2025年8月9日
全国高等学校情報教育研究会
分科会発表

プログラミングの授業 段階を踏んでやってみた

アサンプション国際高等学校
教諭　岡本弘之
okamoto@assumption.ed.jp
https://okamon.jp

1. はじめに

最初に反省から・・・

情報A・社会と情報では

- 授業であまりプログラミングをやってこなかった・・

情報Ⅰが始まって

- コードを使ったプログラミングは必須
- プログラミングによる問題解決も体験させないと

今まで取り組んできたことと他の実践を組み合わせて
教員も生徒もハードルの低い授業を考えてみた

具体的には

今まで(5h)

アルゴロジック(2h)

改善後(11h)

アルゴロジック(2h)

Scratch(2h)

コードの基本実習(2h)

LEGOマインスドストームによる問題解決(3h)

シミュレーション(2h)

工夫としては

- Sratchから始めてみた
  →プログラミングって楽しいと思ってほしい
- コードで短いプログラムで基本を実習した
  →変数・配列、分岐・繰り返し処理は実習で学ぶ
- プログラミングによる問題解決はモノを動かした
  →画面だけでは難しい、物が動く方が興味を引く
- シミュレーションするためにコードを使った
  →目的のためにコードを使ってみた

2. 授業の流れ

授業の範囲と目標

授業の範囲

- 「コンピュータとプログラミング」
  - 「イ. アルゴリズムとプログラミング」
  - 「モデル化とシミュレーション」

授業の目標

- 体験や実習を通じて学ばせたい
- 写経だけでなく工夫させたい
- 試行錯誤して問題解決させたい
- 共通テストでも役立つ内容に

第3章 コンピュータとプログラミング

日本文教出版「情報Ⅰ」教科書目次

授業の流れ(11時間)

教科書の項目	授業での実習
アルゴリズム	2.1　アルゴロジックで学ぶ(2時間) ・ゲーム形式で学んだあとアルゴリズムについて学ぶ
プログラミング①	2.2　Scratchでプログラミング(2時間) ・Scratchを使って簡単なゲームを作ってみる
プログラミング②	2.3　Pythonの基本を学ぶ(2時間) ・演算・変数・配列・条件分岐・繰り返しなど基本を実習
プログラミング③	2.4　ロボットを動かそう(3時間) ・プログラミングで問題解決をする
モデル化と シミュレーション	2.5　Pythonでシミュレーション(2時間) ・ガチャの確立、釣銭のシミュレーションをやってみる

2.1 アルゴロジックで学ぶ

ここは実習を通じて学ばせたい

1 アルゴリズム

アルゴリズムとは何か，よいアルゴリズムはどうあるべきかについて理解しよう。

考えてみよう

1〜40までの数字が書き込まれてバラバラに並んでいるカードを，番号順に効率よく並べる方法について考えてみよう。

❶アルゴリズム
algorithm
何らかの目的を達成するための処理手順のこと。プログラムを作成する際の論理的な基礎となる。

プログラミング言語 p.137

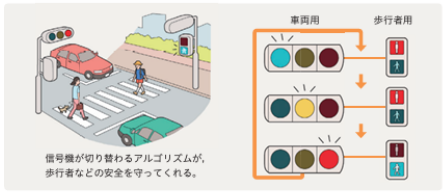
プログラム p.134

1 アルゴリズム

わたしたちが何かを行うとき，手順や方法を工夫して，より効率よく，より見栄えよく終わらせようとすることがあるだろう。コンピュータに何か処理を行わせるには，コンピュータに対して処理手順を指示しておく必要がある。この処理手順のことを**アルゴリズム**という。

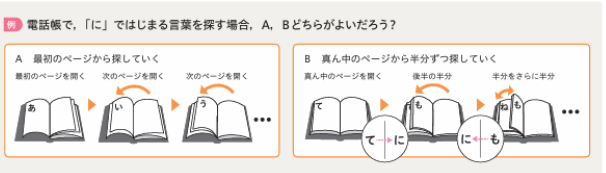
コンピュータに実際に処理手順を与えるときには，アルゴリズムを**プログラミング言語**で表現した**プログラム**を用いる。コンピュータによる処理で意図した結果を得るためには，まずアルゴリズムの入念な検討が重要になる。

▼図1　正確性が求められるアルゴリズム



ある問題を解決しようとするとき，何通りもの手順が考えられる場合がある。このようなときは，複数考えられる手順（アルゴリズム）からより効率的なものを選ぶ必要がある。

▼図2　よりよい手順（アルゴリズム）を選ぶ



2 アルゴリズムの効率性

コンピュータ上で実行されるプログラムは，アルゴリズムしだいで処理効率が大きく変わる場合がある。

たとえば，複数回の計算が必要な問題を解決する場合，すべての計算を個別に記述する方法と，すべての計算に共通する処理をまとめて記述する方法とでは，後者のほうが処理時間が短くなる可能性が高まる。

▼図3　まとめて処理するほうが効率がよい

1本ずつ輪切りにするより，複数本をいっしょに輪切りにするほうが効率がよい。

コンピュータの処理性能が高くても，入力するデータ量や演算の回数が増えれば，処理にかかる時間は増える。むだな処理を省いたアルゴリズムや，効率性を高めたアルゴリズムを用いれば，時間がかかる問題も，わずかな時間で解決できるようになる。

例　素数の判定（101が素数であることを調べる）

❶2〜100のすべての整数で割り切れないか調べる

101 ÷ 2 = 割り切れない
101 ÷ 3 = 割り切れない
101 ÷ 4 = 割り切れない
101 ÷ 5 = 割り切れない
⋮

全部で99個の整数を試すことになる。

❷2と3〜100までのすべての奇数で割り切れないか調べる

101 ÷ 2 = 割り切れない
101 ÷ 3 = 割り切れない
101 ÷ 5 = 割り切れない
101 ÷ 7 = 割り切れない
⋮

全部で50個の整数を試すことになる。

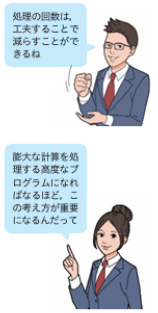
最初に2で割り切れるかを計算するから，2以外の偶数は計算しなくてもよい

❸2と3〜10までのすべての奇数で割り切れないか調べる

101 ÷ 2 = 割り切れない
101 ÷ 3 = 割り切れない
101 ÷ 5 = 割り切れない
101 ÷ 7 = 割り切れない
101 ÷ 9 = 割り切れない

全部で5個の整数を試すだけで済む。

101は，10 × 10 = 100より1大きいだけの数だから，割る数を10までに絞ることができる。



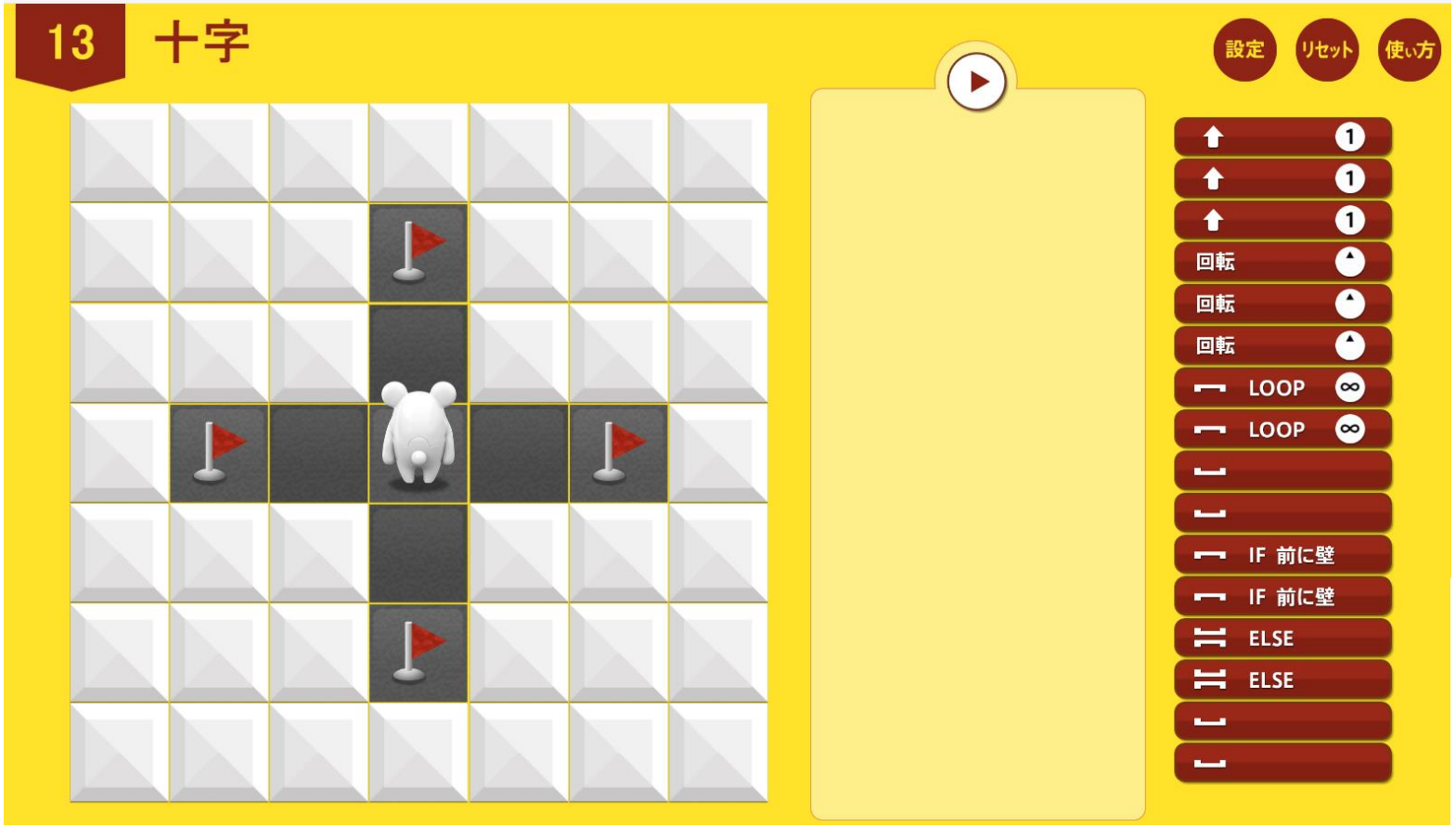
やってみよう

自動販売機で飲み物を購入するアルゴリズムを言葉であらわしてみよう。

日本文教出版「情報Ⅰ」教科書

2.1 アルゴロジックで学ぶ

- アルゴロジック(https://algo.jeita.or.jp/prm/2/index.html)



2.1 アルゴロジックで学ぶ

授業の組み立て
①実習：アルゴロジックをやってみる（25分）
②説明：教科書の内容を説明（下スライド）

①（アルゴリズム）＝何か目的を達成させるための
↓　　処理手順
（プログラム）＝アルゴリズムをコンピュータで処理できるようにプログラム言語で書いた

（例）アルゴリズム体操（NHKピタゴラスイッチ　LINK）
・「アルゴリズム」とは課題を解決する手順や計算方法を表す。アルゴリズム体操では、決まった動きを繰り返し、隣の人とぶつからずに体操できる。（同説明より）

②アルゴリズムの効率性

・同じ目標でも複数の手順がある
→短いアルゴリズムを見つければ効率が良い

2.1 アルゴロジックで学ぶ

②教科書の内容も説明しておく

【知識の整理】

① （　　）＝何か目的を達成させるための処理手順

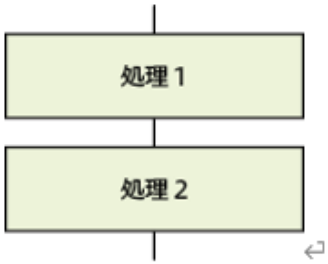
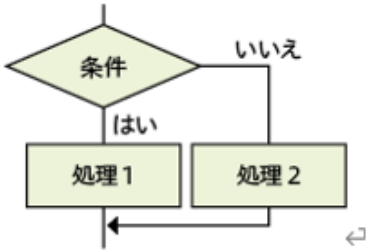
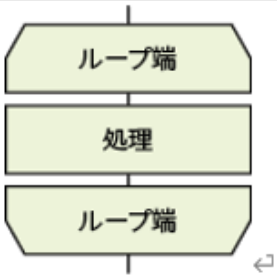
↓　この分解した手順をプログラム言語で書いたもの

（　　）＝コンピュータが処理できるようにプログラム言語で書いたもの

② アルゴリズムの効率性＝何か目的を達成させる手順は複数ある

→効率の良い手順を見つけプログラミングすれば処理速度が速くなる

③ アルゴリズムの基本構造

（　　） ＝順番に処理する	（　　） ＝条件により処理が分かれる	（　　） ＝条件が成り立つ間繰り返す
		

授業プリントより

2.1 アルゴロジックで学ぶ

③課題：身近なことのアリゴリズムを考える

歯磨きをする手順を分解しよう

①歯ブラシと歯磨き粉を用意する
②歯ブラシを左手に持つ
③歯磨き粉を右手に持ちふたを取る
・・・・

☞10手順以上に分解する

何に役立つ？

- 例えば歯磨きをするロボットを作るとき、どのように動かすといいか、手順に分解する→この手順をもとに、コンピュータが理解できるようプログラミング言語でプログラミングする

授業プリントより

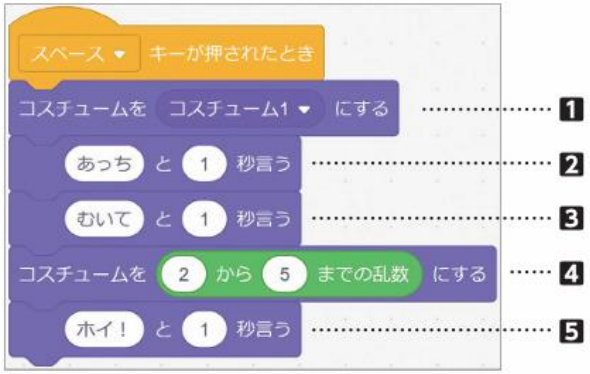
2.1 アルゴロジックで学ぶ

- 生徒の振り返り
  - スマホのアプリは膨大なプログラムで成り立っている
  - コンピュータに手順を教えるには労力がかかる
  - 小さな工夫でも手順に差が出ると思った
  - 自由な動きが必要なゲームは複雑なプログラムだろう
  - 人間のように経験や勘で動けない分、コンピュータには全ての動きを支持しないといけない
  - AIは賢いが、そのプログラムを作る人間はもっと賢い
- 授業者の振り返り
  - アルゴロジックの実習から入るとアルゴリズムの説明が理解させやすい

2. 2 Scratchでプログラミング

①簡単なプログラミングを写経で作ってみる

手順 5 「あっちむいてホイ！」をつくろう。



- **プログラムの意味**

1. スペースが押されたらネコを最初の状態にするために，［コスチューム1］に変える。
2. 「あっち」と1秒言う。
3. 「むいて」と1秒言う。
4. コスチュームを2～5までのどれかに変える。
5. 「ホイ！」と1秒言う。

実行結果 スペースキーを押して実行する。

中野先生
ありがとう

日本文教出版「情報Ⅰ図解と実習」教科書

2. 2 Scratchでプログラミング

②簡単なプログラミングを写経で作ってみる

【プログラムの意味】

①「数1」を1から 20 までのランダムな数とする

②「数2」を1から 20 までのランダムな数とする

③「数1」と「＋」と「数2」を表示し、入力を待つ

④入力された値と「数1」＋「数2」の計算結果が等しいか調べ、一致すれば「正解」不一致なら「不正解」という

参考：日本文教出版「情報Ⅰ 図解と実習」教科書

2. 2 Scratchでプログラミング

③習ったことから改善してみる

Ⅲ．実習２のプログラムをベースに下のような工夫を加えよう。方法も考えよう。
発展例１：正解・不正解の時に音を鳴らす
発展例２：３つの数字を足す計算問題にする

④習ったことをもとに自分でプログラミングしてみる

【発展課題】作ってみよう！

① 実習１のプログラミングを参考に、「じゃんけん」のプログラムを作成してみよう。
手順：①コスチューム２・３・４を作る、それぞれ「グー」「チョキ」「パー」とセリフを加える
※背景・キャラクターもコスチュームから変更可能です。
②コードは基本的には実習１と全く同じもの　※5 行目のみ「2 から 4 までの乱数」
③プログラムのスクリーンショットを Classroom の「プログラミング３」に提出する

Scratchなら生徒はスムーズに取り組めた

2. 2 Scratchでプログラミング

- 生徒の振り返り
  - 思ったより簡単に自分でもゲームが作れてうれしい
  - 乱数を利用しただけで、いろいろなゲームが作れそう
  - プログラミングは難しいと思っていたが思ったより簡単
  - プログラミングができれば、ほしいアプリを作れそう
  - アプリを使うときに、条件分岐・乱数と考えるようになった
  - ゲームのガチャの「乱数調整」の意味が分かった
- 授業者の振り返り
  - Scratchは生徒にとってとりくみたすい「かんたん」
  - 命令の理解しやすく、間違いも発見しやすい
  - 発展課題についても少しのヒントで制作することができた
  - 達成感を感じやすい課題であった。「できた」「うれしい」

2. 3 コードの基本実習

- 環境:Paiza.ioを利用　←ブラウザ上でできる
- 言語:Python　→コードを学ばせたい

2.3 コードの基本実習

- 変数・配列、分岐・繰返し処理など、短いコードを入力するプログラミングを実習を通して学ぶ

（2）変数

TRY2　次の実行結果を予想し、実際に入力してみよう

例　税抜き価格と税率を入力し、税込み価格を計算するプログラム

```
1 # coding: utf-8
2 # Your code here!
3 zeinuki=100
4 zeiritsu=10
5 kakaku=zeinuki+zeinuki*zeiritsu/100
6 print(kakaku)
```

←変数 zeinuki に 100 を代入する
←変数 zeiritsu に 10 を代入する
←変数 kakaku に税込み価格を計算して代入する
←変数 kakau を表示する

プログラムの意味を説明

【知識の整理】

変　数	数値や文字列などデータを 1 つだけ保管する箱（入れ物）のこと
代　入	変数に値を設定すること=を使う　（例）zeinuki=100←変数 zeinuki に 100 を代入

プログラムの意味を説明

（4）繰り返し

TRY　次の実行結果を予想し、実際に入力してみよう

例　0から9まで順番に加算して合計を表示するプログラム

```
1 # coding: utf-8
2 # Your code here!
3 i=1
4 gokei=0
5 for i in range(10):
6     print(i)
7     gokei=gokei+i
8     i=i+1
9 print("合計は",gokei)
```

←変数i（カウント変数）に初期値1を代入する
←変数gokeiに初期値0を代入する
←以下の処理をiが0から9まで10回繰り返す
　←変数iを表示する
　←合計を計算する
　←上の処理が終わればカウント変数に1を足す
←繰り返しが終わればここまでの合計を表示する

（5）配列

TRY　次の実行結果を予想し、実際に入力してみよう

例　飲み物の種類と価格を登録し、対応する飲物の価格を表示するプログラム

```
1 # coding: utf-8
2 # Your code here!
3 Nomimono=["水","お茶","オレンジ","アップル"]
4 Kakaku=[100,120,140,160]
5 print(Nomimono[1],"の価格は",Kakaku[1],"円です")
```

←配列Nomimonoの要素を定義
←配列Kakakuの要素を定義
←配列Nomimono・Kakakuの
　要素の添え字1番を表示

応用：水の価格が表示されるようにプログラムを変更しよう

【知識の整理】

配　列	複数の値を一つの名前で管理するデータのこと 配列の中の1つ1つの値を要素と言い、要素の順番を添え字で表す ※添え字は0から　Kakaku=["100"　,　"120"　,　"140"　,　"160"　] ↑　↑　↑　↑ Kakaku[0]　Kakaku[1]　Kakaku[2]　Kakaku[3]

適宜スクリーンショットの提出を求めて評価の対象に

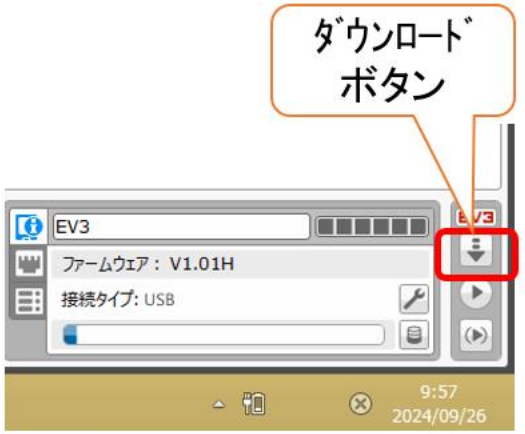
2.3 コードの基本実習

- 生徒の振り返り
  - プログラミングはすごく複雑で難しい
  - 少しでも間違えるとうまくいかず、頭が混乱した
  - エラーが出るともやもやした
  - プログラミングコードの意味がわかった
  - ゲームでランダムに武器が出てくる仕組みが分かった
  - Scratchよりもっと複雑なプログラムができる
  - プログラミングは難しいが、案外行けた
- 授業者の振り返り
  - あちらこちらでエラー→一人での指導に限界を感じた
  - 短いプログラムの実習にしておいてよかった・・

2.4 ロボットを動かそう

- 目標:プログラミングで問題解決する
- 方法:グループでLEGOマインドストームのプログラミングを考え、課題を達成する

①ケーブルで接続する
②電源を押す



③プログラムを
ダウンロードする

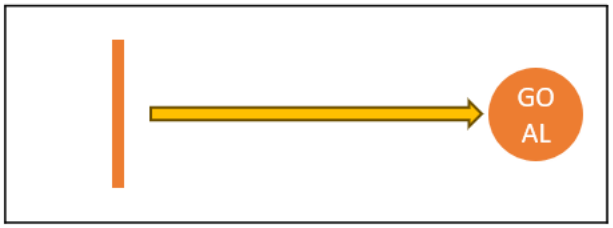
2.4 ロボットを動かそう

- 最初は単純なコースから

ターゲットでぴったり停めよう

コース1

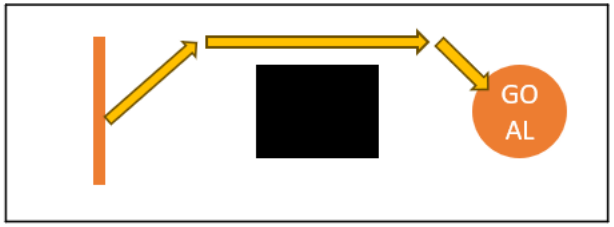
- まっすぐすすんでぴったり停める



- 個人で1回ずつ試す
- 試行錯誤の記録をワークシートNo.17に記録

コース2

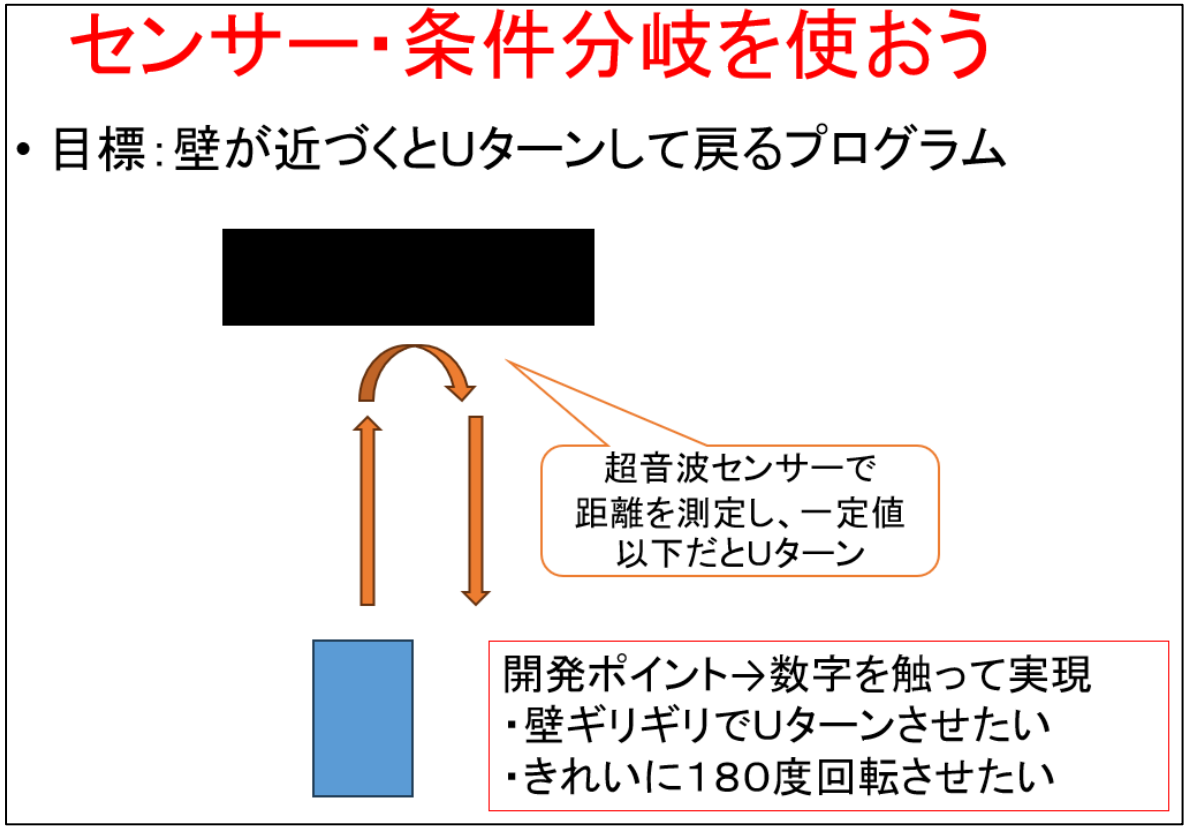
- 障害物をよけてぴったり停める



- 個人で1回ずつ試す
- 試行錯誤の記録をワークシートNo.17-2に記録

2.4 ロボットを動かそう

- 慣れたらセンサーも使ったプログラムも考える



2.4 ロボットを動かそう

- 試行錯誤の過程を記録して改善していく

◇ワークシート①

①自分で考えたプログラミングの内容

プログラミングの内容（設定した数値など）	他の実行結果を見ての修正・やってみた結果
置いたパーツ設定した数値 （例）タンク　50，50，50	（例）手前で止まったので もう少し数値を増やす

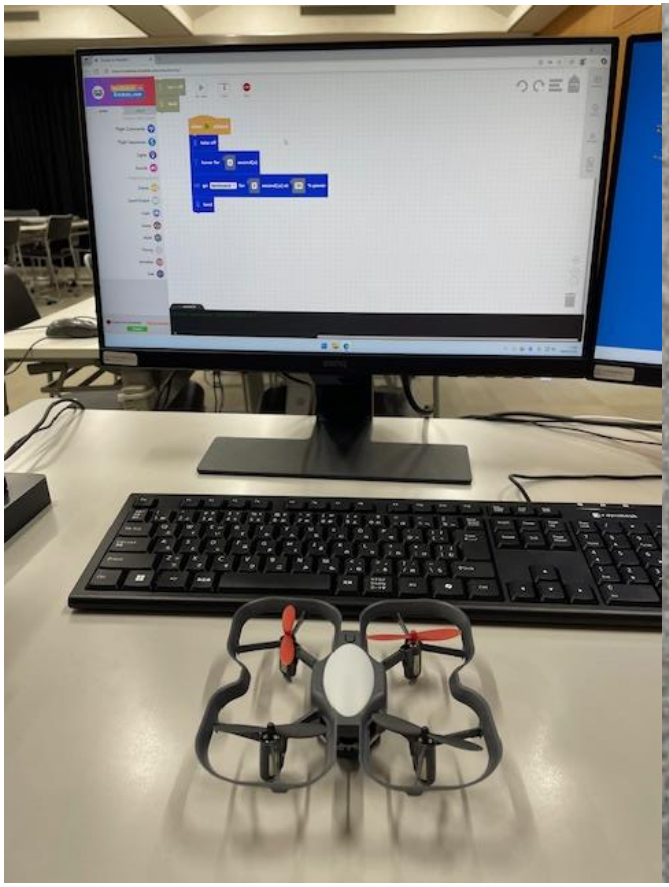
②グループでの改善の経過の記録

	プログラミングの内容	実行結果	振り返り・改善の方向性
1回	タンク 50，50，50	手前で止まる	数値を増やす
2回	タンク 50，50，60	手前で止まる 曲がった	数値を増やす 置き場所を変える
3回			
4回			

グループの他の人の結果も記録して改善案を考える

2.4 ロボットを動かそう

- 今年度はドローンでの実践を予定



2.4 ロボットを動かそう

- 生徒の振り返り
  - 課題を達成できたらうれしい
  - 課題達成のためには他のグループの観察も重要だ
  - 失敗から試行錯誤を繰り返し、成功には時間がかかる
  - 少し数値を変えるだけで動きが変わり成功につながる
  - プログラミングはトライ＆エラーの繰り返しが重要だ
  - 物を動かすには、いろいろな要素が増え難しくなる
  - 今回のようなセンサーはいろいろなところで使われている
- 授業者の振り返り
  - プログラミングによる問題解決まで実習できた
  - 物を動かすのはわかりやすく、おもしろい

2.5 Pythonでシミュレーション

- 最後はPythonでシミュレーション①

目標

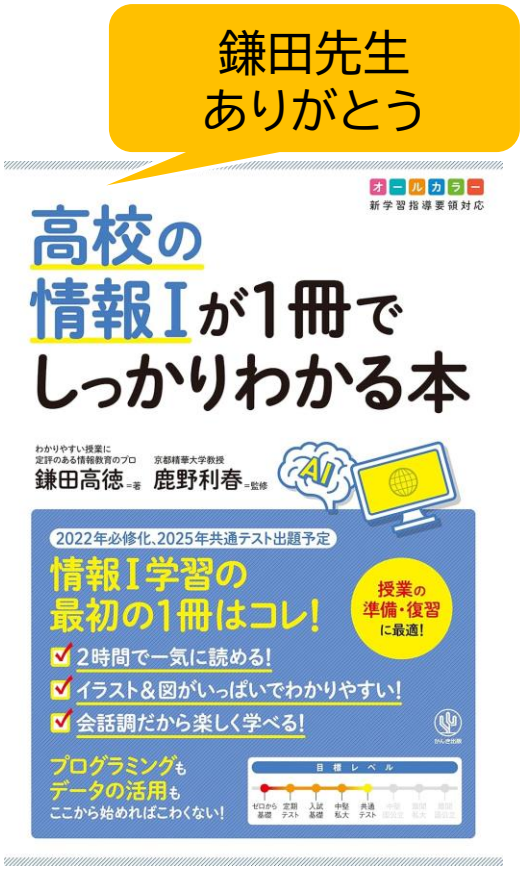
- 出現確率（当選確率）1%のガチャを100回引いて、レアキャラをゲットできる確率は何%か？

方法

- シミュレーションするプログラムを作って調べる

アルゴリズム

- 乱数1～100を発生させ、1が出たときだけ「あたり」と表示（確率1/100）するプログラムを作る
- これを100回繰り返し当たりの回数を数える



2.5 Pythonでシミュレーション

- 手順を追って確認しながら進めていく

STEP1
確率1%のガチャを作る

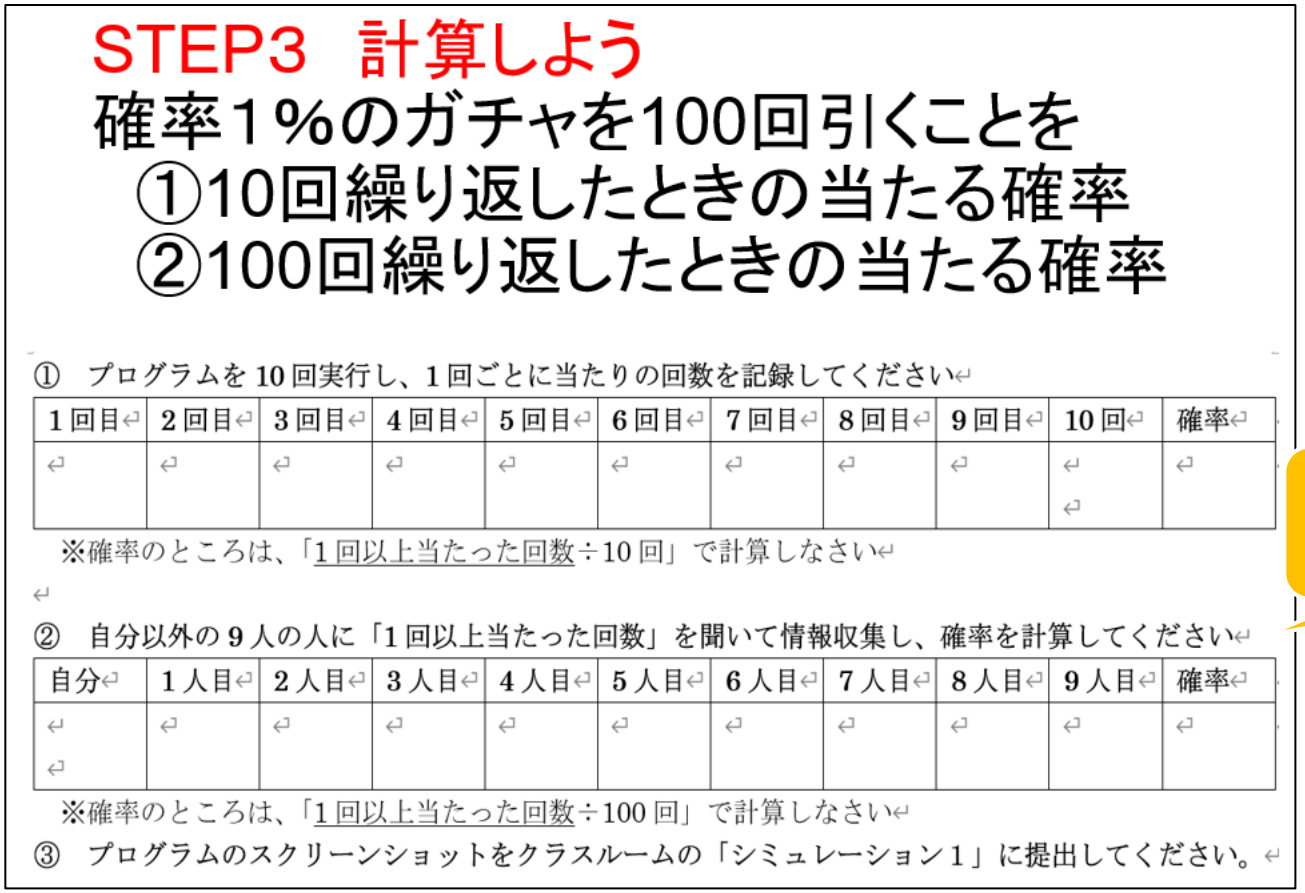
```
1 # coding: utf-8
2 # Your code here!
3 import random              ←乱数のモジュールを組み込む
4 a=random.randint(1,100)   ←変数aに1～100の乱数を代入す
5 print(a)                  ←変数aを表示する
6 if a==1:                  ←もし変数aの値が1と一致した
7     print("あたり")       ←"あたり"を表示する
8 else:                     ←そうでなければ
9     print("はずれ")       ←"はずれ"を表示する
```

STEP2
100回繰り返しあたりの数を数える

```
1  # coding: utf-8
2  # Your code here!
3  import random
4  b=0                      ←当たりの回数を数える変数bを初期化する
5  for i in range(100):     ←カウント変数iが100になるまで繰り返す
6      a=random.randint(1,100)
7      print(a)
8      if a==1:
9          print("あたり")
10         b=b+1            ←あたりが出たら変数bに1を加える
11     else:
12         print("はずれ")
13 print("当たりの回数は",b)  ←変数b＝あたりの回数を表示する
```

2.5 Pythonでシミュレーション

- アナログなこともあえてやってみる



STEP3　計算しよう

確率1%のガチャを100回引くことを
①10回繰り返したときの当たる確率
②100回繰り返したときの当たる確率

① プログラムを 10 回実行し、1 回ごとに当たりの回数を記録してください

1回目	2回目	3回目	4回目	5回目	6回目	7回目	8回目	9回目	10回	確率

※確率のところは、「1回以上当たった回数÷10回」で計算しなさい

② 自分以外の 9 人の人に「1 回以上当たった回数」を聞いて情報収集し、確率を計算してください

自分	1人目	2人目	3人目	4人目	5人目	6人目	7人目	8人目	9人目	確率

※確率のところは、「1回以上当たった回数÷100回」で計算しなさい

③ プログラムのスクリーンショットをクラスルームの「シミュレーション1」に提出してください。

アナログな
とりくみ

2.5 Pythonでシミュレーション

- 教科書の知識も教えておく

【知識の整理】

①モデル化とシミュレーション

・（　　）＝本質的な部分を強調し、それ以外の要素や条件を省略し単純化したモデルを作る

↓　モデルを使えば様々な実験や予想が可能になる！

・（　　）＝モデルを使って現象を予測したり、モデルに変更を加え試してみること

→現物では危険を伴ったり、費用が掛かったり、実験が困難なものに適している

②モデルの分類

1）表現の仕方による分類

（　　）＝物理的な模型	（実物モデル）	実物と同じ大きさのモデル（例）衝突実験の人形
	（拡大モデル）	実物を拡大したモデル（例）分子模型
	（縮小モデル）	実物を縮小したモデル（例）地球儀
（　　）＝式や図で表現	（図的モデル）	対象を図で表現したもの（例）ベン図、家具の配置図
	（数式モデル）	対象を数式で表現したもの（例）速さ＝距離÷時間

2）表現するものの特性による分類

時間の概念の有無	（動的モデル）	時間の経過によって変化するモデル（例）落下する物体
	（静的モデル）	時間の経過を考える必要のないモデル（例）図形の面積の求め方

不確定要素の有無	（確率モデル）	不確定要素を含むモデル（例）さいころの目、台風の進路
	（確定モデル）	不確定要素のない規則的な現象のモデル（例）水槽に水を貯める

データの連続性	（連続モデル）	データの連続的な状態を表現するモデル（例）気温の変化
	（離散モデル）	データが散らばった状態を表現するモデル（例）来客の予想

授業プリントより

2.5 Pythonでシミュレーション

- 最後に、違うシミュレーションもやってみる

【発展課題】 調べよう・考えよう

目的

- 文化祭で200円の商品を60人に販売した場合に必要な100円玉の数をシミュレーションする
- 条件：支払いは300円ちょうどの人と500円玉で支払う人のみとし割合は同じとする

手順

- 1人から60人めまで、300円と500円を支払う人をランダムに出現させ100円玉の枚数を見る
- 乱数（0～1の数をランダムに発生）を使い、0.5未満の場合を300円ちょうど支払う人、0.5以上の場合は500円支払う人とし出現確率を50%ずつとする。

```
3 import random                                  ←乱数のモジュールを組み込む
4 coin=0                                        ←変数 coin の初期値を 0 にする
5 min_coin=0                                    ←変数 min_coin の初期値を 0 にする
6 for i in range(1,61):                         ←カウント変数を 1 から 61 まで繰り返す
7     rn=random.random()                          ←変数 rn に乱数（0~1 の間で発生）を代入
8     if rn<0.5:                                  ←変数 rn が 0.5 未満ならば
9         pay="300円"                               ←300 円を支払う人とする
10        coin=coin+3                               ←変数 coin に 3 枚加えて代入する
11    else:                                       ←変数 rn が 0.5 以上ならば
12        pay="500円"                               ←500 円支払う人とする
13        coin=coin-2                               ←変数 coin から 2 枚引いて代入する
14    print(i,"人目",pay,"　１００円玉は",coin,"枚")  ←一人一人の結果を表示する
15    if coin<min_coin:                           ←変数 coin が変数 min_coin より小さければ
16        min_coin=coin                             ←変数 min_coin に変数 coin の枚数を代入
17 print("最大不足枚数は",min_coin,"枚です")        ←最後に変数 min_coin を表示する
```

2.5 Pythonでシミュレーション

- 生徒の振り返りより
  - 紙やペンで計算するよりも早くにシミュレーションできる
  - プログラムを使えば、条件を変えることも簡単にできる
  - ゲーム好きなのでガチャの確立を求めるのは楽しかった
  - コツをつかんだらコードも早く打てるようになった
  - 乱数を使ったシミュレーションは人によって結果が違った
  - 活用すれば危険を回避し出費を抑制できる
  - シミュレーションでは防災で役に立つ
- 授業者の振り返り
  - 2回目なのでコード入力がスムーズ→教師も生徒も慣れた
  - ～するためにという目的をもってプログラミングできた

2. 共通　評価をどうするか？

- アルゴリズム＝記録＋課題＋振返り
- プログラミング①＝課題の提出＋振返り
- プログラミング②＝課題の提出＋振返り
- プログラミング③＝試行錯誤の記録＋振返り
- シミュレーション＝課題の提出＋考察＋振返り

ワークシート＝記録や考察・課題、振返りの記入

課題の提出＝取り組み内容を確認

3. まとめ

生徒の振り返りのキーワード

2.2 Scratchでプログラミング

- 楽しい、思ったより簡単、自分でもできた
- がっぱれば複雑なプログラミングもできそう

2.3 Pythonでプログラミング

- 難しい、エラーが出るともやもやする
- うまくいったら楽しい、意味が分かった

2.4 ロボットでプログラミング

- 難しいが楽しい、
- 試行錯誤、動かしてみて修正

2.5 Pythonでシミュレーション

- 条件を簡単に変えられる、素早くできる
- 重ねるごとにプログラミングが楽しくなった

工夫の評価

- Sratchから始めてみた
  →プログラミングって楽しいと思ってくれた
- コードで短いプログラムで基本を実習した
  →最初はエラー続出。短いコードにしてよかった
- プログラミングによる問題解決はモノを動かした
  →楽しく試行錯誤しながら問題解決できた
- シミュレーションするためにコードを使った
  →プログラミングの復習にもなった

まとめ

- Scratchから始めていいんじゃない？
  - いきなりコードより生徒は取り組みやすい
- コードは最初は短い写経でもいいんじゃない？
  - 短い方が修正しやすい、生徒は写経でも達成感
- モノを動かすプログラミングはおもしろい！
  - 情報Ⅰだから問題解決はやらせたい
  - 画面で完結するより動く方がおもしろい
- 仕組みを知ったら見えないものが想像できる
  - 身近なアプリ・製品のプログラムなど気になる

最後に授業を共有します！

- 今日のプリントもWebに掲載しています
「情報科の授業アイデア」 https://www.okamon.jp

情報科の授業アイデア

since2005.2.22　update2023.8.1
アサンプション国際中学校高等学校
情報科社会科探究科教諭　岡本弘之

情報科の教員岡本弘之のホームページです。生徒にとって「おもしろく(興味深く)」かつ「役に立つ」情報科の授業をめざし、日々教材研究に取り組んでいます。
このWebは情報科教員同士の情報共有を目的に、勤務校での情報科の授業実践について、授業内容・プリント・スライドを公開しています。(授業利用は許諾不要です)

Topics
・情報Ⅰの実習プリントを追加しました(2023.5.8)
・2・3学期の授業プリントを追加しました(2023.3.7)
・TOPページを情報Ⅰ向けの内容に更新しました(2022.7.18)

- プロフィール・今までの発表履歴
- 情報科の授業アイデアブログ
- 授業で使えるＬＩＮＫ集
- 探究(総合学習)の授業アイデア
- MAIL johoｱｯﾄassumption.ed.jp

高校1年　情報Ⅰ(2単位)　授業プリント集

	1．情報社会の問題解決	
①情報とメディアの特性	④知的財産権を知ろう	⑦マイブームをプレゼンしよう
ﾌﾟﾘﾝﾄ(Word PDF)	ﾌﾟﾘﾝﾄ(Word PDF)ｽﾗｲﾄﾞ(PowerPoint)	ﾌﾟﾘﾝﾄ(Word PDF)ｽﾗｲﾄﾞ(PowerPoint)
②文書作成アプリに慣れよう	⑤情報セキュリティ	⑧ネット社会の課題(情報モラル)
ﾌﾟﾘﾝﾄ(Word PDF)	ﾌﾟﾘﾝﾄ(Word PDF)ｽﾗｲﾄﾞ(PowerPoint)	ﾌﾟﾘﾝﾄ(Word PDF)
③問題解決(紙飛行機を作ろう)	⑥情報技術の発展と社会の変化	
ﾌﾟﾘﾝﾄ(Word PDF)ｽﾗｲﾄﾞ(PowerPoint)	ﾌﾟﾘﾝﾄ(Word PDF)ｽﾗｲﾄﾞ(PowerPoint)	

情報科の先生方つながりましょう！
Facebook友達申請歓迎します