プログラミングに関する授業実践報告

埼玉県高等学校情報教育研究会
宮﨑　万希子

会誌第13号及び教材→埼玉県高等学校情報教育研究会Web
http://www2.spec.ed.jp/krk/jyoho/comm2/?page_id=245

埼玉県高等学校情報教育研究会
研究委員会について

* 教科「情報」で採用されたメンバーで構成
* 毎年共通のテーマを決めて指導案を研究
* 一昨年は「プログラミング」教えるにあたってのアンケートと授業案の検討
* 昨年はアンケートの継続と授業実践

今回の発表では授業案と実践報告を行います

前半の発表　アンケート結果より

- ほぼ半数がプログラミングの経験がない
- プログラミングは難しいという先入観
- 日常との結びつきが薄い

＊**プログラミングは難しい**という先入観を払しょくし、

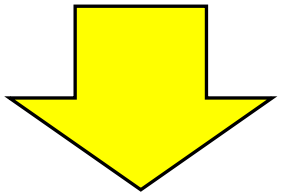
プログラミングへ**興味を持たせる**授業をしたい！

code.orgを利用した授業案の作成

前半の発表　アンケート結果より

- ほぼ半数がプログラミングの経験がない
- プログラミングは難しいという先入観
- 日常との結びつきが薄い

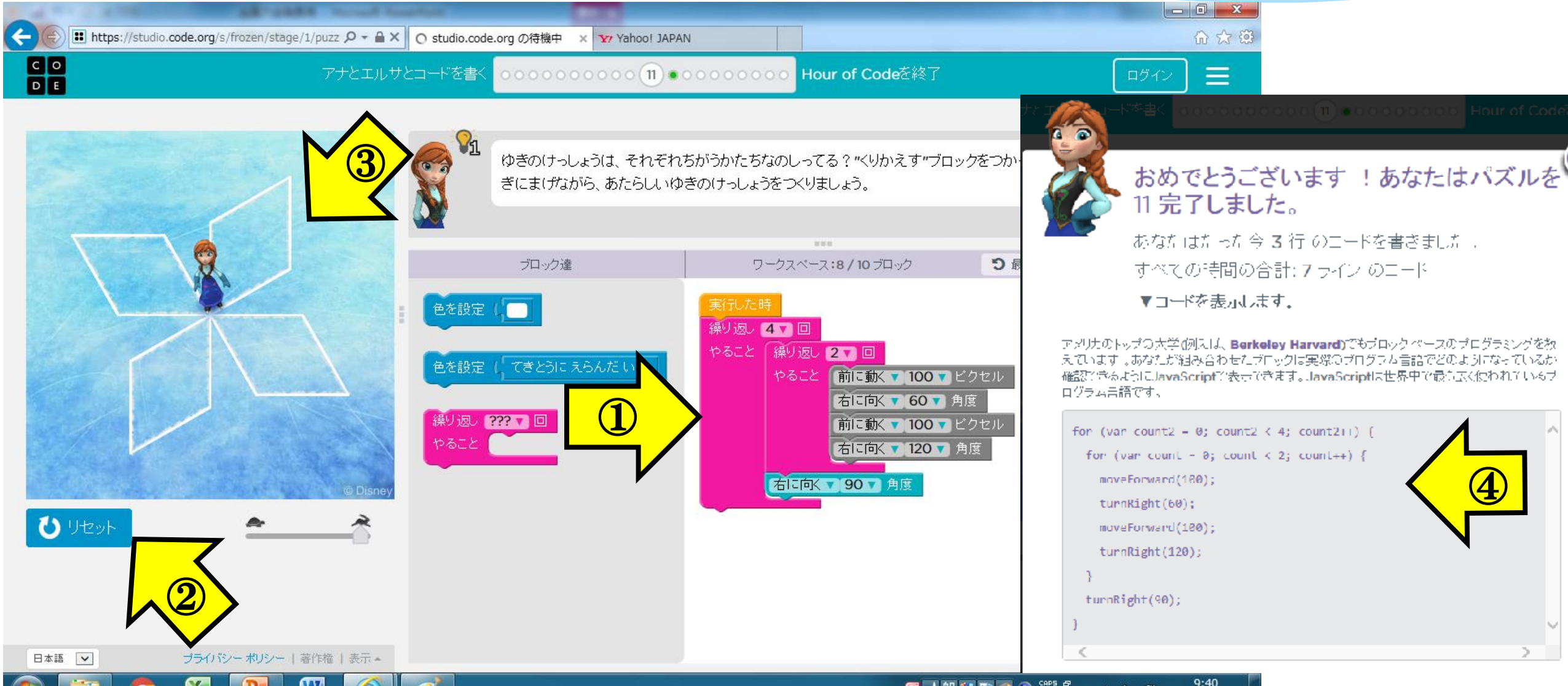
身の回りにはプログラムで動いているものが多くあるのに、
それを認識していない生徒が多い。



「身近なところでプログラムが動いている」ことを
生徒に実感してもらう授業（追加目標）

code.org

- ブロックでアルゴリズムを確認
- コード表示で実際に動いているプログラム（JavaScript）を確認



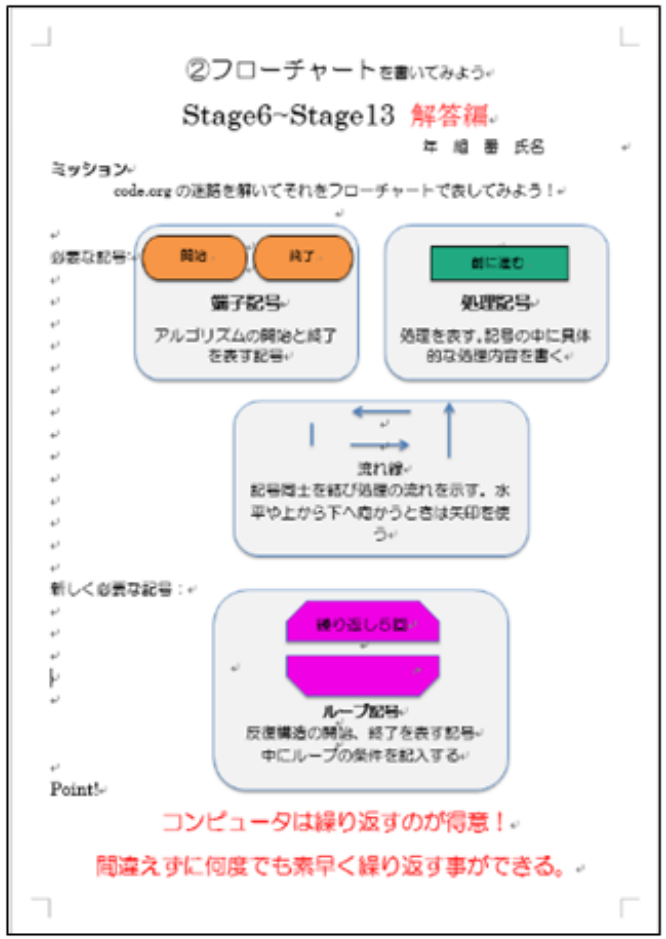
code.studioを用いた　授業案と実践報告

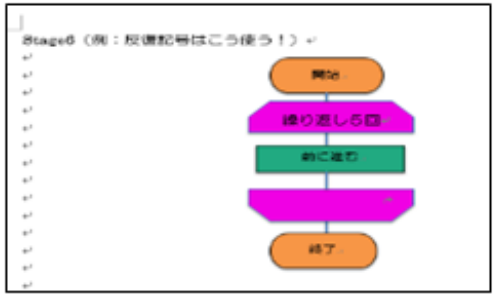
1. ブロック　→　フローチャート　→　プログラム確認
2. フローチャート　→　ブロック　→　プログラム確認
3. ブロック　→　フローチャート　→　プログラム確認
4. ブロック　→　フローチャート　→　問題解決
5. ブロック　→　プログラム確認　→　プログラミング
6. ブロック　→　プログラム確認　→　コーディング

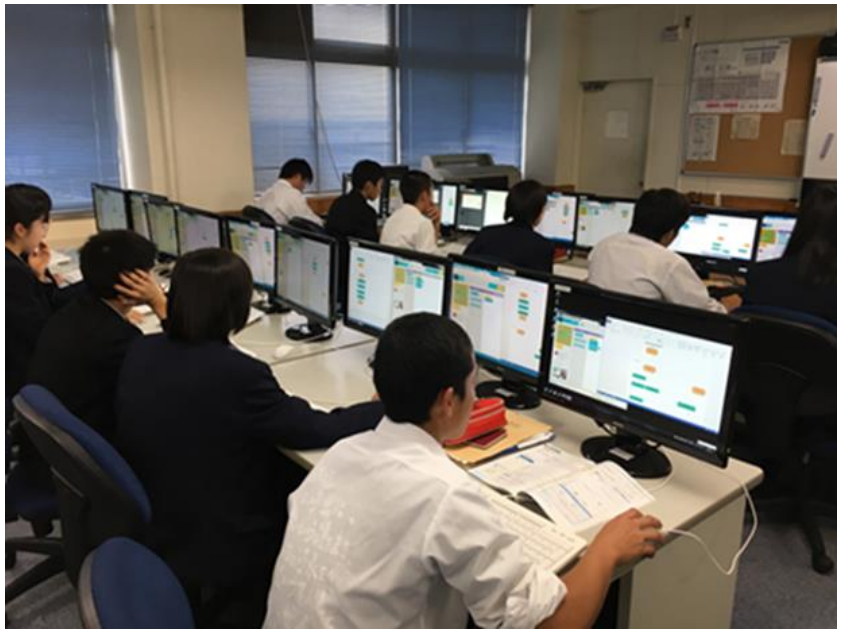
code.studioを用いた　授業案と実践報告

1. ドリル形式のワープロソフト教材でフローチャートを作成

1. ブロック→フローチャート→プログラム確認







実施内容と振り返り

プログラムの構造の理解＋フローチャートの書き方の習得＋ワープロソフトの技能習得

楽しみながらフローチャートの書き方、論理的な考え方や構造の理解ができた。
ゲーム感覚になりすぎずに段階的に学習ができた。

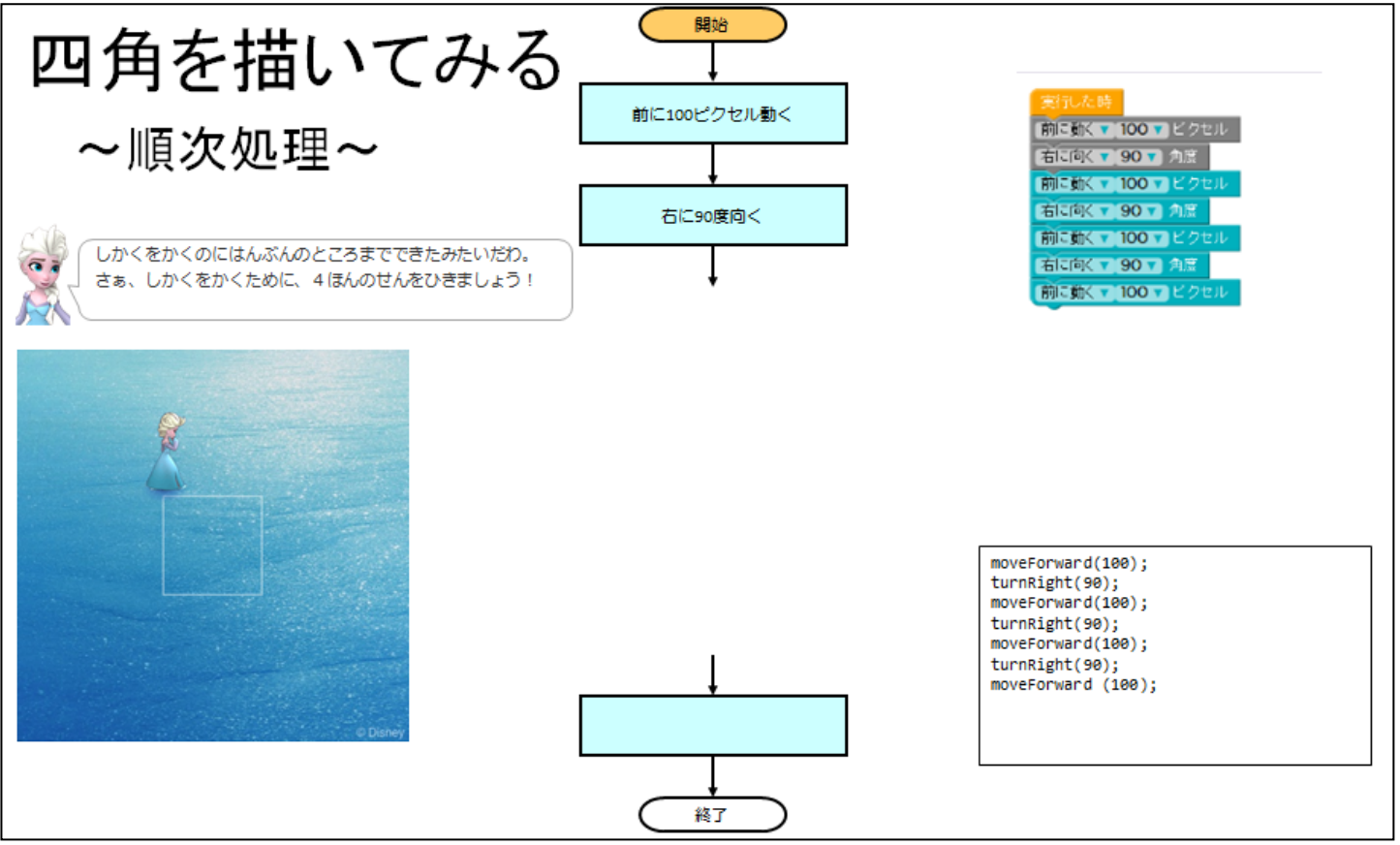
「繰り返すブロックが欲しい」等の解答が得られた。
生徒自ら、ワープロソフト教材で作業することで、反復構造の必要性や構造を理解することができたと考察できる。

フローチャート問題　正答率14％→48％へアップ

code.studioを用いた　授業案と実践報告

2. フローチャートとビジュアルプログラミング

2. フローチャート→ブロック→プログラム確認



実施内容と振り返り

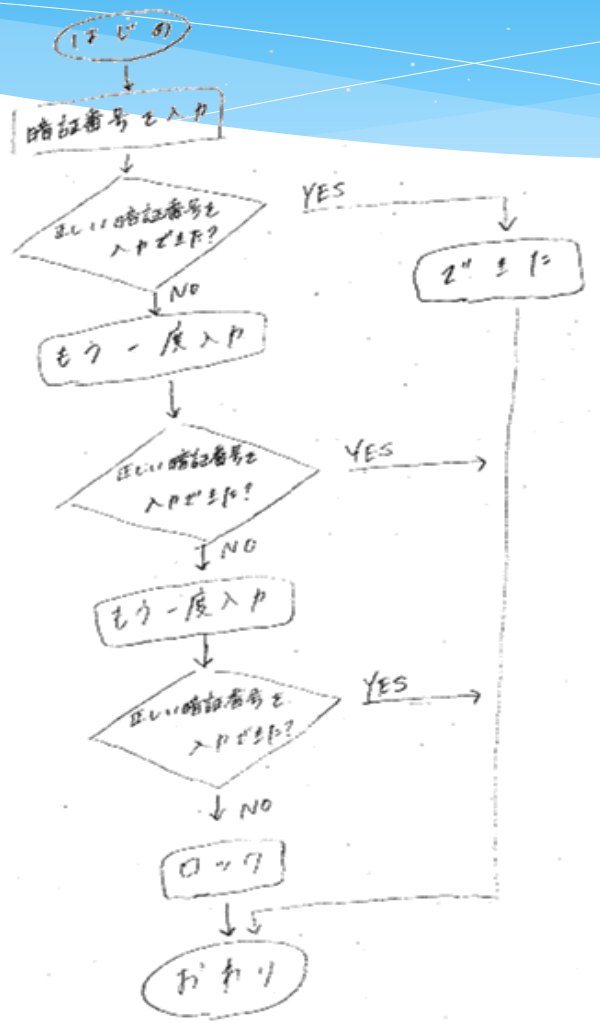
課題A：暗証番号の認証
課題B：店頭での電子マネーの支払い
課題C：自動改札機の入場

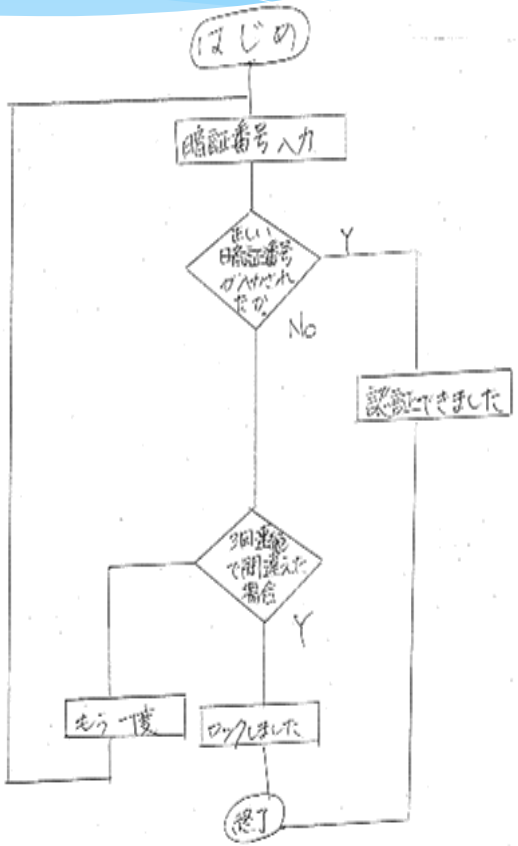
ICカード乗車券やATMなど
身近な情報システムの仕組みを
フローチャートにすることができた

フローチャートを作成を通じて、あらゆるコンピュータやシステムがプログラムによって動いていることを理解させた。

フローチャートで問題整理ができた
自分達で考えアレンジするグループもあった

グループで課題解決にはフォローが必要
課題Aは変数の概念を理解させる必要あり



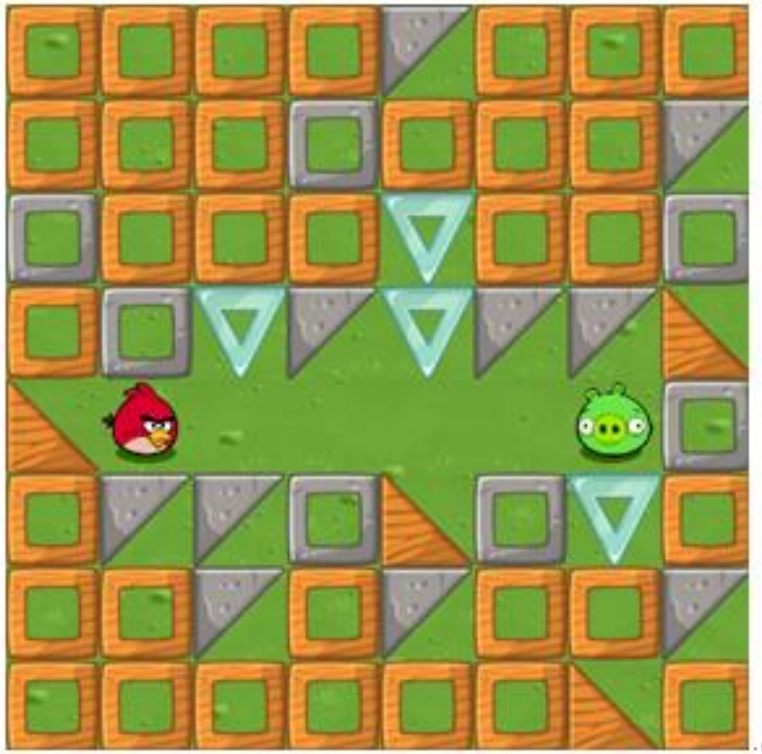


code.studioを用いた　授業案と実践報告

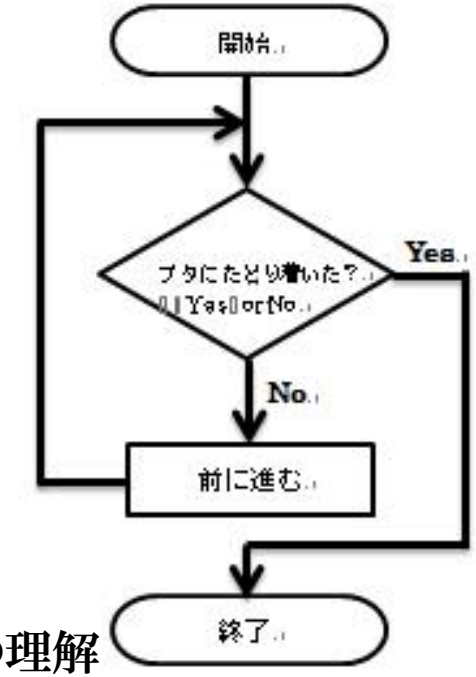
3. ブラックボックスからホワイトボックスへ～挙動を知る

3. ブロック→フローチャート→プログラム確認

（3）反復（繰り返し、ループ）□その2
例題1□



```
while (notFinished()) {
  moveForward();
}
```

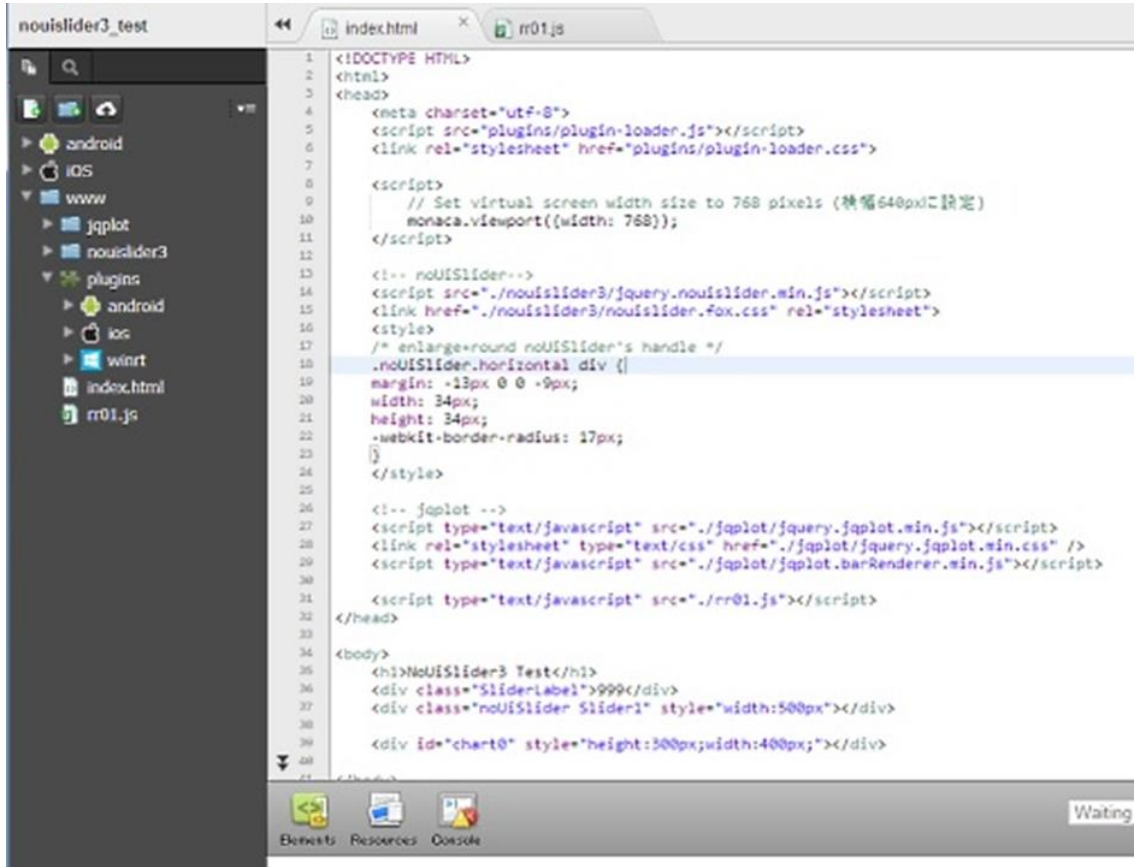


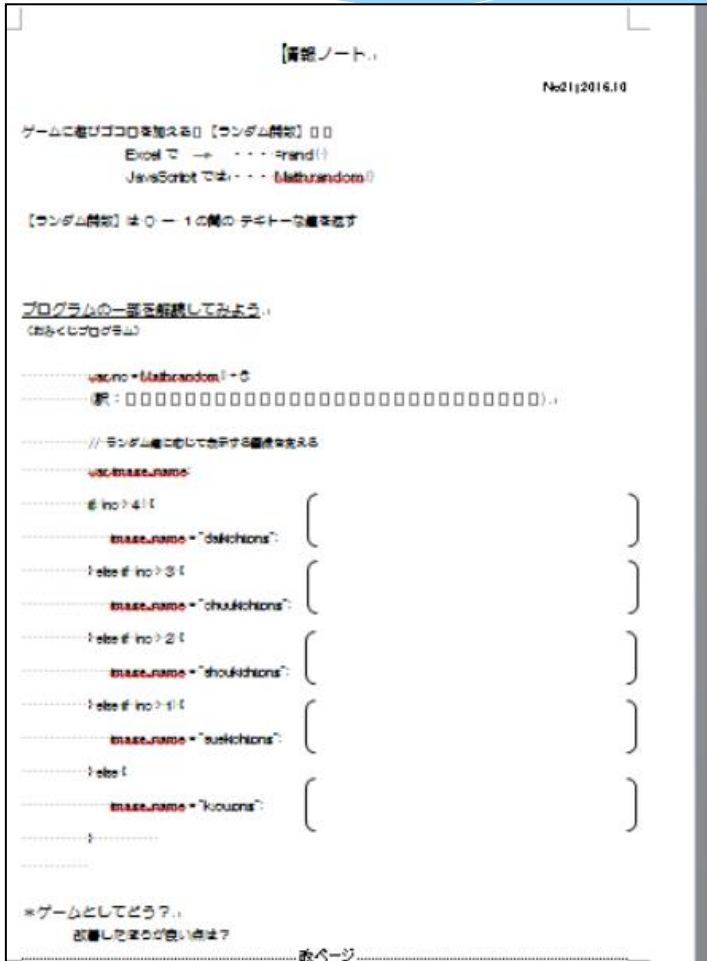
Code.org＋フローチャート＋動画教材で構造の理解

実施内容と振り返り

※Webを利用する教材は、
つながらない場合に備えて
別教材の準備が必要

フローチャートの解読とランダム関数の理解
表計算ソフト、総合開発環境monacaを使用し、ガチャゲームの一部改造
＋追加機能について期末で出題、出てきた回答をもとに意見交換をさせた




【実習ノート】

No21j2016.10

ゲームに遊び心を加える【ランダム関数】

Excel で　→　・・・rand()

JavaScript では・・・Math.random()

【ランダム関数】は０～１の間の デキトーな値を返す

プログラムの一部を解読してみよう

（おおくじプログラム）

var no = Math.random() * 5;

駅：

// ランダム値に応じて当たる賞品を決める

var mase_name;

if(no > 4){

mase_name = "daikichions";

} else if(no > 3){

mase_name = "chuukichions";

} else if(no > 2){

mase_name = "shoukichions";

} else if(no > 1){

mase_name = "suekichions";

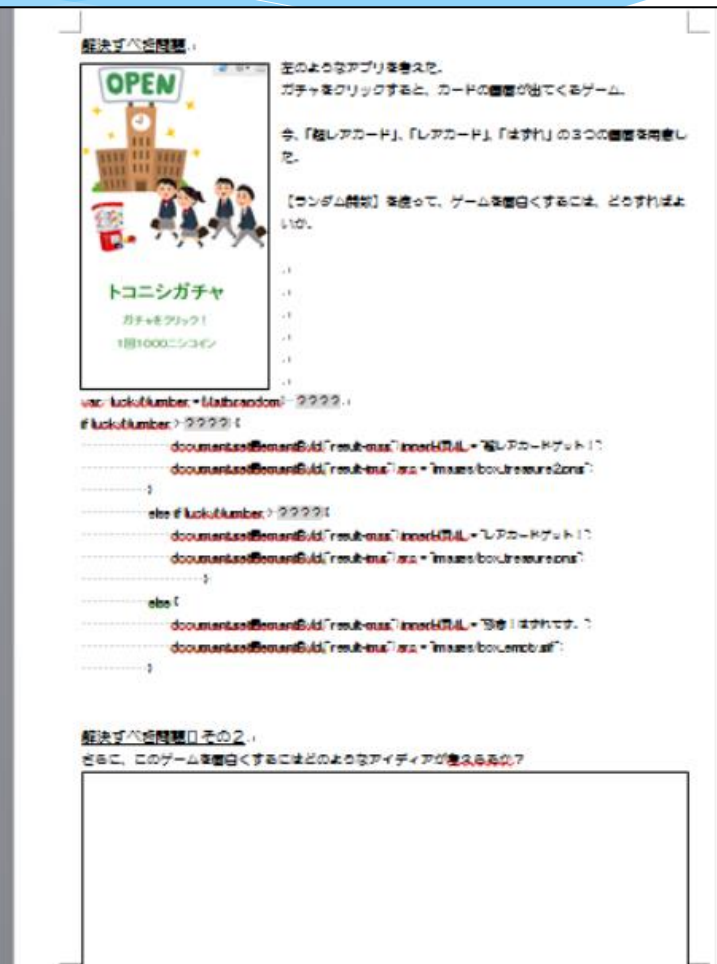
} else {

mase_name = "kyouns";

}

※ゲームとしてどう？

改善した方が良い点は？


解決すべき問題

左のようなアプリを考えた。
ガチャをクリックすると、カードの画像が出てくるゲーム。

今、「超レアカード」、「レアカード」、「はずれ」の３つの画像を用意した。

【ランダム関数】を使って、ゲームを面白くするには、どうすればよいか。

var luckyNumber = Math.random() * 5;

if(luckyNumber > 4){

document.getElementById("result-msg").innerHTML = "超レアカードゲット！";

document.getElementById("result-img").src = "images/box_treasure2.png";

}

else if(luckyNumber > 3){

document.getElementById("result-msg").innerHTML = "レアカードゲット！";

document.getElementById("result-img").src = "images/box_treasure.png";

}

else {

document.getElementById("result-msg").innerHTML = "残念！はずれです。";

document.getElementById("result-img").src = "images/box_empty.gif";

}

解決すべき問題 その２

どうに、このゲームを面白くするにはどのようなアイディアが考えられるか？

code.studioを用いた　授業案と実践報告

4. アルゴリズム学習から問題解決実習へ

4. ブロック→フローチャート→問題解決

1限～2限　次のコードは何をあらわしているか

```
for (var count2 = 0; count2 < 4; count2++) {
  for (var count = 0; count < 2; count++) {
    moveForward(100);
    turnRight(60);
    moveForward(100);
    turnRight(120);
  }
  turnRight(90);
}
```

フローチャートを書いてみよう

例）遠足のカレー

作業内容（順番を考えよう）

- 野菜を切る
- 肉を切る
- 材料を炒める
- 鍋に水を入れる
- ルーを入れて煮込む
- 米を炊く
- 火をおこす
- 使った皿や鍋を洗う
- 調理器具を用意する
- 使った調理器具を洗う
- 全員で「いただきます」
- ごみを捨て、流しを掃除する
- テーブルを拭く
- お皿に米とカレーをよそう
- 皿をテーブルまで運ぶ
- 味見する
- 米を研ぐ
- 煮込む
- あくをとる

確認事項

- 材料に火は通ったか
- 全員分のカレーはそろったか
- 片づけは終わ

実施内容と振り返り

機械に伝えるコーディングではなく、
人に伝えるフローチャートを作り、
問題解決へ利用する。

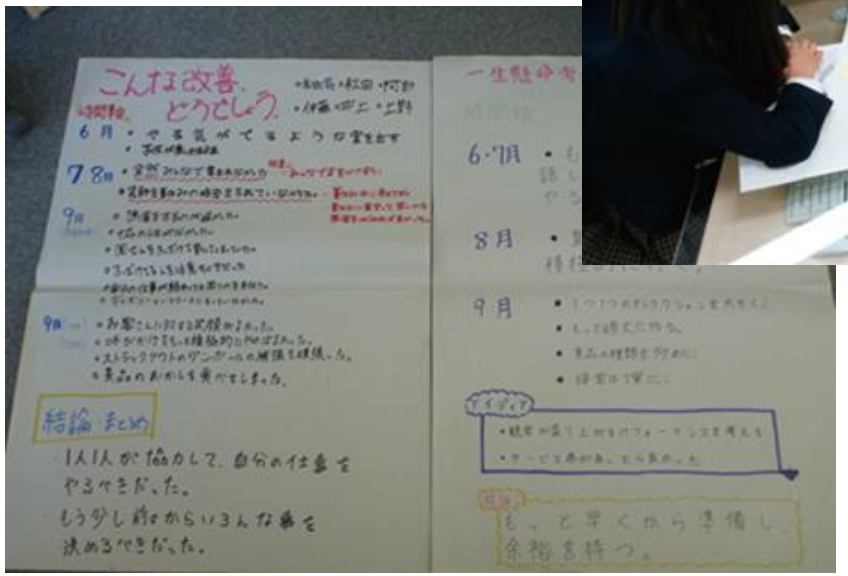
作業をリスト化し、
流れをフローチャートにすることで、
パターン化できる部分を共有できる。
フローチャートをもとに話し合い、
今までできなかった部分に意識を向け、
新しい工夫を生み出すことができた。

課題
発表用資料の作成に時間がかかる
フローチャート図が残らなかった

課題：文化祭のクラス企画を振り返る

- 文化祭準備～当日までの期間に、何をしたか？
- いつ、どんな作業を、誰が担当していたか？
- 自分がした仕事、係りで担当した仕事、クラス全体でした仕事、誰もやらなかった仕事・・・
- 無駄な部分はどこか、失敗した点はあるか、それに対しての改善案はあるか・・・

できるだけ具体的に書き出してみよう!!

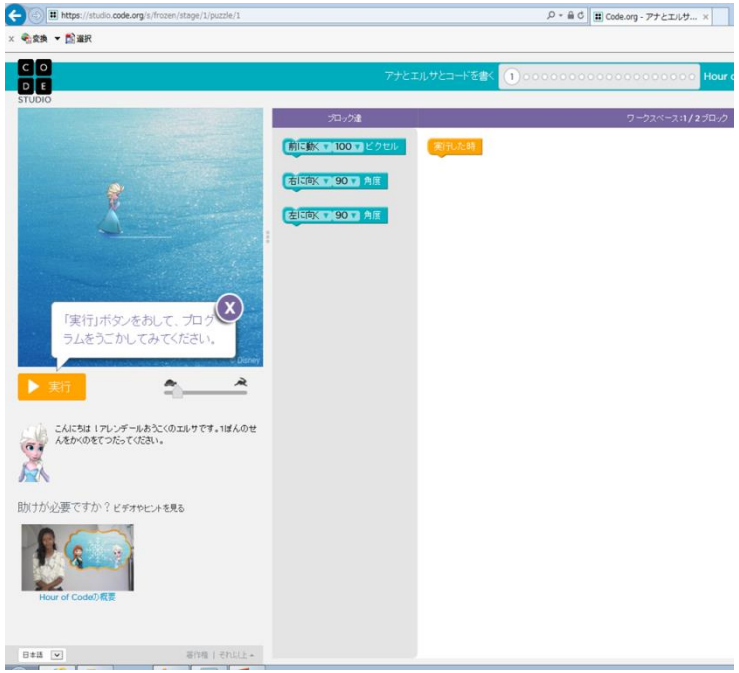


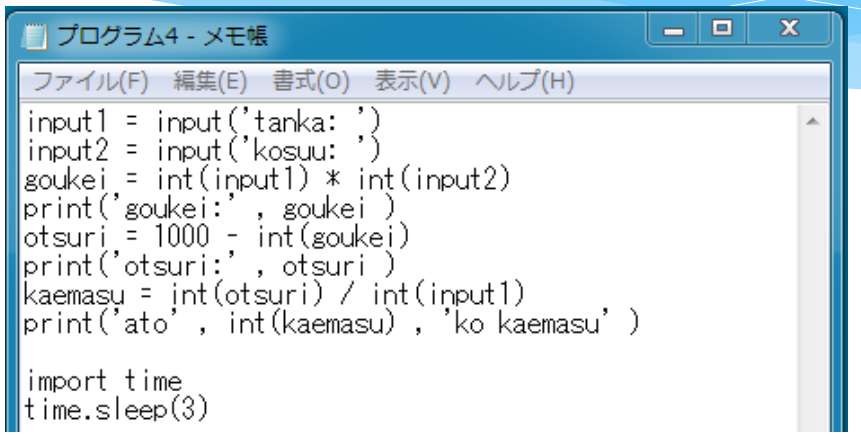
code.studioを用いた　授業案と実践報告

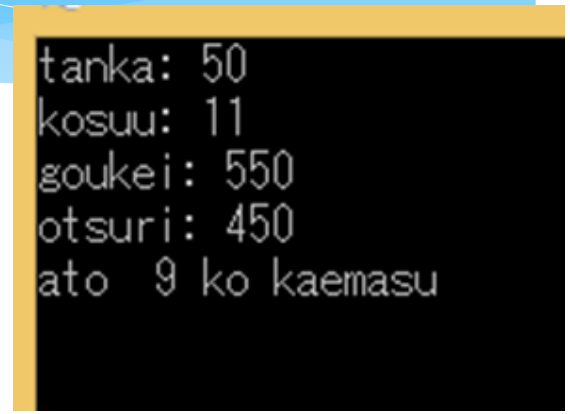
5. VBScriptとPythonを使ったプログラミング

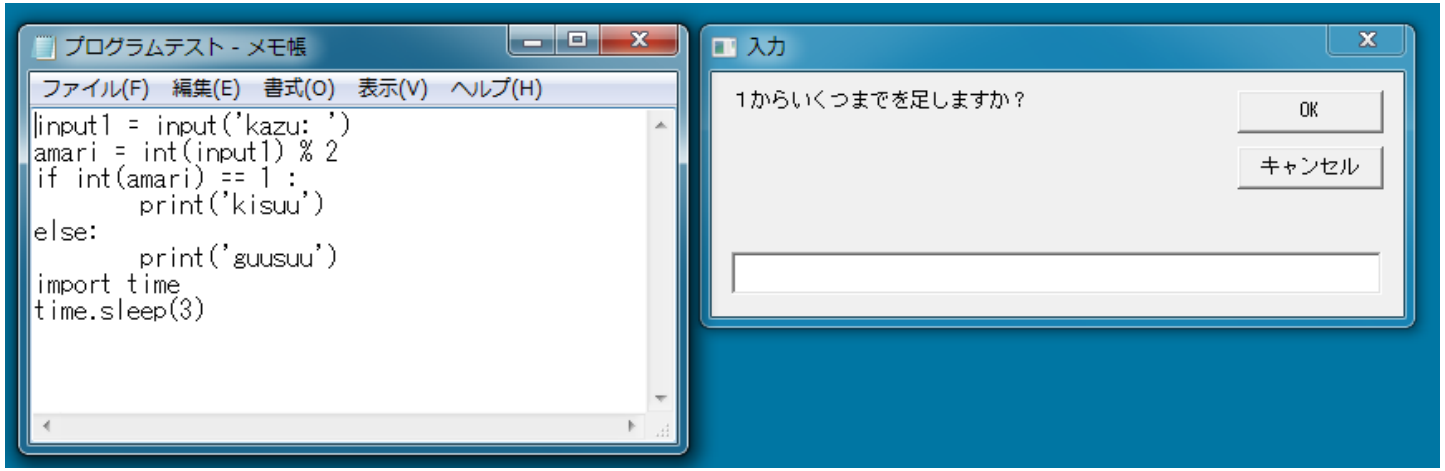
5. ブロック→プログラム確認→プログラミング

- code.orgの実行
  プリントでの復習（1・2限目）
- VBScriptまたはPythonを利用した
  プログラミング学習（3～6限目）









実施内容と振り返り

Code.orgはプログラミングの3つの基本構造の理解につながる

- プログラムをヒント無しで、自力で作れる生徒は1～2割程度
- 原因は、変数の宣言の理解に多く時間を取れなかったことや、
  剰余の計算等の数学的思考力が不足していたため、
  時間内にコーディングまでたどり着けなかったと考えられる。

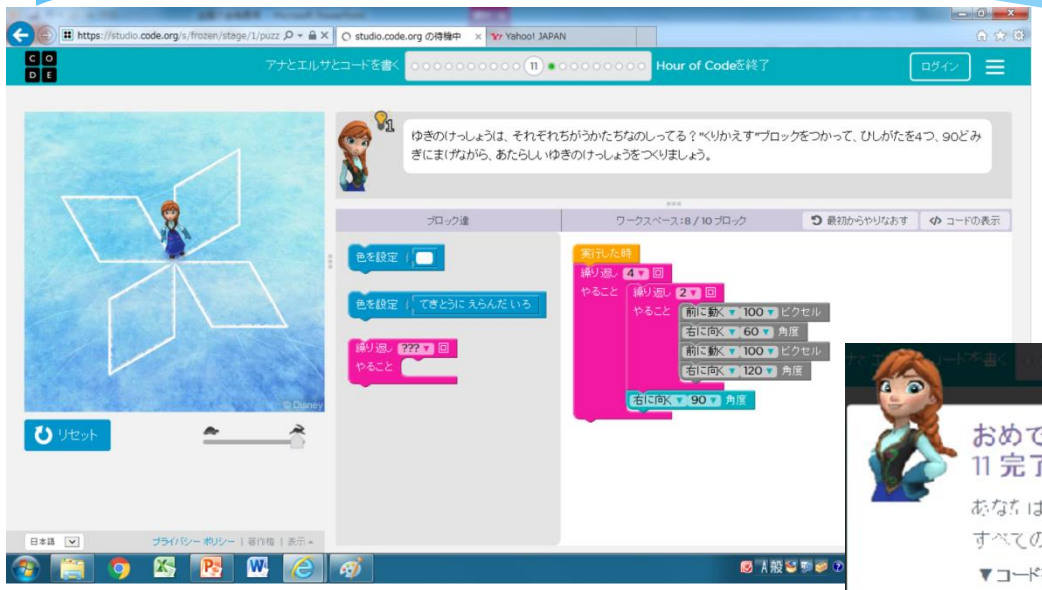
- 今後の課題
  変数の扱い方の学習方法
  エラーの発見をしやすくするエディタ
  について研究を深めるといったことがあげられる。

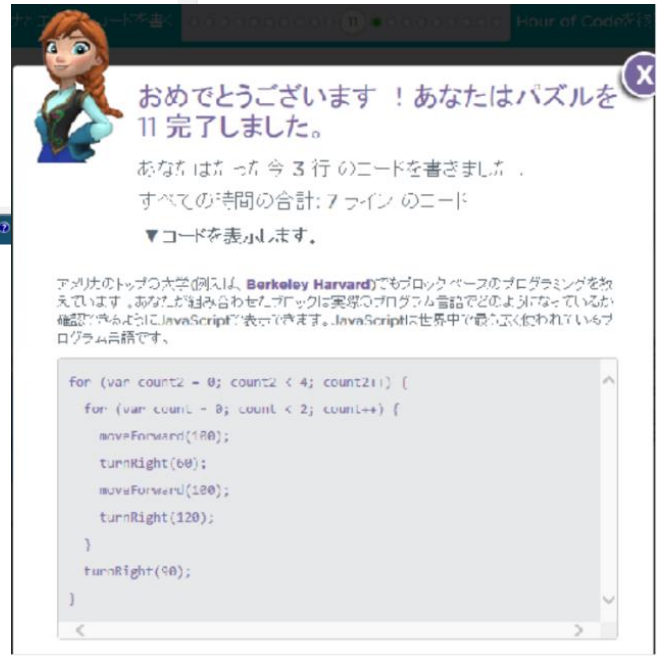
本授業実践ではVBScriptの方が適切だと感じた。

code.studioを用いた　授業案と実践報告

6. 身近なプログラムのロジックを知る

6. ブロック→プログラム確認→コーディング





Java･Script でプログラムを組んでみよう！

メモ帳で作る

保存するときには□拡張子.html（すべてのファイル）

保存先は□３〇〇〇→プログラム（新しいフォルダを作る）

実行は web ファイルで開く

課題１□パスワードのチェック□□□□□ファイル名□pw.html
□□予め４桁数字のパスワードを設定しておき、入力されたパスワードが正しいかどうかの判定をする（例□ログインできました□ログインに失敗）

課題２□おみくじ□□□□□□□□□□ファイル名□kuji.html
□□□予め任意の数字にコメントを用意しておく
□□□数字が入力されるとコメントが表示される

課題３□奇数・偶数判定□□□□□□□ファイル名□odd.html
□□□□好きな整数を入力すると奇数か偶数かを判定する

課題４□１から Ｎまでの総和を求める□ファイル名□sum.html
□□□□好きな整数を入力すると１からその数字まですべての整数を足した合計を計算する
□□□□例□５を入力すると□１＋２＋３＋４＋５の答えを計算して出力する

課題５□素数判定□□□□□□□□□ファイル名□prime.html
□□□□好きな整数を入力するとその数字が素数かどうかを判定する

実施内容と振り返り

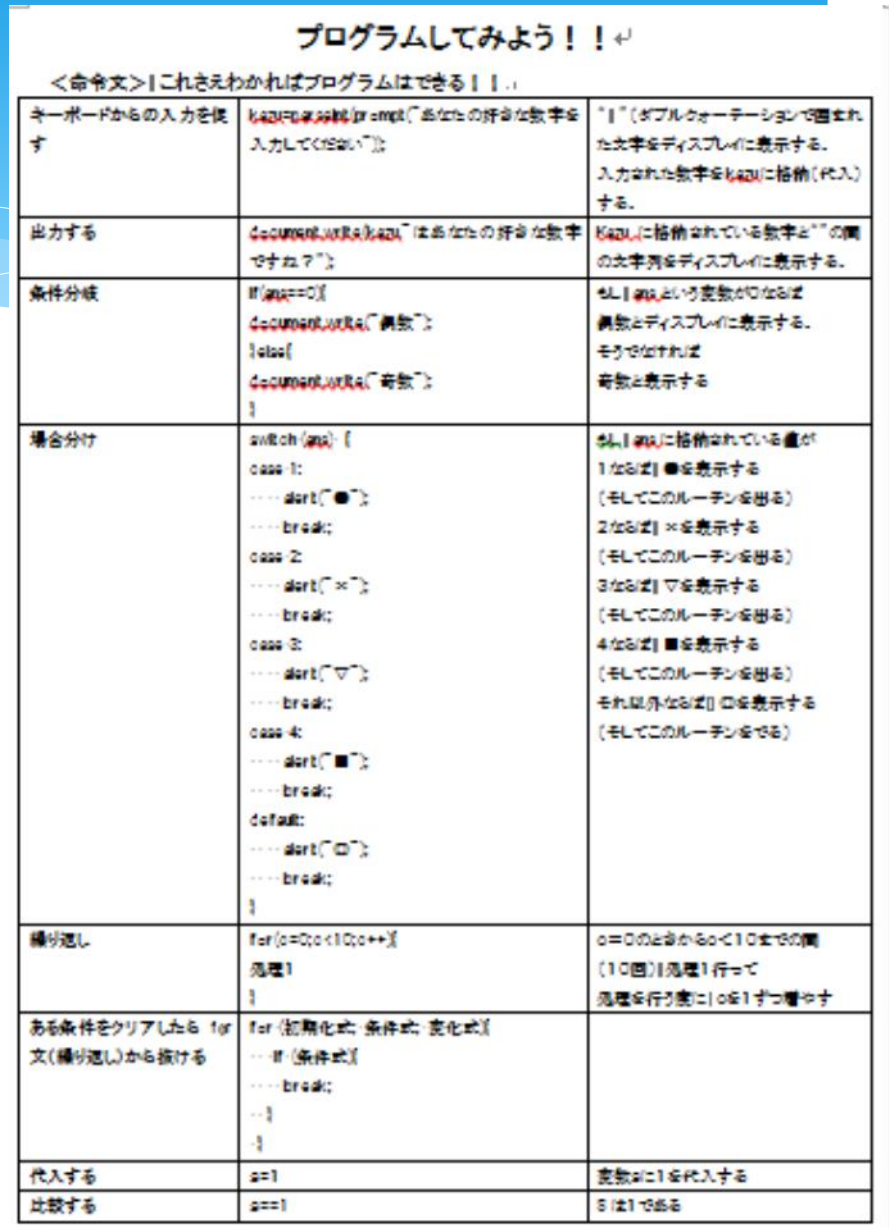
フローチャート　→　プログラミング　だった流れを
Code.org　　→　プログラミング　へ変えた。
興味を持たせやすく処理手順の言語化がしやすい

課題解決に数学的理解が必要（課題の内容要検討）

自力でできた生徒は1割程度

↓追加

今年は1学期に実施し、実施時間を増やしたところ、
全体の理解度がかなり高まった。
プログラミング実習にはある程度の時間をかけ、
慣れさせることが必要と感じた



プログラムしてみよう！！

＜命令文＞これさえわかればプログラムはできる！！

キーボードからの入力を使う	kazu=window.prompt("あなたの好きな数字を入力してください");	"||"（ダブルクォーテーションで囲まれた文字をディスプレイに表示する。 入力された数字をkazuに格納（代入）する。
出力する	document.write(kazu"はあなたの好きな数字ですね？");	kazuに格納されている数字と""の間の文字列をディスプレイに表示する。

条件分岐	if(gyo==0){
	document.write("偶数");
	}else{
	document.write("奇数");
	}
		もし||gyoという変数が0ならば
		偶数とディスプレイに表示する。
		そうでなければ
		奇数と表示する

場合分け	switch (gyo) {
	case 1:
	……alert("●");
	……break;
	case 2:
	……alert("×");
	……break;
	case 3:
	……alert("▽");
	……break;
	case 4:
	……alert("■");
	……break;
	default:
	……alert("□");
	……break;
	}
		もし||gyoに格納されている値が
		1ならば|| ●を表示する
		（そしてこのルーチンを出る）
		2ならば|| ×を表示する
		（そしてこのルーチンを出る）
		3ならば|| ▽を表示する
		（そしてこのルーチンを出る）
		4ならば|| ■を表示する
		（そしてこのルーチンを出る）
		それ以外ならば|| □を表示する
		（そしてこのルーチンをでる）

繰り返し	for(o=0;o<10;o++){
	処理1
	}
		o=0のとおかるかく10までの間
		（10回）||処理1行って
		処理を行う度に1ずつ増やす

ある条件をクリアしたらfor文（繰り返し）から抜ける	for (初期化式; 条件式; 変化式){
	……if (条件式){
	……break;
	……}
	}	

代入する	a=1	変数aに1を代入する
比較する	a==1	aは1である

実践報告を終えて全体を振り返ると

* 2年同じテーマで研究を続けることで、
  落ち着いて計画的な授業実践ができた

* どの授業実践もcode.orgは1～2時間かけて実施している。
  - ✓ 導入に使うことによって、生徒の興味関心を引くことができた。
  - ✓ ハードルが低い・生徒が自分のペースで取り組める・理解が早い
  - ✓ 「アナ雪」と「古典的な迷路」など生徒の実情に合わせて教材が選べる
  - ✓ フローチャート・プログラミングそれぞれに移行しやすい

* 導入にcode.orgは有用　→　そのあとの教材が重要！

実践報告を終えて全体を振り返ると

* 生徒の実情に応じて、何を、どこまで教えるかは様々。
* ゴールはどこにすればいいのだろうか？

- ✓Wordのスキルアップ（反復の必要性）
- ✓フローチャートに落とし込む
- ✓身近なシステムを考えさせる
- ✓プログラムの改善案を考えさせる
- ✓プログラムを書かせる
- ✓フローチャートから問題解決

プログラミング的な考え方（論理的思考）と問題解決能力

実践報告を終えて全体を振り返ると

* 今回の授業実践は4～9コマかけている。
  * プログラミング学習にはある程度の時間が必要。
    - ✓「反復」の理解が難しい
    - ✓変数の概念・数学的知識
    - ✓フローチャートの書き方
    - ✓プログラミング言語の読み方
    - ✓数学的思考力が必要な課題になってしまう
    - ✓グループで行う話し合い　　・・・どれも時間がかかる

より効率よく、深い学習ができないか？

おわりに

目的　「論理的思考」「問題解決能力」を身につける

手段　「プログラミング学習」

目標　**楽しく、効率よく、深い学び**
ができる授業案を作っていこう

会誌第13号及び教材→埼玉県高等学校情報教育研究会Web
http://www2.spec.ed.jp/krk/jyoho/comm2/?page_id=245

ご清聴ありがとうございました。