はずれー ! 残念 ..."
  printf("正解は %d\n", sum)
end
```

第6章　教えるための要点

この章では主に実際に小学生にプログラミングを教えるための注意点をまとめている。

6.1　教える上で注意すること

授業担当者と授業アシスタントの両方の視点から述べていく。

6.1.1　授業担当者

授業担当者は、小学生に教えるということで内容を噛み砕いて教えることが重要になってくる。授業は基本的に操作など全員一緒に進んでいくため常に**小学生の顔を見るように**する。すると操作が終わっているのか、まだ作業の途中なのかなど確認することができる。進み具合をみて、かなり遅れている場合はアシスタントに頼んでサポートしてもらう。授業中は、ただ話すのではなく、常に問いかけたり、手を上げてもらったりと反応を見るようにする。話を聞く、ノートを取る、ホワイトボードに書き込む、プログラムを作成するなど指示をしっかり出して、**何をするのか明確に**することが大切である。

6.1.2　授業アシスタント

授業アシスタント者は操作、授業で理解できなかった部分の説明など個人に合わせてサポートする。教える小学生に対して**目線を合わせて教える**ことが大切である。難しい内容でも分かりやすい言葉を使う。プログラム作成の時は、全てサポートするのではなく考えてもらう時間を大切にする。操作が遅くて焦ってしまう小学生がいた場合は、声をかけて、ゆっくりでも問題ないことを伝えて、確実に操作してもらうようにする。

6.1.3　小学生の理解度に合わせる

一人一人授業の内容の理解度、操作速度などは違う。そのため一人ひとりに合わせた教え方をしなければならない。

- 理解が早い小学生

  理解が早い小学生には、サンプルのプログラムを改造してもらうということをして、何もしていない時間をなくすようにした。最初にどの部分を変えると良いのかヒントを出しながら少しずつ教えていった。

- 操作が遅い小学生

  操作が遅いと追いついていかなければと焦ってしまう小学生がいる。そのときは、どこまでできているのか確認し、話しかけながら操作のサポートをする必要がある。

6.2　その他

その他として授業が計画通りにいかなかった場合、参加者が来れなくなった場合、PC が調子悪くなった場合の対応を述べる。

6.2.1　授業の内容が予定より進まなかった場合

どこを中心に教えるのかということを考える。説明に時間がかかりすぎてしまったときは、次の授業でどのぐらいの時間で説明するかのと打ち合わせをして教える範囲を全て 5 回で終わらせるようにする。

6.2.2　参加者が来れなくなった場合

全 5 回の授業で 1 つどこかの回の授業に参加できなかった場合は次回の授業の 30 分前に来てもらい簡単な授業を受けてもらい、その後次の授業に参加してもらった。すると、次の授業においていかれることなく参加してもらうことができた。

6.2.3　利用している PC の調子が悪くなった時

インターネットを使い Web ページを見る時に無線 LAN ではうまく繋がらない場合があった。そのときは、有線で接続をした。次の授業からは予備のパソコンを使い対応をした。

第7章　実際に行うための一例

Ruby てらこったの活動を通して良かった部分、悪かった部分があった。そのことを踏まえて企画、運営を円滑に行うための提案をする。プログラミング教室を企画から開催するまでの大まかな流れ、内容をまとめる。詳しい部分は前の章を参考にすると良い。

7.1　企画

企画ではこのような流れで役割、教える範囲などを決めた。

1. 活動するメンバーを集める

   活動をするにあたってどのようなメンバーが必要か考え集める。

2. 責任者の選出

   リーダー、副リーダー、企画班リーダー、授業班リーダーを決める。

3. プログラム教室の目的を明確にする

   プログラミング教室を通して小学生に何を学んで欲しいのか、身につけた力などを考える。

4. 教える範囲を決める

   教える範囲は小学生でも理解しやすい内容、最終的に小学生にどのようなプログラムを自分で考え作成できるようになってもらうのかを決める。

5. 役割を決める

   企画班 (書類作成係、連絡事項係、備品作成係) 授業班 (授業担当係、教材作成・管理係) を決める。

6. 授業について決める

   教える範囲をどの順番で教えていくか考える。それぞれの授業で教えるイメージ図についてなどを決める。

7. 日程について決める

   日程は全 5 回の授業であるため小学生が参加しやすい日程を考えて組み立てる。

8. 資料の配布について決める資料を作成したら配布する範囲、方法など決める。

7.2　運営

運営の流れは以下の通りである。

1. 役割決め

   運営が始まると外部の人と連絡が取ることが多くなるため連絡をして日程調整などをする役割を決める。

2. 必要な道具の決定

   授業で必要な道具 (パソコン、USB メモリ、マグネット、ホワイトボード、教科書など) を決める。

3. 教える体制を決める

   どのような体制でアシスタントを何人にするか決める。

4. 内容の吟味

   授業担当者が考えてきた授業の構成、教える内容、タイムテーブルなどを話し合い決める。

5. サンプルプログラムを決める

   授業のポイントが入ったサンプルプログラムを決める。

6. 模擬授業

   実際の授業を想定して、作成したタイムテーブルをもとに授業をする。スムーズにできるまで繰り返す。

7.3　広報関係提案

広報関係では本学から教育員会に資料を持っていた。その後教育委員会から各小学校へポスターを配布した。ポスターには以下のような内容をまとめる。

- 内容
- プログラミングとは何か
- 日程
- 参加条件
- 時間
- 場所
- 問い合わせ先
- 申し込み方法
- 申し込み締め切り

7.4 ミーティング

週に 1 回メンバーと約 1 時間ほどのミーティングを行う。内容としては以下のような項目である。

- 成果物の発表
- 成果物に対して意見をもらう
- 授業に向けて
- 次回まで準備するもの

7.5 申し込み方法の提案

申し込み方法で情報、確認が確実なのは、申し込みを Web で行い、参加者が決定後に電話でやり取りするのが良い方法だと考える。その理由としては、Web でおこなうことで必要な情報を残すことができる。入力してもらうことは、名前、学校名、住所、メールアドレス、電話番号などである。実際に使用した申し込みフォームは図 7.1 である。また、参加する人に電話をすることで Web 申し込みの内容を直接確認することができ、迅速かつ臨機応変に対応できるからである。

図 7.1: Web 申し込みフォーム

7.6 授業

授業では、分かりやすく丁寧ということを意識する。小学生の反応、操作の進み具合などを確認して教える速度を調整する。アシスタントは、小学生が途中で困らないように声をかけながらサポートする。

7.7　授業後のミーティング

授業後のミーティングは、授業担当者、アシスタント、タイムキーパーなど全員が授業を振り返る時間にする。悪かった部分に対しては、改善策を考えるようにする。

7.8　6 月にプログラミング教室を開催の実例

下の表はメンバー募集から 6 月に開催までの流れをまとめたものである。

表 7.1: 開催までの流れ

日付	企画	運営
2 月 23 日	メンバー募集	
2 月 28 日	ミーティング (企画について)	
3 月 8 日	責任者選出	運営計画
3 月 8 日		Web 作成開始
3 月 26 日	授業内容の決定	必要な道具を決める
4 月 2 日	授業内容の吟味	
4 月 10 日	授業担当者の決定 授業体制の決定 教材作成者の決定	使用する OS の決定 必要な道具発注
4 月 17 日	資料配布小学校決定 授業内容の吟味	ポスター作成開始
4 月 19 日	配布ための連絡調整	
4 月 24 日	配布資料完成 授業の大まかな流れ確認	
5 月 1 日	成果物の確認	連絡状況の確認
5 月 8 日	模擬授業について 教科書作成開始	
5 月 10 日		各小学校へ資料配布
5 月 15 日	マグネットの作成	授業スライドの確認
5 月 15 日		使用する教室の予約
5 月 22 日	各授業の確認	当日の日程について
5 月 28 日	当日に必要なモノの確認	
5 月 29 日	申し込み締め切り	
6 月 5 日		参加者への連絡
6 月 7 日		Web 公開
6 月 8 日		1 回目模擬授業
6 月 10 日		1 回目授業 1 回目授業についてミーティング
6 月 12 日	2 回目の授業に向けて	2 回目模擬授業
6 月 17 日		2 回目授業

表 7.1: 開催までの流れ

日付	企画	運営
		2 回目授業についてミーティング
6 月 19 日	5 回目授業の必要なモノの確認 3 回目の授業に向けて	
6 月 21 日		3 回目模擬授業
6 月 24 日		3 回目授業
		3 回目授業についてミーティング
6 月 26 日	4 回目の授業に向けて	4 回目模擬授業
7 月 1 日		4 回目授業 4 回目授業についてミーティング
7 月 3 日	5 回目の授業に向けて	
7 月 8 日		5 回目授業 5 回目授業についてミーティング

第8章　課題の考察

プログラミング教室を企画運営をし、1.3 で述べた課題について考察をする。

プログラミングツールに関する課題

Ruby てらこったでは Ruby をツールとして使うのを決めた理由は 3 つある。1 つ目の理由は、本学では 2.1.1 で述べたように Ruby を用いたプログラミングを学んでいる。学生が教えるということを前提としたので Ruby とした。2 つ目は、他ではあまり Ruby を教えてないということである。1.2 のように Scrach を教えているプログラミング教室が多いことから他のプログラミング言語にした。3 つ目の理由は、小学生のうちに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ある。

プログラミング環境に関する課題

一人一台の PC を用意した。私立大学ブランディング授業で補助金で PC を揃た。プログラミング教室を行う程度であれば、中古の 2 万円前後の PC で十分である。そのため、中古の PC を参加想定人数分を準備した。

プログラミング教育を行う教員の課題

小学生に指導する大学生は、分かりやすく教えるために 3.3 で述べたように授業準備を行った。準備をするときは、本番の授業をシミュレーション (タイムテーブルなど) して行った。

第9章　付録

使用した補助教材について

9.1　コマンド集

コマンド集(簡単 Ver.)

Emacs(イーマックス)へ移動	kterm(ケータム)へ移動	console(コンソール)へ移動
C-1	C-2	C-3

Emacs 編

C-x C-f	ファイルの作成 / ファイルを開く
C-x C-s	ファイルの保存

kterm 編

cd ディレクトリ	[ディレクトリ] に移動
cd	ホームディレクトリに移動
mkdir ディレクトリ	[ディレクトリ] を作成
ls	今いるディレクトリの中にあるファイルを表示
ruby ファイル名	[ファイル名] を実行

console 編

exit	ログアウト
shutdown -p now	電源をおとす

9.2　アルファベット対応表

ー アルファベット表 ー

A a	B b	C c .chomp	D d
E e elsif , else , end	F f	G g gets	H h
I i if	J j	K k	L l ls
N n \n	M m	O o	P p print , puts
Q q	R r ruby , rand	S s sleep	T t .to_i
U u	V v	W w while	X x
Y y	Z z		

参考文献

[1] 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, (参照日 2019-8-1).

[2] 上松理恵子 久野靖 萩谷昌己 .「小学校にプログラミング教室がやってきた!」株式会社三省堂,2016,p144

[3] 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.

[4] 広瀬雄二.「Ruby プログラミング基礎講座」, 技術評論社,2006,p312

[5] 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/afieldfile/2018/11/06/1403162_02_1.pdf, (参照日 2019-5-24).

[6] 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.