きのくに
ICT教育

高等学校プログラミング教育

学習指導案集

Python 版

和歌山県教育委員会

令和６年度　追補版

は じ め に

平成 31 年３月，和歌山県教育委員会は，全国に先んじて独自のプログラミング教育の指針（きのくに ICT 教育）を策定し，小学校・中学校・高等学校及び特別支援学校において，発達段階に応じた体系的なＩＣＴ教育のカリキュラムを示しました。高等学校では，JavaScript をベースとしたクラウド型アプリ開発環境「Monaca」を用いて，モバイル端末のアプリ開発をおこなうなど，実践的で特色ある授業展開を実施してきました。

小・中学校では，この間に GIGA スクール構想によって１人１台端末環境が整い，コンピュータルームでおこなわれていたプログラミングの授業が，普通教室でも自宅でも実施できるようになりました。これを受けて，小学校では，ブロック言語を用いて，プログラミングで思考・創作する楽しさを学ぶ，中学校では，micro:bit ををを用いてプログラミングを活用した問題解決的な学習を実施する新たな指導案を追加しました。

また，高等学校では，令和７年度大学入学共通テストから「情報Ｉ」が課せられるようになりました。「情報Ｉ」のテストにおいては，プログラム言語によらない疑似言語（ＤＮＣＬ）で出題されますが，テキスト形式のプログラミングの操作や手順，アルゴリズムの理解が必要な中，「情報Ｉ」の授業でのプログラミング指導は多くの困難さを抱えています。そこで，小学校・中学校で体系的にに学んできた創作的・問題解決的なプログラミングの取組を継承しつつ，汎用的なテキストプログラミングの基本を学び，また，共通テストの出題範囲である「コンピュータとプログラミング」にも対応できることを目指し，この追補版を作成することとなりました。この追補版は，Python をベースとした Google Colaboratory と JavaScript をベースとした p5.js を用いて，２つのメジャーなプログラミング言語を学ぶことができ，自己のペースに合わせて，自学自習できる内容となっています。

今や，プログラミングは，生成 AI を用いれば簡易に作成できるようになりました。しかしながら，プログラミングを体系的・系統的に理解するためには，コード入力における基本的な操作手順の学習は重要であり，普遍的であると考えられます。また，論理的思考力を生かし目的に応じたアルゴリズムを考え適切な方法で表現する能力や，発想力を生かしプログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善する能力の育成は，情報の授業を通して身に付ける必要があります。本書は，こういった資質・能力の育成の一助となることを目指したものであり，生徒達にプログラミングの面白さや有用感を改めて実感してもらえることを願っています。

令和 7 年 3 月

和歌山県教育委員会

情報Ⅰ　コンピュータとプログラミング

授業タイトル			当冊子	生徒ﾃｷｽﾄ
基礎編	（1）	テキストプログラミングの基本操作	P6-10	1. Google Colaboratory ～ 3. 変数
	（2）	多様な「条件分岐」の方法について学ぶ	P11-15	4．条件分岐
	（3）	「配列変数」の使い方について学ぶ	P16-19	5．配列
	（4）	ループ（for と while）の使い方	P20-22	6．ループ
	（5）	関数の定義とその活用	P23-25	7．関数
発展編	（6）	プログラミングでのグラフィックス	P26-29	8．グラフィックス
	（7）	幾何学模様を描こう	P30-33	9．幾何学模様
	（8）	デジタル単語帳	P34-38	10. デジタル単語帳
応用編	（9）	Hit and Blow：数字当て	P39-42	11. Hit and Blow
	（10）	「迷路」の作成と解法	P43-47	12. 迷路を自動生成するには

※上記の解説については，各 OS やアプリのバージョン，各自治体の導入している各タブレット端末のセキュリティ設定等の条件によっては動作しない場合があります。

＃本時展開内の記号の解説
◯ 主要な学習活動及び指導・支援（必須の学習・指導事項）
・ 細部の学習活動及び指導や支援（原則的には行うが，場合によってはスキップしても構わない）
※ 補足・諸注意及び応用等（必ずしもしないといけないわけではない）

情報Ⅰ

コンピュータとプログラミング（Ⅰ）

テキストプログラミングの基本操作

授業概要

小学校の各教科や中学校の技術・家庭科（技術分野）で行なってきたブロック形式のプログラミングを経て，高等学校の情報科では，テキスト形式のプログラミングが本格的に行われる。情報Ⅰの「(3) コンピュータとプログラミング」では，アルゴリズムを考え，プログラミングによって諸課題の解決を図ることが求められている。そのために，本時にて，まずはテキスト形式のプログラミングの基本操作及び文字・数字の表示方法，数値の計算方法，変数の使い方等を学ぶこととする。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

→上記を目指すために，本時では，まずプログラミングの基本的な操作を学ぶ。これまでの小・中学校におけるブロック形式のプログラミングとは異なり，テキストコードをキーボードで入力し，プログラム記述上の基礎的な決まりなどを理解する。

前提条件

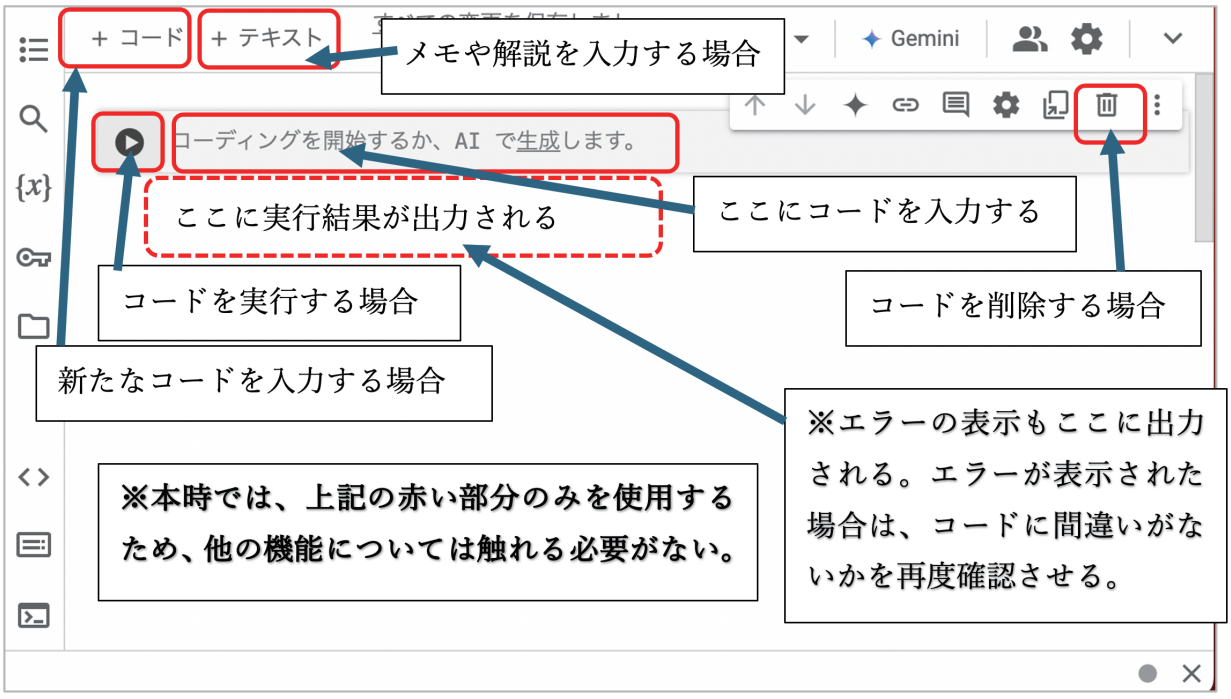
- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。

準備

- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF で各 PC へ配布する。本時では１．Google Colaboratory から３．変数までの項目を扱う）。

- プログラミング言語についての説明資料を準備する（テキスト形式のプログラミング言語は数百種類あると言われているが，それぞれの言語の特徴や用途を説明するための資料を準備する）。

- Google Colaboratory のサイトの基本的な説明画面を準備しておく。▶の部分からテキストコードを入力することや，▶を押すとプログラムが実行されることなどを示しておく。

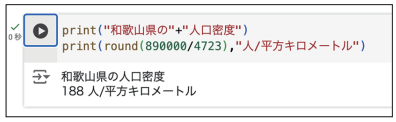


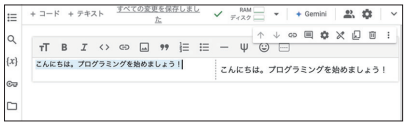
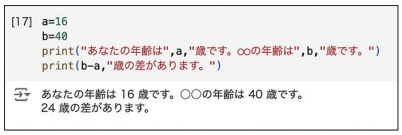
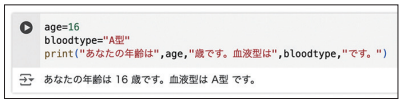
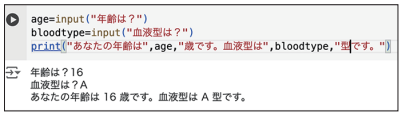
※各時限の学習目標は，最低限達成させていただきたい目標として設定しています。これを元に，指導者の意図に応じて，更に目標をアレンジ･付加することで，より横断的･発展的な学習活動につなげて下さい。

授業展開例（１～２時限分での実施を想定）

学習目標

●テキストプログラミングの基本的な操作を理解し，サンプルプログラムの入力・実行ができる。また，Python の基本コード及び記述上の決まりを理解した上で，アレンジしたり組み合わせたりすることができる。

	指導過程	教師の働きかけ	備考
導入	■テキスト形式のプログラミング言語について，その種類，特徴や用途について知る。	■プログラミング言語について，その種類，それぞれの特徴や用途について解説する。 ※スライドサンプル (1) を表示する。	※生徒らに，どのようなプログラミング言語が存在するのか，それぞれどのような特徴・用途があるのかを調べさせても良い。
展開1	■ Google Colaboratory のサイトにアクセスする。 ※初期のログイン設定，新規ファイルの作成などを行う。 《別紙生徒テキスト》の「１．Google Colaboratory」を参照	■これからブラウザ上で実行できる Google Colaboratory を用いて，Python をベースにプログラミングを学ぶことを伝え，その基本的なアクセス方法・初期設定，画面構成等について解説する。	※「corab」のキーワードで検索を行うよう伝えると，自宅からでもアクセスできるようになる。
展開2	■文字や数字を表示するプログラムを入力して実行する。 《別紙生徒テキスト》の「２．数値型と文字列型」を参照 ・プログラム入力の支援機能について知る。 ・プログラムの実行方法や新規のプログラムの作成方法について知る。 ・複数行のプログラムを入力し実行する。	■ 共通課題として，下記のようなプログラムコードを入力・実行させる。  ・▶を押せばプログラムが実行され，その結果が下部に出力されること，「＋コード」を押せば，次のプログラムが書けることなどを説明する。 ・上記の文字表示と数字表示を組み合わせて，複数行のプログラムを書くことを伝える。  ▲複数のプログラムを入力した例 ※ここでは，round 命令で小数を四捨五入している。	※ print と入力すると，候補が表示されること，パラメーターの入力＝”” が自動で表示されることなど，プログラミング入力の支援機能について知らせる。 ※文字同士を＋で繋いだり，「,」を入力して数値と文字を連続して表示するなども可能。

	指導過程	教師の働きかけ	備考
	※プログラミングとは別に解説メモや振り返り等の文章を記述する。	※ corab のサイトの特徴として，プログラミングとは別に文章を書いておけることを知らせる。 	※「＋テキスト」を押して，文章を入力しておける。「プログラムの解説」や「授業の振り返りメモ」などを残しておける。
展開3	■「変数」を扱うプログラムを入力する。 《別紙生徒テキスト》の「3．変数」を参照 ・《別紙生徒テキスト》を参照しながら，変数同士での四則演算を行う。特に，割り算・掛け算には特有の記号を用いることを理解する。 ・変数には自由な文字が使えることを理解する。 ・変数はプログラムの実行時に入力することができることを知る。	■ 数値や文字を代入したもの＝変数の扱いについて説明し，変数を扱うプログラムを入力させる。 ・一例として，下記のように一定の意味のあるフレーズを作成させてみる。  この場合，a や b といった変数では，何を意味するかが分かりづらいために，下記のように単語を変数にすることができることを知らせる。  ・応用編として，年齢や星座などを入力するプログラムにも変更できる。 	※ corab では，数値も文字も「変数」で同じように扱える。 ・変数同士での四則演算を行うような課題を出す。 ※但し，コードとして使う単語は変数としては使用できない。 ※この場合，年齢も文字列として扱われるため，数値として扱う場合は，int で文字列を数値化する必要がある。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの基礎的な知識や技能を振り返るとともに，これらの情報技術が身近な社会でどのように役立てられているかを考える。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。

※ corab は，基本的には自動保存されるため，保存方法についての解説は必要がありません。しかし，作成したプログラムをダウンロードして提出させたり，クラス内で共有する場合は，クラウドへの保存・ダウンロードの方法，拡張子の説明などを行っておく必要があります。

※スライドサンプル（１）＝プログラミングの説明に関するスライドの例

情報Ⅰ

コンピュータとプログラミング（2）

多様な「条件分岐」の方法について学ぶ

授業概要

前回は，テキスト形式のプログラミングの基本操作及び文字列・数値の表示方法，数値の計算方法，変数の使い方などを学んできた。本時では，「条件分岐」のための if 関数（elif,else を含めて）を学び，条件に当てはまった場合と，そうでない場合にプログラムの実行の流れを分岐させる方法を理解する。条件分岐には様々な方法があること，複数の条件を設定できることなどを，サンプルプログラムの入力によって体験的に学ぶことを重視し，それらのプログラムを積極的にアレンジすることで，試行錯誤する楽しさも実感して欲しい。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。

→当授業では，プログラミングの基本的な動作である「条件分岐」を学ぶ。これまでの小・中学校におけるブロック形式のプログラミング（例えば Scratch の場合）でも，「制御」や「イベント」といった種類のブロック（もしならば）を用いてきた経験があるはずである，それらを思い出しながら，テキスト形式での条件分岐の記述方法等をおさえていきたい。

準備

●《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF で各 PC へ配布する。本時では「４．条件分岐」の項目を扱う）。

※各時限の学習目標は，最低限達成させていただきたい目標として設定しています。これを元に，指導者の意図に応じて，更に目標をアレンジ･付加することで，より横断的･発展的な学習活動につなげて下さい。

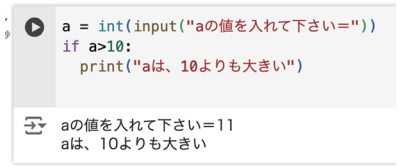
授業展開例（１～２時限分での実施を想定）

学習目標

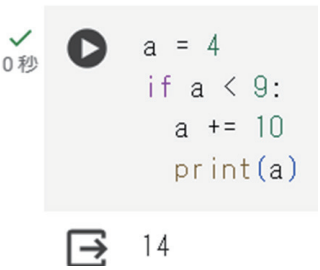
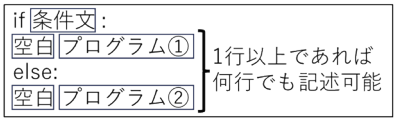
●条件分岐のためのプログラムコードの記述方法を理解し，それらを用いた基本的なプログラムを作成することができる。
●記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。

	指導過程	教師の働きかけ	備考
導入	■「条件分岐」について，これまでのブロック形式のプログラミングとの記述上の違いがあることを知る。	■これまでのブロック形式での「条件分岐」（もし□なら）を思い出させ，同様のプログラムを作成することを知らせる。	※例：Scratch の「作る」の画面の「制御」から，条件分岐に関するものを表示させる。

展開1	■Google Colaboratory のサイトにアクセスし，新しいコード入力のための準備をする。
	
	■《別紙生徒テキスト》の「図　4．2」のサンプルプログラムを入力・実行する。
	
	■サンプルプログラムを実行し，a=5 より大きい数値にすると，print 文が実行されること，5 以下の数値にすると print 文が実行されないことを確認する。
		■if 文の使い方の決まりを説明する。
		・《別紙生徒テキスト》の 4.1. の①～⑤を提示・参照させて，条件分岐の記述上の厳格な決まりについて知らせる。
		if 条件文:
		空白 条件が正しい時に実行するプログラム　} 1行以上であれば何行でも記述可能
		⋮
		（図 4.1）
		■共通課題として，条件分岐のサンプルプログラムを入力・実行することを指示する。
		0秒
		a = 10
		if a > 5:
		print("aは5よりも大きい")
		aは5よりも大きい
		（図 4.2）	

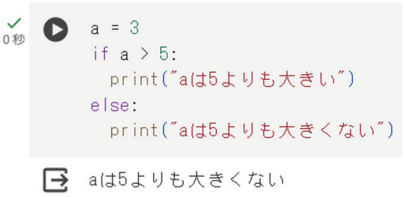
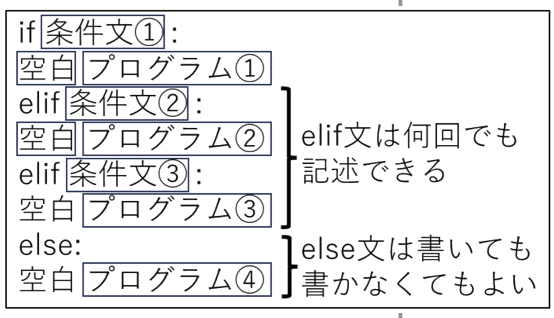
<table>
<tr><th></th><th>指導過程</th><th>教師の働きかけ</th><th>備考</th></tr>
<tr><td></td><td>■使用できる条件文について知る。
・ 下記の条件を理解した上で，実際にプログラムを変更して実行する。</td><td>※ a = 10 の箇所に input コマンドを使うと，数値を入力することで，変数 a に代入できる。
 
■下記の条件文について説明し，a の値や条件を変更して実行することを指示する。
・ 場合によっては print 内の記述を変更しても構わない。</td><td>※ input コマンドは文字列としての入力となるため，int にて，文字列から数値への変換を行う必要がある。</td></tr>
<tr><td>展開2</td><td>■《別紙生徒テキスト》の図 4.5，図 4.6，図 4.7 のプログラムを順次入力し，条件分岐の多様なパターンを試行する。

■練習問題４－１を行う。</td><td>■図 4.5（複数行の実行），図 4.6（計算式を条件に設定する），図 4.7（偶数を判定する）のサンプルプログラムを順次，入力・実行するよう指示する。
※すべて実施する必要はなく，指導者の実演でも構わない。

■練習問題４－１を行うことを指示する。
※早くできた生徒には，input コマンドを使うことや，条件に当てはまらない場合は，別のメッセージを出す方法を考えさせる。</td><td> 
図 4.5 のサンプルプログラムの例</td></tr>
<tr><td>展開3</td><td>■条件に当てはまらない（条件を満たさない）場合に用いる命令（else）について知る。</td><td>■条件に当てはまらない（条件を満たさない）場合の処理方法について説明する。
 
（図 4.9）</td><td>※ else 文の使い方：条件文に当てはまればプログラム①を，当てはまらなければ②が実行される。</td></tr>
</table>

種類	記号	機能	使用例	使用できる型
等式	==	左辺と右辺が等しい	a == b	数値型・文字列型
不等式	!=	左辺と右辺が等しくない	a != b	数値型・文字列型
大なり	>	左辺の方が右辺より大きい	a > b	数値型
大なりイコール	>=	左辺が右辺以上の値である	a >= b	数値型
小なり	<	左辺の方が右辺より小さい	a < b	数値型
小なりイコール	<=	左辺が右辺以下の値である	a <= b	数値型

（表 2.1）

	指導過程	教師の働きかけ	備考
	■図 4.10 のサンプルプログラムを入力する。 ・ 変数 a に代入する数値を変更して，プログラムの実行が変化することを確認する。 ■練習問題４－２を行う。	■図 4.10 のサンプルプログラムのように else 命令を追加することを指示する。  （図 4.10） ■練習問題４－２を行うよう指示する。正しく実行できれば，アレンジを加えることを促す。	※ if 文をさらに重ねて様々な条件分岐を試させてもよい。
展開４	■さらに多くの条件を設定して，分岐できる方法があることを知る。 ■ 4.3.elif 文の解説から図 4.12 を参照する。 ■図 4.13，図 4.14 のサンプルプログラムを入力・実行する。 ・ 変数 a に代入する数値を変更して，プログラムの実行が変化することを確認する。 ■練習問題４－３を行う。	■さらに高度な条件分岐を行う方法があることを説明する。 ・ elif 文及び if,else を複数回連ねることで，より複雑な条件分岐をおこなえることを伝える。  （図 4.12） ■図 4.13, 図 4.14 のサンプルプログラムを入力・実行させ，変数 a を変えることで，どのような条件分岐の処理がおこなわれたかを確認する。 ■練習問題４－３を行うよう指示する。正しく実行できれば，アレンジを加えることを促す。	
配慮事項	《別紙生徒テキスト》4.4. if 文の入れ子」についても，原則的には展開（４）と同様に実施する。しかしながら，時間的・難易度の都合上，展開（４）は elif を用いず，「if 文の入れ子」で代替も可能である。または，いずれかを「解説のみ」として，サンプルプログラムの入力・実行については「elif」か「if 文の入れ子」かを選択させるなどをしても構わない。		

終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの基礎的な知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する	※「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。

情報Ⅰ

コンピュータとプログラミング（3）

「配列変数」の使い方について学ぶ

授業概要

これまでは，テキスト形式のプログラミングの基本操作及び文字・数字の表示方法，数値の計算方法，変数の使い方，多様な条件分岐の方法などを学んできた。本時では，これまでの「変数」の使い方を発展させて，「配列変数」について学びたい。１つの変数に複数の値を格納できる「配列変数」によって，大量のデータを簡単に扱えるようになるため，より高度な処理が可能となる。なお，本時では，サンプルプログラムの入力によって体験的に学ぶことを重視し，それらのプログラムを積極的にアレンジすることで，試行錯誤する楽しさも実感して欲しい。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。
- 「変数」の使い方，「条件分岐」について理解している。

準備

●《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF で各 PC へ配布する。本時では「５．配列」の項目を扱う）。

※各時限の学習目標は，最低限達成させていただきたい目標として設定しています。これを元に，指導者の意図に応じて，更に目標をアレンジ･付加することで，より横断的･発展的な学習活動につなげて下さい。

授業展開例（１～２時限分での実施を想定）

学習目標

●配列変数の記述方法を理解し，基本的な配列変数を組み入れたプログラムを作成することができる。
●記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。また，与えられた課題に対して，既習のプログラミング技能を生かして，適切なプログラムの構想や組み立てを行うことができる。

	指導過程	教師の働きかけ	備考
導入	■「配列変数」について，通常の変数との使い方の違いについて知る。	■配列変数の使い方の例を示し，通常の変数との違いや，配列変数によってできるようになること，その利点等を説明する。	※ Scratch の「変数」での「リスト」が，この配列変数に該当するため，既習事項として取り上げてもよい。
展開1	■ Google Colaboratory のサイトにアクセスし，新しいコード入力のための準備をする。 ■《別紙生徒テキスト》の「図 5. 2」のサンプルプログラムを入力・実行する。 ■サンプルプログラムを実行し，a[1] の数値を 0 や 2 に変更することで，どの単語が表示されるか予想して確認する。	■共通課題として，配列を用いたサンプルプログラム（図 5.2）を入力・実行することを指示する。  ``` a = ["apple","banana","orange"] print(a) print(a[1]) ['apple', 'banana', 'orange'] banana ``` （図 5.2） ・ 特に，print(a) と print(a[1]) の違いを理解できているか，a[2] とすると，何が表示されるかを予想させる。	※ ここでは，３つの果物の名称を配列に格納しているが，個数やデータの種類は，適宜変更しても構わない。

	指導過程	教師の働きかけ	備考
	■配列に値を追加する方法や削除する方法について知る。 ■配列の長さを求める方法について知る。	・配列の数を増やしたり，サンプルになるフルーツ名を別のデータに置き換えるなどをしてもよい。 ■ append や pop といったコマンドを使用することで，配列の値を増やしたり，減らしたりできることを説明する（図 5.5 および図 5.6 のサンプルプログラムを用いる）。 ■ len コマンドを使用することで，配列の長さ（個数）を調べられることを説明する（図 5.7 のサンプルプログラムを用いる）。	※図 5.3 の文字列と数値が混在した場合や，図 5.4 のように，配列の 1 部を入れ替える方法も学ぶことが望ましい。

展開 2	■既習のプログラミング技能を発揮させて，与えられた課題のプログラムを構想する。
	
	■構想したプログラムを入力・実行する。
	
	・正常に動作するようになるまで，デバッグを繰り返す。
	・正常に動作した場合は，実行したプログラムを相互に確認する。
	
	■ここでは，５つの条件分岐を記述したが，もっと条件が増えた場合でも効率的に処理できる方法がないかを考える。⇒繰り返し処理（ループ）の学習へつなげる。
		■既習事項を使って，プログラムを作成してみましょう。
		【演習の例】
		1. 好きな食べ物（料理）の名前のベスト 5 を考え，１番から順番に配列変数に入れましょう。
		2.input を使って，食べ物（料理）の名称を入力できるようにします。
		3. 配列変数から，入力された食べ物（料理）の名称と同じものがあるかを調べます。
		4. 同じであれば，その「○○は，△番目に好きです！」と表示する。
		5. なければ，「それは好きな食べ物（料理）ではありません。」と表示する。
		※プログラムの例：
		[17] food = ["寿司","焼肉","かつ丼","ラーメン","カレー"]
		a = input("食べ物の名前を入れて下さい")
		
		if a == food[0]:
		print(a,"は、１番目に好きです！")
		elif a == food[1]:
		print(a,"は、２番目に好きです！")
		elif a == food[2]:
		print(a,"は、３番目に好きです！")
		elif a == food[3]:
		print(a,"は、４番目に好きです！")
		elif a == food[4]:
		print(a,"は、５番目に好きです！")
		else:
		print("それは好きな食べ物ではありません。")
		
		食べ物の名前を入れて下さい焼肉
		焼肉 は、２番目に好きです！
			※使用が想定されるコードの例
			food = [" 寿司 "," 焼き肉 "," かつ丼 "," ラーメン "," カレー "]
			a = input()
			if a == food[0]:
			print(a," は， 1 番目に好きです！ ")
			elif a == food[1]:
			print(a," は， 2 番目に好きです！ ")
			
			※５つの判定をするために，５つの条件分岐をあえて記述させた。これが，10 も 20 も続くようであれば，入力作業に手間がかかることを理解させ，繰り返し処理（ループ）の便利さに気づかせたい。

終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの基礎的な知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■さらに効率的にプログラムを組むために，「繰り返し処理」の方法について予告する。	※「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。

情報Ⅰ

コンピュータとプログラミング（4）

ループ（for と while）の使い方

授業概要

プログラミングの基本である順次処理・条件分岐の学習をこれまでおこなってきた。加えて，本時では，繰り返し処理（ループ）について学習する。繰り返し処理とは，①一定の回数を同じ処理を行う場合と，②条件に応じて同じ処理を行う場合があることを理解したうえで，それぞれの機能に適した使い方，使い分けができるようになって欲しい。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。
- 「配列変数」の使い方，「条件分岐」について理解している。

準備

- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF で各 PC へ配布する。本時では 6．ループの項目を扱う）。

授業展開例（1～2時限分での実施を想定）

学習目標

- ループ（繰り返し処理）の2通りの記述方法（回数で繰り返す場合＝「for 文」と条件を満たすまで繰り返す場合＝「while 文」）を理解し，それぞれの機能を使い分けながら，基本的なプログラムを作成することができる。
- 記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。また，与えられた課題に対して，既習のプログラミング技能を生かして，適切なプログラムの構想や組み立てを行うことができる。

	指導過程	教師の働きかけ	備考
導入	■前時で作成したプログラムを読み込む。 ・プログラム例では，5つの条件分岐が連続して記述されている。これ以上増えると，プログラムがしづらくなることに気づく。	■前時では，同様の処理を繰り返すプログラムを作成したことを思い出させる。 ・数回程度の繰り返しであれば，列挙した記述も可能ではあるが，100回・1000回になると到底記述できないため，本時では「繰り返し処理」について学ぶことを伝える。	
展開1	■for 文の使い方を学ぶ ・《別紙生徒テキスト》の図6.2及び図6.3を参照し，サンプルプログラムを入力・実行する。	■《別紙生徒テキスト》の図6.1のfor 文の記述の方法を説明する。 ・i は変数で，繰り返す回数の初期値であり，実行されるたびに，1ずつ加算されていくことを伝える。	for i in range(回数)： 空白プログラム
	・《別紙生徒テキスト》の図6.5を参照し，サンプルプログラムを入力・実行する。	■《別紙生徒テキスト》の図6.4のfor 文に配列を指定できることを説明する。	for i in 配列： 空白プログラム
展開2	■while 文の使い方を学ぶ ・《別紙生徒テキスト》の図6.7，図6.8を参照し，サンプルプログラムを入力・実行する。	■《別紙生徒テキスト》図6.6のwhile 文の記述の方法を説明する。	
	・《別紙生徒テキスト》の図6.10，図6.11を参照し，サンプルプログラムを入力・実行する。	■《別紙生徒テキスト》図6.9の配列を参照するwhile 文の記述の方法を説明する。	
展開3	■繰り返し処理のプログラム（練習問題6）を作成する。	■《繰り返し処理のプログラム（練習問題6）に取り組むことを指示する。	※問題文だけの提示もしくは，解答例の一部を表示する等のヒントを与えても良い。

	［練習問題６］配列変数内の変数の数だけ繰り返し，それぞれの数値を２倍して表示するプログラムを実行する。 ■【発展課題】前時のプログラム（if,elif 文を用いた条件分岐）を while 文を使ったプログラムに書き換える。	※解答例では，for 文を使っているが，「値が 1000 を超えた場合は，プログラムを終了する」といった while を用いたプログラムに変更するなどの発展課題を提示しても良い。 ※一斉もしくはグループ協議にて，プログラムの改良方法を話し合わせても良い。	※単純に配列内の数値を２倍するのではなくて，もっと複雑な計算式を書かせても構わない。 ※ while 文を用いるとどういった利便性が出るかについても考えさせたい。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの基礎的な知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。

情報Ⅰ | コンピュータとプログラミング（5） | 関数の定義とその活用

授業概要

これまで，プログラミングの基本である順次処理・条件分岐・繰り返しの学習をおこなってきた。これらのプログラミングの３大機能を学んだ上で，本時は「関数」について学ぶ。一連の処理をひとまとまりにすることを関数を定義すると言うが，この定義された関数を呼び出す事によって，特定の処理を必要に応じて何度でも実行することができるようになることを学ぶ。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。
- 「配列変数」の使い方，「条件分岐」「繰り返し処理」について理解している。

準備

- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF で各 PC へ配布する。本時では「７．関数」の項目を扱う）。

授業展開例（１～２時限分での実施を想定）

学習目標

●関数の定義の方法及び呼び出し方法について理解するとともに，「引数」と「戻り値」の記述の仕方についても理解する。

●記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。また，与えられた課題に対して，既習の技能を生かして，適切なプログラムの構想や組み立てを行うことができる。

	指導過程	教師の働きかけ	備考
導入	■プログラミングを行う上で，新たな機能について学ぶ。 ・「関数」の定義（宣言）方法と呼び出し方法を理解し，引数と戻り値についてもその使い方を把握する	■プログラムを組んで行く上で，一連の処理をひとまとまりとして定義する方法があることを説明する。 ・「関数」の定義（宣言）の方法と呼び出し方法の記述方法及び「引数」と「戻り値」の使い方について解説する。 ※関数で定義された変数は，ローカル変数と呼ばれ，関数内でしか使用できないことを説明する。	※記述方法の提示 def 関数名(): } 宣言 空白 プログラム 関数名() } 呼び出し （図 7.1） def 関数名(引数1,引数2,…): 空白 プログラム 関数名(値1,値2,…) （図 7.4）
展開1	■《別紙生徒テキスト》の図 7.6 を参照・入力して実行する。 ■《別紙生徒テキスト》の図 7.10 を参照・入力して実行する。	■《別紙生徒テキスト》の図 7.6 を参照・入力して実行することを指示する。 ■次に，「引数」と「戻り値」を使った記述方法に取り組むことを伝える。 ■《別紙生徒テキスト》の図 7.10 を参照・入力して実行することを指示する。	※配列の数を増やしたり，複雑な計算式に変更してもよい。
展開2	■《別紙生徒テキスト》の図 7.11 を参照し，入力して実行する。	■《別紙生徒テキスト》の図 7.11 のサンプルプログラムについて解説する。 ・10 の数値を変更するとどういう結果になるか，または数値を入力させるプログラムへの変更などを提案する。	※関数名は簡易なものに変更してもよい。

展開3	■《別紙生徒テキスト》の練習問題７を行う。 ■【発展課題】関数内の計算や処理をより複雑にするプログラムを考案する。 もしくは複数の関数を定義して，呼び出す関数を分けるなども考えさせたい。	■関数を定義して呼び出すプログラム（練習問題７）に取り組むことを指示する。 ※図 7.11 のプログラムとほぼ同じ内容なので，できるだけ自力で達成できるように促したい。 ※一斉もしくはグループ協議にて，プログラムの改良・発展方法を話し合わせても良い。	※問題文だけの提示もしくは，解答例の一部を表示する等のヒントを与えても良い。 ※これまで学んだ条件分岐・繰り返し処理などを想起させ，問題解決的に取り組みたい。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの基礎的な知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※ Google Colaboratory の「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。 ※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。

情報Ⅰ

コンピュータとプログラミング（6）

【発展編】プログラミングでのグラフィックス

授業概要

これまで，プログラミングの基本である順次処理・条件分岐・繰り返し及び配列や関数の定義・呼び出しといった学習を行ってきた。本時は，これまでの授業で習得した技能とコンピュータグラフィックスのコマンドを組み合わせた応用編という位置付けとなる。コンピュータでの基本的な描画処理の方法を学んだ上で,「プログラミングならでは」の描画手法を学ぶ。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- Google Colaboratory のサイトにアクセスすることができる。
- キーボードによってテキストコードを入力することができる。
- 「基本編」を一通り終えていること。
- 特に「関数」の定義・呼び出しの記述の仕方ついて理解している。

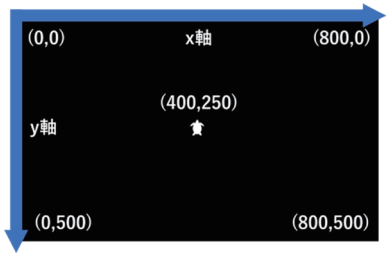
準備

- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか PDF にて各 PC へ配布する。本時では「発展編」の「8. グラフィックス」の項目を扱う）。
- [展開 3] まで，解説を詳細にしても 2 時限程度を標準と想定している。[選択・発展] の「正多角形を描くといったプログラムの改良に挑戦する。」は状況に応じて選択する。また，[展開 4] については，発展編「9. 幾何学模様」の 1 時限目として実施しても構わない。「9. 幾何学模様」を実践する場合は，[展開 4] は必須なので，授業計画に応じて時間数を設定する必要がある。

授業展開例（1 ～ 2 時限分での実施を想定）

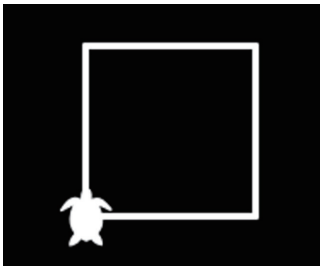
学習目標

- 描画コマンドを組み合わせて，意図した図形を描くことができる。
- 関数の定義の方法及び呼び出し方法や引数についての知識を応用することができる。
- 記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。また，与えられた課題に対して，既習のプログラミング技能を生かして，適切なプログラムの構想や組み立てをおこなうことができる。

	指導過程	教師の働きかけ	備考
導入	■本時で作成するプログラム及び達成目標について理解する。 ・ グラフィックコマンドを用いて図形を描画する。 ・ これまでに学習した「繰り返し」や「関数」を応用していく。	■これまでのプログラミングの発展編として，コンピュータグラフィックの基礎を学ぶことを説明する。 	※「タートルグラフィックス」の歴史や小学校でのプログラミングの復習などを取り入れてもよい。
	■ Google Colaboratory で，Turtle Graphics のインストール作業をおこなう。	■事前の準備として，「Turtle Graphics」のインストールの必要性及びその作業内容を指示する。	※インストールコマンドを提示する。 !pip3 install ColabTurtle
展開 1	■基本的な描画コマンド（直線を引く，角度を指示する）を理解する。 ・《別紙生徒テキスト》の図 8.3，図 8.4，図 8.5 に示されたコマンドを入力して図形を描画する。	■直線を引いたり，角度を変更するなどして，連続した直線を描画することを説明する。 ・《別紙生徒テキスト》の図 8.3，図 8.4，図 8.5 に示されたコマンドを入力することを指示する。	※基本的には下記の 4 つのコマンドを用いる。 forward(長さ) backward(長さ) right(角度) left(角度)

<table>
<tr><td></td><td>■基本的な描画コマンド（位置や色の指定等）を理解する
・《別紙生徒テキスト》の図 8.7. を参照・入力して実行する。</td><td>■座標（描画を行う起点）を指定することができること，描画する・しないの切り替え，色の指定について説明する。
・《別紙生徒テキスト》の図 8.7. を参照・入力して実行することを指示する。</td><td>※基本的には下記のコマンドを用いる。
goto (x,y)
penup()
pendown()
color()</td></tr>
<tr><td>展開2</td><td>■正方形を描く①
・《別紙生徒テキスト》の図 8.10 を参照し，サンプルコードを入力して実行。

■正方形を描く②
・繰り返しコマンド (for) でプログラムを短縮する。

・関数を用いて，同様の結果となるようにプログラムを改変する。
  </td><td>■正方形を描くプログラムを記述するよう指示を行う。
・《別紙生徒テキスト》の図 8.10 のサンプルプログラムについて解説を行うが，自力で達成できるよう促してもよい。

■正方形のプログラムを改良することを指示する。
・ for 文を入れて短縮化する。

・ 関数を用いて異なる大きさの複数の正方形を描いてみる。

※関数を用いる場合のほうがプログラムリストが長くなるが，後々のことを考えて，関数を用いる意味・意義を伝えておく。</td><td>4秒
initializeTurtle()
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)

4秒
initializeTurtle()
for i in range(4):
forward(100)
right(90)
（図 8.11）
※関数名は簡易なものに変更してもよい。

4秒
[12] def drawSquare():
for i in range(4):
forward(100)
right(90)

initializeTurtle()
drawSquare()
（図 8.12）</td></tr>
<tr><td>展開3</td><td>■正三角形を描くプログラムを作成する。
・《別紙生徒テキスト》の図 8.15 を参照し，サンプルコードを入力して実行する。

■関数を用いたプログラムに変更する。
・《別紙生徒テキスト》の図 8.16 を参照し，サンプルコードを入力して実行する。

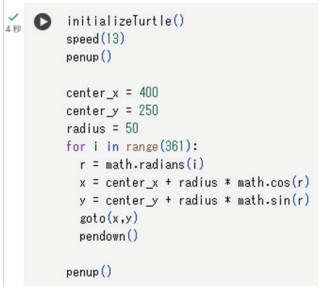
【選択・発展】正多角形を描くといったプログラムの改良に挑戦する。</td><td>■正三角形を描くプログラムを作成することを指示する。
・図 8.15 を提示し，角度の指定の仕方（外角を指定する）について説明する。

■関数を用いたプログラムに変更することを指示する。
・ サイズを変更して連続して描画することで，相似の図形を描画できることを説明する。

※正多角形を描く場合にはどのようにプログラムを改良すればいいのかを考えさせても良い。
※一斉もしくはグループ協議にて，プログラムの改良・発展方法を話し合わせても良い。</td><td>3秒
initializeTurtle()
for i in range(3):
forward(100)
right(120)

3秒
def drawTriangle():
for i in range(3):
forward(100)
right(120)

initializeTurtle()
drawTriangle()</td></tr>
</table>

展開4	■円を描くプログラムを作成する。 ・《別紙生徒テキスト》の図 8.18 を参照する。なお，関数の理解を完全にできている場合は，図 8.21 から実施しても構わない。	■円を描くためには，「三角関数」を用いることを説明する。 ・ 図 8.19 及び図 8.20 について解説する。 ※三角関数を未履修もしくは理解できていなくても，変数の一部として認識していればプログラムの学習自体は可能である。	 ※事前に，数学で三角関数（sin,cos）を履修しているかを確認すること。次章の「9 幾何学模様」を行うためにも，一定の三角関数の理解があることが望ましい。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で学んだプログラミングの知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※Google Colaboratory の「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。 ※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。

情報Ⅰ

コンピュータとプログラミング（7）

【発展編】幾何学模様を描こう

授業概要

前時では，コンピュータでの基本的な描画方法を学んだ上で，プログラミングならではの描画手法を学んだ。本時では，人間の手では描けないような幾何学的な模様を正確に素早く描画できるプログラムを作成していく。また，これまで学習した内容を活かして，オリジナリティのあるプログラミングへ発展させていきたい。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- 「基本編」を一通り終えていること。
- 特に「関数」の定義・呼び出しの記述の仕方ついて理解していること。
- 「発展編」の「8.グラフィックス」にて「TuttleGraphics」でのグラフィックス系の命令を理解していること。

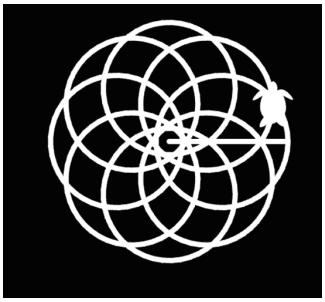
準備

●《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか PDF にて各 PC へ配布する。本時では「発展編」の「9. 幾何学模様」の項目を扱う）。
●[展開 2] まで，解説を詳細にしても 2 時限程度を標準と想定している。サンプルプログラムを配布するのであれば，1 時限に圧縮することも可能であると考えられる。[展開 3] で，オリジナルプログラムの作成を，どの程度行うかを考慮して，授業時間数を設定する。

授業展開例（1 ～ 2 時限分での実施を想定）

学習目標

●関数の定義の方法及び呼び出し方法や引数についての知識を応用することができる。
●グラフィック系の命令を必要に応じて使いこなすことができる。
●記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり組み合わせたりすることができる。また，与えられた課題に対して，既習のプログラミング技能を生かして，適切なプログラムの構想や組み立てをおこなうことができる。

	指導過程	教師の働きかけ	備考
導入	■本時で作成するプログラム及び達成目標について理解する。 ・グラフィックコマンドを用いて幾何学的な模様を描画する。 ・これまでに学習した「繰り返し」や「関数」を応用していく。 ■ Turtle Graphics のインストール作業ができていることを確認する。	■これまでのプログラミングの発展編として，コンピュータグラフィックによる幾何学模様を描くことを説明する。 ・普段目にするような幾何学的な模様があれば例に取り上げる。（数学の教科書に解説があれば，それを用いてもよい）。 ※事前の準備として，「Turtle Graphics」のインストールの必要性及びその作業内容を指示する。	※「幾何学模様」とは何かについての説明や，それらの例を示す。 
展開 1	■円を用いた幾何学模様を描画するプログラムを確認する。 ・《別紙生徒テキスト》図 9.1 及び表 9.1 のプログラム及び図 9.3 の解説を参照する。	■発展編「8。グラフィックス」にて円を描いたプログラムを提示して復習する。 ・《別紙生徒テキスト》図 9.3 について，sin,cos についての解説を行う（但し，完全に理解できていなくても，45 度間隔で角度をすることだけ理解できればプログラム自体の作成は可能である）。	・発展編「8. グラフィックス」のプログラムを生徒向けに配布できるように準備しておく。

<table>
<tr>
<td></td>
<td>■円を用いた幾何学模様を描画するプログラムを入力する。
・《別紙生徒テキスト》図 9.2 に示されたコマンドを入力して実行する。</td>
<td>

```
def drawCircle(radius, center_x, center_y):
 for i in range(46):
    r = math.radians(8 * i)
    x = center_x + radius * math.cos(r)
    y = center_y + radius * math.sin(r)
    goto(x,y)
    pendown()
 penup()

initializeTurtle()
color("white")
speed(13)

radius = 40
center_x = 400
center_y = 250
for i in range(8):
  r = math.radians(i * 45)
  x = center_x + radius * math.cos(r)
  y = center_y + radius * math.sin(r)
  drawCircle(50,x,y)
```

円を描く関数部分

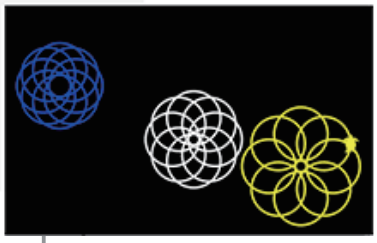
複数の円を重ねて複数回表示する部分
</td>
<td></td>
</tr>
<tr>
<td>展開2</td>
<td>■幾何学模様の描画自体を関数にしたプログラムへ改良する。

・《別紙生徒テキスト》の図 9.4 を参照し，サンプルコードを入力して実行する。</td>
<td>■幾何学模様の描画自体を関数にしたプログラムへ改良することを指示する。
・《別紙生徒テキスト》の図 9.5 のサンプルプログラムについて解説を行い，関数の中で，別の関数を呼び出しているところに注目される。


```
def drawCircle(radius, center_x, center_y):
  for i in range(46):
    r = math.radians(8 * i)
    x = center_x + radius * math.cos(r)
    y = center_y + radius * math.sin(r)
    goto(x,y)
    pendown()
  penup()

def drawGeometory(radius, center_x, center_y):
  for i in range(8):
    r = math.radians(i * 45)
    x = center_x + radius * math.cos(r)
    y = center_y + radius * math.sin(r)
    drawCircle(50,x,y)

initializeTurtle()
speed(13)
penup()

drawGeometory(40,400,250)
color("blue")
drawGeometory(30,150,150)
color("yellow")
drawGeometory(60,600,300)
```

円を描く関数部分

複数の円を重ねて複数回表示する部分


</td>
<td></td>
</tr>
<tr>
<td>展開3</td>
<td>（選択・発展）
■【挑戦してみよう！①】
色や大きさ・位置（座標）が異なる幾何学模様が一定の順序で表示されたり，ランダムに表示されるようなプログラムを考える。</td>
<td>※一斉もしくはグループ協議にて，プログラムの改良・発展方法を話し合わせても良い。</td>
<td></td>
</tr>
</table>

	■**【挑戦してみよう！②】** 円ではなくて，正方形や三角形で同様の幾何学的な模様を描かせるプログラムを考える。		・応用編「8 グラフィックス」の正方形や正三角形のプログラムを配布できるように準備しておく。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で新たに学んだプログラミングの知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※ Google Colaboratoryの「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。 ※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。

情報Ⅰ

コンピュータとプログラミング（8）

【発展編】デジタル単語帳

授業概要

発展編「10. デジタル単語帳」は，「配列」を用いて英単語と日本語訳を登録し，日本語で出題された単語に対応した英単語を，全て正しく回答できた場合に終了するプログラムを開発する。これまでの繰り返し処理や条件分岐，「random」や，「input」といった基本的なコマンドを用いつつ，「配列」を扱うことで，その活用方法への理解を深める。また，「クイズ」や「単語学習」のためのアプリとしての操作性を高める提案ができること，そしてプログラムを自らアレンジしていけることを目指す。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- 「while」，「if else」といった繰り返しや条件分岐のコードの記述の仕方ついて理解している。
- 「配列」の原理を理解して，プログラム上でどのように記載するかについて理解している。

準備

- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか，PDF にて各 PC へ配布する。本時では「発展編」の「10. デジタル単語帳」の項目を扱う）。
- [展開 1] まで，解説を詳細にしても 2 時限程度を標準と想定している。[展開 2] でのプログラムのブラッシュアップをどの程度実施するかで授業時間数を設定する。

授業展開例（2 ～ 3 時限分での実施を想定）

学習目標

- 配列変数の理解を定着させて，実用的なプログラムを作成することができるようになる。
- 「デジタル単語帳」の条件を理解した上で，そのアルゴリズムを予想する。
- 記述上の決まりを理解した上で，サンプルプログラムをアレンジしたり，組み合わせるたりすることができる。また，与えられた課題に対して，既習のプログラミング技能を生かして，適切なプログラムの構想や組み立てをおこなうことができる。

	指導過程	教師の働きかけ	備考

導入	■本時で作成するプログラム及び達成目標について理解する。
	・ 配列変数や random，input コマンド等これまで学習したプログラムを思い出す。
	
	※右の図 10. 3 のプログラムを入力・実行して，1 ～ 4 までのいずれかの数字が表示されることを確認する。
		■これまでのプログラミングの発展編として，配列変数や random を用いて「デジタル単語帳」を作成することを説明する。
		
		※図 10.2 及び図 10.3 のコードについての理解を確認して，必要に応じて，入力・実行させる。
		
		0秒
		`import random`
		`print(random.randint(1,4))`
		3
		▲図 10.3 の乱数を表示するコード
			※基礎編の「5. 配列」についての理解が十分できているかを確認しておく。
			
			※図 10.2 は，既習事項であり，input で入力した文字列を print 文で表示する。
			図 10.3 は乱数の数値を表示する。

展開 1	■「デジタル単語帳」の機能を踏まえて，どういったアルゴリズムが必要かを予想する（個人もしくはグループワーク）。 ※可能であれば，フローチャートの作成やどういったコードを用いるかの予想を行う。 ■考案したアルゴリズムを共有する。	■「デジタル単語帳」の作成条件にに基づいて，どういったアルゴリズムが必要かを発表させる。 ■他者のアルゴリズム（フローチャート等）と比較して，自分のものが正しい手順を踏まえているかを検討させる。	

	■それぞれが提案したアルゴリズムを確認して，プログラムとして入力した際に，正しく動作するかを検証する。		
展開2	■「○配列を用いた単語帳プログラムを確認する。 ・《別紙生徒テキスト》表 10.1 のプログラムの 3 行目までを参照する。 ※ 3 つの words と answers が入力されているが，これを変更してもよい。	■発展編「10. デジタル単語帳」の表 10.1 の上から 3 行目までの配列部分について説明する。 ・ 図 10.4 をもとに詳細に解説する。配列番号は [0] から開始されることや，len は length（＝長さ）を求めるコマンドであること等について理解させる。 ・ 表 10.1 では，日本語と英単語と対応させているが，ここは変更しても構わないことを伝える。また，words 及び answers の要素の数を増やしてもいいことを伝える。	※発展編「10. デジタル単語帳」の表 10.1 のプログラムを生徒向けに配布できるように準備しておく。 バナナを英語に訳してください 答え:banana 正解! みかんを英語に訳してください 答え:mikan 不正解... リンゴを英語に訳してください 答え:apple 正解! みかんを英語に訳してください 答え:orange 正解! 全問正解! ▲実行結果の例
	■生徒テキストの表 10.1 のプログラムとその解説の①から⑧までを読み，どのように対応しているかを理解する。 ・《別紙生徒テキスト》表 10.1 のサンプルプログラムを入力して実行する。	■生徒テキストの表 10.1 のプログラムについては，原則として解説の①から⑧までを順次補足を加えながら説明していくが，生徒らに読解させて自力で取り組ませ，質問にのみ対応するようにしても構わない。	

【《生徒テキスト》掲載のプログラム例と簡易解説（表 10.1)】

```
import random
words = ["リンゴ","みかん","バナナ"]
answers = ["apple","orange","banana"]

while len(words) >= 1:
    index = random.randint(0, len(words) - 1)
    print(words[index] + "を英語に訳してください")
    your_answer = input("答え:")
    if answers[index] == your_answer:
        print("正解!")
        words.pop(index)
        answers.pop(index)
    else:
        print("不正解")

print("全問正解!")
```

ランダムモジュールのインポート
出題する日本語の単語の配列を変数 words に代入
解答である英語の単語の配列を変数 answers に代入

words の長さが 1 以上の時に繰り返す
　index に 0 以上 len(words)-1 以下のランダム値の代入
　「words[index]を英語に訳してください」と出力
　your_answer にユーザの入力を代入
　もし、answers[index]と your_answer が等しい時
　　「正解!」と出力
　　words の index 番目を削除
　　answers の index 番目を削除
　違えば
　　「不正解」と出力

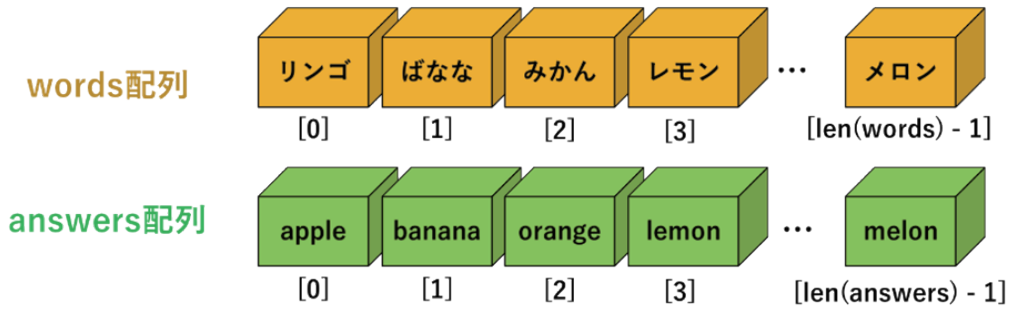
「全問正解!」と出力

展開3	■単語帳プログラムをブラッシュアップする。 ・単語帳プログラムがより使いやすく，より実用的になるように改良点を出し合う。（プログラミングできるかどうかは問わず多くのアイディアを出す）。 ・実現可能な改良プログラムを追加していく。	■単語帳のプログラムをブラッシュアップすることを指示する。 ・ブラッシュアップのアイディアを生徒から発想させる。 ・ 備考欄に記載したような例を提示してもよい。もしくは分担して作成させても構わない。 ※一斉もしくはグループ協議にて，プログラムの改良・発展方法を話し合わせても良い。	例：不正解の時に「×不正解。正解は■■です。」と出力する。 例：今何問中の何問目かを表示する。 例：「解説」やヒントを表示する。 例：タイマーをつけて所要時間を表示したり，制限時間を設けたりする。
終末	■自らのプログラムの工夫や試行錯誤した点などを共有する。 ■本時で新たに学んだプログラミングの知識や技能を振り返る。	■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，クラス内にて共有する。 ■このようなプログラムがどのような場面で活用されているかについて考えさせる。	※Google Colaboratory の「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。 ※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。

【サンプルプログラムの解説例】（《別紙生徒テキスト》にも掲載）

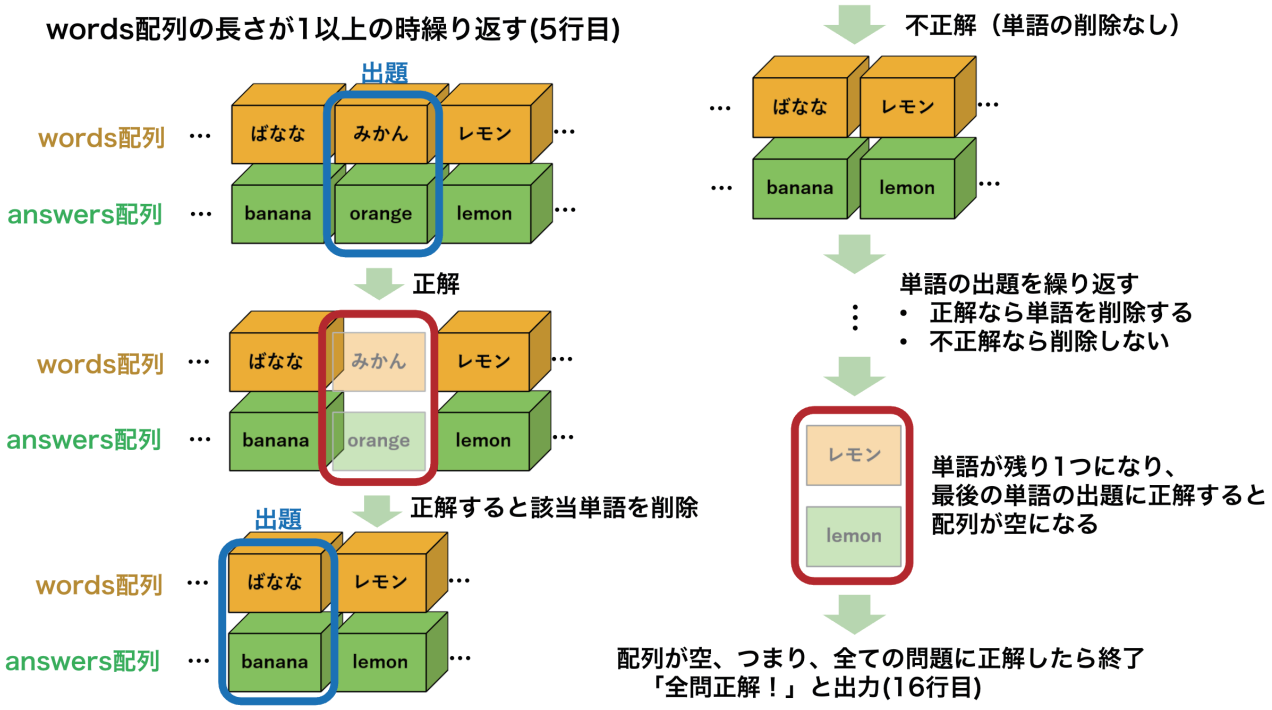
◯単語の設定（2,3 行目）

日本語と英語の単語セットを配列として宣言します。words 配列 0 番目の日本語の単語には，answers 配列 0 番目の英語の単語が，words 配列 1 番目の日本語の単語には，answers 配列 1 番目の英語の単語が…といったように，同じ番号同士の単語が対応します。



◯ words 配列の長さが 1 以上の時にずっと繰り返す（5 行目）

正解をした時，words・answers 配列が削除されるため，words 配列に含まれる要素は未出題または不正解の問題となります。全ての問題に正解をしたら要素が空になります。Words 配列が空になるまで繰り返されます。


words配列の長さが1以上の時繰り返す(5行目)
出題
words配列
answers配列
ばなな
みかん
レモン
banana
orange
lemon
正解
words配列
answers配列
ばなな
みかん
レモン
banana
orange
lemon
正解すると該当単語を削除
出題
words配列
answers配列
ばなな
レモン
banana
lemon
不正解（単語の削除なし）
ばなな
レモン
banana
lemon
単語の出題を繰り返す
• 正解なら単語を削除する
• 不正解なら削除しない
レモン
lemon
単語が残り1つになり、
最後の単語の出題に正解すると
配列が空になる
配列が空、つまり、全ての問題に正解したら終了
「全問正解！」と出力(16行目)

情報Ⅰ

コンピュータとプログラミング（9）

【応用編】Hit and Blow：数字当て

授業概要

「11. Hit and Blow：数字当て」のプログラムを作成するために必要となるコードの使い方を事前に学習する。数値と文字列を変換したり，配列の位置を参照するためのコマンドなど，新たなコードの使い方を学んだ上で，プログラムの作成を行う。なお，数当てのルールを厳格に適用するための工夫（＝エラーを起こさないための配慮）を付け加えて，プログラムの誤動作への考え方，デバッグの重要性等についても理解させたい。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- 基礎編・発展編を理解している。（特に，「配列」の原理とプログラム上でどのように記載するかについての理解をしている。）。
- 一定のアルゴリズムを考案・図解することができる。

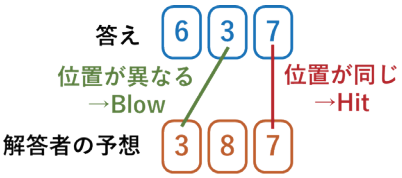
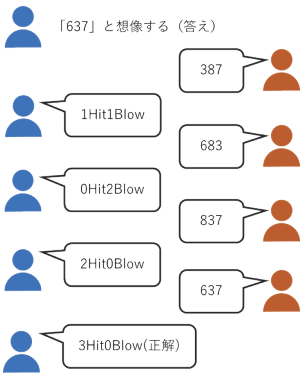
準備

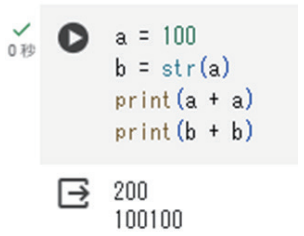
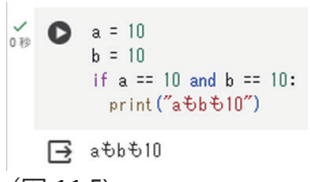
- 《別紙生徒テキスト》を準備しておく（生徒分を印刷しておくか PDF にて各 PC へ配布する。本時では「応用編 1」の「11. Hit and Blow：数字当て」の項目を扱う）。
- [展開 1] まで，解説を詳細にしても 2 時限程度を標準と想定してる。[展開 2] でのプログラムのブラッシュアップを，どの程度実施するかで授業時間数を設定する。

授業展開例（２～３時限分での実施を想定）

学習目標

- 数当てゲーム（Hit and Blow）のルールを理解した上で，そのアルゴリズムを予想する。
- 数当てゲームのサンプルプログラムの各所の動作を理解した上で，より改善したプログラムにすることができる（ユーザーに配慮し，誤動作をおこさないプログラムに修正できる）。

	指導過程	教師の働きかけ	備考
導入	■本時で作成するプログラム及び達成目標について理解する。 ■数字当てゲームのルールを理解し，どういったアルゴリズムが必要かを予想する（個人もしくはグループワーク）。 ※可能であれば，フローチャートの作成やどういったコードを用いるかの予想を行う。	■これまでのプログラミングの応用編として，３桁の数字を予想して当てる「数字当てゲーム（Hit and Blow）」を作成することを示す。 （詳細は《別紙生徒テキスト》に記載）  （図 11.7） ▲ゲームのルール	※この時点で，サンプルプログラムを配布・実行して，体験しながら，ゲームのルールを理解させても構わない（但し，アルゴリズムを予想することも学習目標であるため，プログラムリストはできるだけ見ないよう指示する）。
展開1	■考案したアルゴリズムを共有する。 ■それぞれが提案したアルゴリズムを確認して，プログラムとして入力した際に，正しく動作するかを検証する。	■ゲームのルールに基づいて，どういったアルゴリズムが必要かを発表させる。 ■他者のアルゴリズム（フローチャート等）と比較して，自分のものが正しい手順を踏まえているかを検討させる。	 （図 11.8） ▲ゲームルールを確認する際の提示例。
展開2	■サンプルプログラム（《別紙生徒テキスト》の「11.6. Hit and Blow をプログラムで記述する。」）を参照し，各部の動作について確認する。	■サンプルプログラム（《別紙生徒テキスト》の「11.6. Hit and Blow をプログラムで記述する。」）を参照させる。 ・ これまで未修得の新しい機能が４個所あることを伝える。	

	■未修得事項について確かめる。 ・学習事項①（数値を文字列に変換する）のサンプルプログラムを入力・実行し，その機能を理解する。 ・学習事項②（文字列の n 番目を取得する）のサンプルプログラムを入力・実行し，その機能を理解する。 ・学習事項③（in 演算子の使い方）のサンプルプログラムを入力・実行し，その機能を理解する。 ・学習事項④（条件式の「かつ」「または」）のサンプルプログラムを入力・実行し，その機能を理解する。	■未修得の新しい機能について，確認していく。 ※左の学習事項①～④については，できるだけ生徒による入力・実行をおこなう。 ■以下の４つの使い方について解説する。必要に応じてサンプルプログラムを入力して各コードの使い方を習得させる。 ・数値を文字列に変換する方法 ・文字列の n 番目を取得する方法 ・in 演算子の使い方 ・条件式の「かつ」「または」の使い方	 (図 11.1) ▲数値を文字列に変換  (図 11.2) ▲[1 番目] の文字列を取得  (図 11.3) ▲文字列を for 文で参照  (図 11.4) ▲特定の文字列が含まれるかを in 演算子で確認  (図 11.5) ▲条件式での「かつ」(and) の使い方
展開 3	■ユーザが誤って入力した時に，正しく入力するように教えるプログラムに改良する。 ※より，配慮された（誤動作しない）プログラムに仕上げる。	■プログラムに改良を加えることを指示する。 ・プログラム例では，入力する数字が 3 桁の数字でなかったり（例：26 や 1543），3 桁の数字に同じ数字があったりしても入力できてしまう（例：646 や 355）ことを伝える。	※改良されたプログラム例を示して，どこの部分がどのように改良されているかを考えさせても構わない。

<table>
<tr>
<td>展開3</td>
<td>・3桁の数字しか入力を受け付けないように改良する。
・3桁の中に同じ数字があった場合に受け付けないように改良する。</td>
<td>

```
if len(user_input) != 3:
  print("数字は3桁の必要があります。")
elif user_input[0] == user_input[1] or
user_input[1] == user_input[2] or
user_input[0] == user_input[2]:
```

</td>
<td></td>
</tr>
<tr>
<td>終末</td>
<td>■自らのプログラムの工夫や試行錯誤した点などを共有する。
■本時で新たに学んだプログラミングの知識や技能を振り返る。</td>
<td>■本時のプログラムの工夫した点を発表させる。優れた応用プログラムがあれば，学級内にて共有する。
■このようなプログラムがどのような場面で活用されているかについて考えさせる。</td>
<td>※Google Colaboratoryの「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。
※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。</td>
</tr>
</table>

【サンプルプログラムと解説】（生徒テキストにも掲載）

```
import random
↵
numbers = []
for i in range(10):
  numbers.append(str(i))
↵
select_num = ""
for i in range(3):
  index = random.randint(0,len(numbers) - 1)
  select_num = select_num + numbers[index]
  numbers.pop(index)
↵
user_input = ""
count = 0
while select_num != user_input:
  user_input = input("数値を予測してください:")
  hit = 0
  blow = 0
  for i in range(len(user_input)):
    if user_input[i] == select_num[i]:
      hit += 1
    elif user_input[i] in select_num:
      blow += 1
  print(str(hit) + "Hit" + str(blow) + "Blow")
  count += 1
↵
print(str(count) + "回で正解しました！")
↵
```

① 0～9までの整数を配列として numbers に代入

② select_num に互いに異なる3桁の数字を代入する

③ユーザが数字を当てる

④何回で正解したかを出力する

情報Ⅰ

コンピュータとプログラミング（10）

【応用編】「迷路」の作成と解法

授業概要

毎回異なるルートの「迷路」を自動的に生成し，その最短ルートを求めるプログラムを考案する。まず，新規の内容として「二次元配列」の使い方を学んだ上で，迷路生成やその迷路を最短距離でゴールするルート探索のためのアルゴリズムを考える。これらの迷路生成・探索のアルゴリズムを可視化するためのシミュレーション教材を用いながら，そのプロセスや原理を深く学ぶことが主要な目的である。プログラミング自体は，これまで学習したコードの使い方を応用していくことで実現可能である。

単元目標（学習指導要領との関連項目）

【学習指導要領上の位置づけ】

情報Ⅰの「内容」の（3）コンピュータとプログラミング

ア 次のような知識及び技能を身に付けること。

(ｲ) アルゴリズムを表現する手段，プログラミングによってコンピュータや情報通信ネットワークを活用する方法について理解し技能を身に付けること。

(ｳ) 社会や自然などにおける事象をモデル化する方法，シミュレーションを通してモデルを評価し改善する方法について理解すること。

イ 次のような思考力，判断力，表現力等を身に付けること。

(ｲ) 目的に応じたアルゴリズムを考え適切な方法で表現し，プログラミングによりコンピュータや情報通信ネットワークを活用するとともに，その過程を評価し改善すること。

(ｳ) 目的に応じたモデル化やシミュレーションを適切に行うとともに，その結果を踏まえて問題の適切な解決方法を考えること。

前提条件

- 「基礎編・応用編で用いたコード及びその使い方について理解している。
- 「配列」や「関数」の原理を理解し，プログラム上でどのように記載するかについても理解している。
- 一定のアルゴリズムを考えることができ，それらを図示できる手法（フローチャートやアクティビティ図等）を学んでいる。

準備

●《別紙生徒テキスト》準備しておく（生徒分を印刷しておくか，PDF にて各 PC へ配布する。本時では「応用編 2」の「12. 迷路を自動生成するには」の項目を扱う）。また，サンプルプログラム部分をカットするなど，学習の目的に応じて《別紙生徒テキスト》をアレンジしておく。

・【導入】で使用する迷路アルゴリズムを理解するシミュレーターには事前にアクセスし，生徒は，どの程度のアルゴリズムを考案するかを決めておく。授業展開例では【導入】に実施することとしたが，【展開（1）】と変更しても構わない。

・【展開（2）】以降では，サンプルプログラムの提示方法について検討しておく必要がある。全てのコードを生徒に向けて提示するのか，小出しにするのか，一部を示して穴埋め式にするかなど，生徒自身がコードをどこまで記述するか（もしくはサンプルプログラムの読解やアレンジのみとするか）を事前に決めておく。

授業展開例（2 ～ 3 時限分での実施を想定）

学習目標

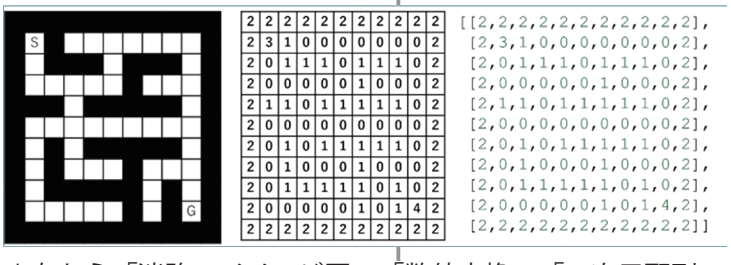
- ● 二次元配列の原理・使い方を理解する。
- ●「迷路」の条件を理解した上で，そのアルゴリズムを予想し，図示することができる。
- ● 既習のプログラミング技能を生かして，サンプルプログラムを読み解くことができる。

	指導過程	教師の働きかけ	備考
導入	■本時で作成するプログラム及び達成目標について理解する。	■これまでのプログラミングの応用編として，新たに二次元配列等を用いて「迷路」を生成し，最短距離を探索するプログラムを作成することを説明する。	※基礎編の「配列」についての理解が十分できているかを確認しておく。
	■迷路作成シミュレーターのサイトにアクセスする。 ■シミュレーターを実行しながら，作成するプログラムのイメージをつかむ（どういったアルゴリズムで動作しているかを捉える）。	■生徒らに，迷路作成のシミュレーター（右図）にアクセスさせて，作成するプログラムのイメージをつかませる。 https://kyurururn.github.io/maze_sample/ （※当サイトの迷路シミュレーターは本時のために独自作成したものである。実行のプロセス＝特に迷路パターンの作成などはしっかりと見て，どのようなアルゴリズムが必要かを考えさせたい。）	 ▲迷路作成シミュレーターの実行画面（黄色が最短コース）

<table>
<tr>
<td>展開1</td>
<td>■「迷路」の生成時には，どういったアルゴリズムが必要かを予想する（個人もしくはグループワーク）。
※可能であれば，フローチャートの作成やどういったコードを用いるかの予想を行う。

■考案したアルゴリズムをクラス内で共有する。</td>
<td>■迷路の生成条件を提示した上で，その生成条件に基づいて，どういったアルゴリズムが必要か考えさせる。
【迷路生成条件の例】
・プログラムを実行するたびに異なる迷路が作成されること。
・迷路の周囲は壁であること。
・スタート地点は左上，ゴール地点は右下として設定すること。

■他者のアルゴリズム（フローチャート等）と比較して，自分のものが正しい手順を踏まえているかを検討させる。</td>
<td>※「12.6 迷路の設計」を提示して，迷路のパターンが，どのように二次元配列に格納されているかを確認させておくこと。
・0 … 通り道
・1 … 内側の壁
・2 … 外側の壁
・3 … スタート
・4 … ゴール

（図 12.13）
 
▲左から「迷路のイメージ図」・「数値変換」・「二次元配列」</td>
</tr>
<tr>
<td>展開2</td>
<td>■「迷路」の作成のためのプログラムを確認する。
・どのような構造になっているか（特に，関数の使われ方），それぞれの関数はどのような機能を持っているかを考える。
・事前に考案していたアルゴリズムとコードを比較する。</td>
<td>■「迷路」の作成のためのプログラムを提示し，どのような構造になっているかを考えさせる。
・3 つの関数の定義をおこなっていることと，それぞれがどのような働きをしているかを理解させる。
・生徒らが考案したアルゴリズムと実際のプログラムコードとを比較させる。</td>
<td></td>
</tr>
<tr>
<td></td>
<td colspan="3">【生徒テキスト掲載のプログラム例と簡易解説】
（それぞれの関数の詳細な説明については《生徒テキスト》を参照してください）。

<pre>
import random

def display_maze(maze):
  for i in range(len(maze)):
    for j in range(len(maze[i])):
      if maze[i][j] == 3:
        print("S", end="")
      elif maze[i][j] == 4:
        print("G", end="")
      elif maze[i][j] == 2:
        print("■", end="")
      elif maze[i][j] == 1:
        print("■", end="")
</pre>
乱数のコマンドを使用できるようにする。

⇒display_maze 関数の定義

※ここでは、2 次元配列内の数値を記号（■や S や G）に変換して画面表示している。</td>
</tr>
</table>

```
      elif maze[i][j] == 0:
        print(" ", end="")
    print()

def grid():
  maze = []
  for i in range(HEIGHT):
    row = []
    for j in range(WIDTH):
      if i == 1 and j == 1:
        row.append(3)
      elif i == HEIGHT - 2 and j == WIDTH - 2:
        row.append(4)
      elif i == 0 or j == 0 or i == HEIGHT - 1 or j == WIDTH - 1:
        row.append(2)
      elif i % 2 == 0 and j % 2 == 0:
        row.append(1)
      else:
        row.append(0)
    maze.append(row)
  return maze

def create_maze():
    maze = grid()

    for i in range(2, HEIGHT - 1, 2):
        for j in range(2, WIDTH - 1, 2):
            directions = []
            if i == 2:
                directions.append([i - 1, j])
            if maze[i+1][j] == 0:
                directions.append([i + 1, j])
            if maze[i][j-1] == 0:
                directions.append([i, j - 1])
            if maze[i][j+1] == 0:
                directions.append([i, j + 1])
            wall = random.choice(directions)
            maze[wall[0]][wall[1]] = 1
    return maze

HEIGHT = 11
WIDTH  = 11
display_maze(create_maze())
```

⇒grid 関数の定義

※ここでは、迷路の外側の枠や迷路内の基本となる壁部分を作成している。

⇒create_maze 関数の定義

※ここは、迷路内の壁と通路のパターンをランダムに形成するプログラムである。

・迷路の高さと幅を設定。

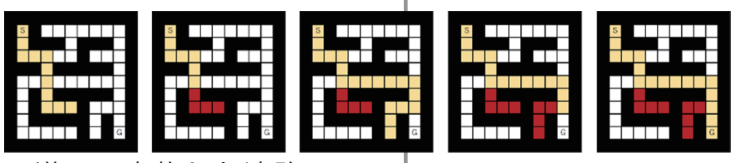
・関数の実行（迷路の表示）を行う。

展開3

■「迷路」を最短距離で抜けるためのルートを探索するために，どういったアルゴリズムが必要かを検討する。

- 導入で使用した迷路シミュレーションを再度見直す。
- 生徒テキストの「図 12.11」（右図）の 5 つの画像を見て，そのアルゴリズムについて協議する。

■作成された「迷路」のスタート地点（S のマーク）から，ゴール地点（G のマーク）までの最短距離を示すプログラムを考えることを指示する。



※導入で実施した迷路シミュレーションを再度実行して考えるようにする。

	■アルゴリズムを図示する。 ※可能であれば，フローチャートの作成やどういったコードを用いるかの予想を行う。	・アルゴリズムについては，文字で記述するのではなく，できるだけフローチャート等で図示させる。 ※新たなコードを用いることはなく，既修事項で実施できることを伝える。	
展開4	■「迷路」の解法についてのプログラムを確認する。 ・どのような構造になっているか（特に，関数の使われ方），そしてそれぞれの関数はどのような機能を持っているかを考える。 ・事前に考案していたアルゴリズムとコードを比較する。 迷路の解法プログラム例は《別紙生徒テキスト 12.8 以降》を参照してください。	■「迷路」の解法のプログラムを提示し，どのような構造になっているかを考えさせる。 ・新たに２つの関数の定義をおこなっていることと，それぞれがどのような働きをしているかを理解させる。 ・生徒らが考案したアルゴリズムと実際のプログラムコードとを比較させる。	
展開5	■プログラムの改良点について話し合う。 ※もしくは，このような迷路生成プログラムが実際に用いられているゲームを探し，アルゴリズムを検討するなど，今まで気づかなかったことを探究する。	■プログラムの改良点や付加したい機能などについて話し合う。 ※もしくは，「このような迷路作成プログラムが実際に用いられているゲームを探し，そのアルゴリズムを検討させる」など，今まで気づかなかったことを探究させても構わない。	※これまでの授業例では，プログラムのブラッシュアップについて検討してきたが，今回のプログラムの改良は非常に難解となるため，この【展開（5）】は実施なくても構わない
終末	■検討したアルゴリズムや実際のプログラムについて気づいたことを発表・共有する。 ■本時で新たに学んだプログラミングの知識や技能を振り返る。	■本時で学んだアルゴリズムや実際のプログラムのコードなどを分析して，気づいたことを発表させる。 ※【展開（5）】で調べた結果を交えても構わない。	※Google Colaboratory の「＋テキスト」を押して，本時で学んだことや振り返りを記入させておく。 ※クラウドサービスなどを利用して，各生徒がアレンジしたプログラムが共有できると良い。

授業コード対応表

情報Ⅰ　コンピュータとプログラミング

授業タイトル			当冊子	生徒テキスト	授業コード
基礎編	（1）	テキストプログラミングの基本操作	P6-10	1. Google Colaboratory ～ 3. 変数	高・情(1)
	（2）	多様な「条件分岐」の方法について学ぶ	P11-15	4. 条件分岐	高・情(2)
	（3）	「配列変数」の使い方について学	P16-19	5. 配列	高・情(3)
	（4）	ループ（for と while）の使い方	P20-22	6. ループ	高・情(4)
	（5）	関数の定義とその活用	P23-25	7. 関数	高・情(5)
応用編	（6）	プログラミングでのグラフィックス	P26-29	8. グラフィックス	高・情(6)
	（7）	幾何学模様を描こう	P30-33	9．幾何学模様	高・情(7)
	（8）	デジタル単語帳	P34-38	10. デジタル単語帳	高・情(8)
発展編	（9）	Hit and Blow：数字当て	P39-42	11. Hit and Blow	高・情(9)
	(10)	「迷路」の作成と解法	P43-47	12. 迷路を自動生成するには	高・情(10)

※「プログラミング教育支援員」の訪問が決定した際には、各単元の進行状況や支援依頼内容についての事前連絡をおこなうようにして下さい。特に、《生徒テキスト》の使用の際には、支援を依頼する該当ページをお伝え下さい。