情報科　学習指導案

単元名	第２編 コミュニケーションと情報デザイン
単元の目標 （単元で育成する資質・能力）	① 知識及び技能 ・メディアの特性とコミュニケーション手段の特徴について，その変遷も踏まえて科学的に理解すること。 ・情報デザインが人や社会に果たしている役割を理解すること。 ・効果的なコミュニケーションを行うための情報デザインの考え方や方法を理解し，表現する技能を身に付けること。 ② 思考力・判断力・表現力等 ・メディアとコミュニケーション手段の関係を科学的に捉え，それらを目的や状況に応じて適切に選択すること。 ・コミュニケーションの目的を明確にして，適切かつ効果的な情報デザインを考えること。 ・効果的なコミュニケーションを行うための情報デザインの考え方や方法に基づいて表現し，評価し改善すること。

具体的な評価規準

知識・技能	思考力・判断力・表現力	主体的に学習に取り組む態度
・メディアの特性とコミュニケーション手段の特徴について，その変遷も踏まえて科学的に理解している。 ・情報デザインが人や社会に果たしている役割を理解している。 ・効果的なコミュニケーションを行うための情報デザインの考え方や方法を理解し，表現する技能を身に付けている。	・メディアとコミュニケーション手段の関係を科学的に捉え，それらを目的や状況に応じて適切に選択している。 ・コミュニケーションの目的を明確にして，適切かつ効果的な情報デザインを考察している。 ・効果的なコミュニケーションを行うための情報デザインの考え方や方法に基づいて表現し，評価し改善している。	・情報と情報技術を活用して効果的なコミュニケーションを行おうとしている。 ・情報デザインを効果的に活用し，情報社会に主体的に参画しようとしている。

単元計画

次	時	評価規準と評価方法	学習活動
一	1 2 3 4 5 6 7	【評価規準】 デジタル化する方法について理解している。 【評価方法】 記述の観察	① アナログとデジタルの違い，デジタル情報の特徴やメリットを理解させる。 ② ビットの概念を理解させ，それを表現するための 2 進法や 16 進法，さらに 2 進法を用いた数のデジタル表現，文字のデジタル表現を理解させる。 ③ 音，画像，動画のデジタル化の原理を理解させる。 ④ デジタル情報ならではのデータ圧縮の原理と具体例について理解させる。

二	8 9	【評価規準】 情報発信の目的に応じた適切なメディアを選択すべきか理解している。 【評価方法】 行動の観察	⑤ 古代からの技術的な進歩を概観し，コミュニケーション手段の発展について理解させる。 ⑥ 情報を発信するときのメディアの性質と特徴を理解させ，目的や状況に応じて適切なメディアの選択ができる力を身に付けさせる。
三	10 11 12 13	【評価規準】 伝えたい情報をわかりやすく表現するための知識を身に付け，表現することができる。 【評価方法】 記述の分析	⑦ 情報デザインが人や社会に果たしている役割を理解させるとともに，情報を抽象化・構造化・可視化する方法や表現を工夫する技能を身に付けさせる。 ⑧ ユーザビリティやアクセシビリティ，ユニバーサルデザイン等について，身近な具体例を挙げながら理解させる。
四	14 15 16 17	【評価規準】 目的や対象者を踏まえた，効果的なプレゼンテーションを行うことができる。 【評価方法】 行動の分析	⑨ プレゼンテーションの流れと注意点を理解させるとともに，効果的なコミュニケーションのために情報デザインの考え方や方法を理解させる。 ⑩ プレゼンテーションソフトウェアの使い方や表現する技能を身に付けさせるとともに，その制作物を評価し改善する活動を行わせる。

宮崎県立妻高等学校　情報科（科目：情報Ⅰ）　指導教諭　河野正臣

１．日時　令和４年9月２８日（水）２限目

２．場所　図書館棟１階コンピュータ室

３．対象クラス　１年３組　２６名

４．小単元及び本時の位置
第３章　情報デザイン(計４時間)
27　情報を表現する方法(0.5 時間)
28　効果的な情報デザイン(0.5 時間)
実習(本時１/３時間)

５．本時の目標
コミュニケーションの目的や受け手の状況に応じて，情報をどのように活用・表現すればよいか考え，その結果を適切に表現している。

６．展開

段階	具体的な内容・活動	指導上の留意・配慮事項	評価内容・評価方法
導入8分	１　タイピング練習を行う。 ２　前時の復習	・前時のポイントを踏まえた実習を行うことを伝える。	・タイピングの状況を目視で確認する。
展開32分	３　本時の内容 ・特別教室などの特徴ある施設のピクトグラムを作成する。	・世の中にあるピクトグラムを紹介し,制作物のイメージを持たせる。 ・評価の観点についても確認する。 ・アプリの使い方を簡単に指導する。 ・周りと相談することを推奨する。	・机間指導にて，評価ルーブリックがC評価となりそうな生徒へアドバイスを行う。
まとめ5分	４　本時のまとめ ・本時のまとめとして「振り返りシート」への記入をする。 ５　次時の予告	・振り返りシートへの記入することで，本時の学習内容を確認させる。 ・(再)提出ボタンを忘れずにクリックさせる。 ・次時の予告を聞く。	

実　習

ピクトグラムの作成

教室や施設を表すピクトグラムを作成してみよう

情報Ⅰ Next

第２編

ピクトグラムってなに？

ピクトグラムってなに？

伝えたい情報の本質を抜き出し，
それ以外の情報を単純化し，
わかりやすく表現したもの。

実習 教室や施設を表すピクトグラムを作成してみよう

・今日を含めて３時間で完成させる。
・評価は，評価ルーブリックにしたがって行う。

	評価項目	S	A	B	C
企画内容	独創性	Aに加え，これまでにない斬新で独創的なアイデアである。	既存のものを興味深い切り口で捉えた新しいアイデアである。	既存の製品等から着想を得て作られたアイデアである。	既存の製品や制度に使用されているアイデアである。
	具体性	Aに加え，詳細な設定が問題の解決に効果的な役割を果たしている。	Bに加え，使用の用途や条件など詳細な設定まで考えている。	問題と解決策が対応している。	アイデアが問題の解決になっていない。
デザイン	レイアウト	配置や余白の工夫など，聞き手が内容を理解しやすいように細かい配慮をしている。	文章を最小限に抑え，図や表，グラフ，イラストで説明している。	図や表，グラフを加えて聞き手が内容を理解しやすいよう工夫している。	スライドが文章だけで構成されている。
	見やすさ	Aに加え，聞き手の注意を引く工夫をしている。	強調の色やアニメーションに規則性を持たせるなど，全体に統一感を持たせている。	強調部分の色を変えたり，文字サイズを大きくしたりするなどの見やすい工夫をしている。	関係のないイラストや過度なアニメーション等が使用されている。
発表		Aに加え，社会問題の解決に有力なアイデア	根拠となる事実や事例について調査・分析さ	聞き手が問題から解決策まで内容を理解でき	聞き手が内容を理解できない

実習　教室や施設を表すピクトグラムを作成してみよう

手順1　作成したい教室や施設の特徴を確認しよう。

・その教室や施設の特徴や目的，伝えたい情報の本質を確認する。

・それ以外の情報を単純化する。

手順2　ピクトグラムを作成する。

・Google classroomにて配布している課題「ピクトグラム」を開く。

・簡単に使い方を説明するので参考に。

・となり近所と相談，大いに結構！！
友達の意見も参考にしよう！

LINK

26 情報を表現する方法

Keyword

ピクトグラム 図1
場所や利用する人の特性についての情報を抽象化した記号。ピクトグラムは，特定の言語によらないことから，公共施設などで広く活用されている。

a 情報デザイン

情報を発信する際，受け手に適切に伝わらないと，かえって混乱を招いてしまうことがある。情報の受け手の知識や状況などに配慮した上で情報の表現方法を検討する手法を，**情報デザイン**という。

情報の抽象化・構造化

図1 情報の抽象化

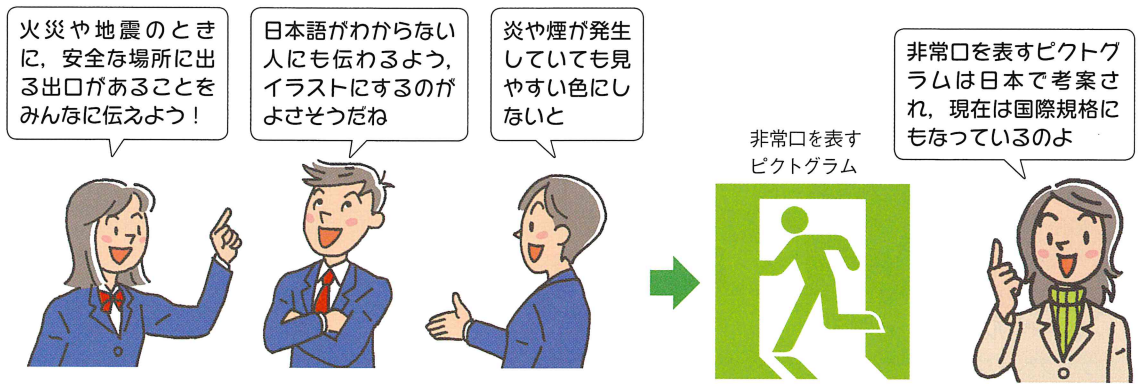


図2 情報の構造化と工夫の例

△△年×月×日

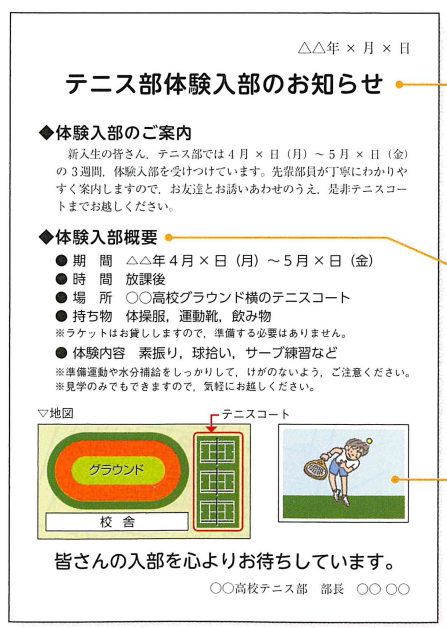
テニス部体験入部のお知らせ

新入生の皆さん，テニス部では４月×日（月）～５月×日（金）の３週間，体験入部を受けつけています。先輩部員が丁寧にわかりやすく案内しますので，お友達とお誘いあわせのうえ，是非テニスコートまでお越しください。

◆体験入部概要
●期間　△△年４月×日（月）～５月×日（金）●時間　放課後●場所　○○高校グラウンド横のテニスコート●持ち物　体操服，運動靴，飲み物※ラケットはお貸ししますので，準備する必要はありません。●体験内容　素振り，球拾い，サーブ練習など　※準備運動や水分補給をしっかりして，けがのないよう，ご注意ください。　※見学のみでもできますので，気軽にお越しください。

皆さんの入部を心よりお待ちしています。

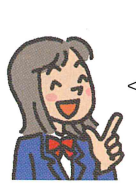
○○高校テニス部　部長　○○ ○○



Point! 文字の配置を工夫しよう！
➡ タイトルは中央揃え，見出しは左揃え，本文は左から数文字分右に寄せる。

Point! 文字の大きさやフォントを考えよう！
➡ タイトルや見出しは大きくしてめだたせる。

Point! 写真やイラストを追加しよう！
➡ 写真，イラスト，図，表などを載せると，より見やすくなる。



ワープロなら，いろいろな文字の大きさや書体，配置を試すことができるから，わかりやすい書類がつくれるね！

b 表現方法の工夫

情報を表現するときは，次の(1)～(3)を意識する。

(1) **情報の抽象化**　伝えたい情報の本質を抜きだし，それ以外の情報を単純化すると，情報をすばやく伝えることができる。

(2) **情報の構造化**　たとえば，文書でメッセージを表現する場合，情報を整理し，整理した情報どうしをまとめたり強調したりすると，構造がわかりやすくなる。

(3) **情報の可視化**　数値を含む情報は，表やグラフを用いて視覚的に表現すると，傾向が分析しやすくなる。 図2 図3

Keyword

ワープロソフトウェア，表計算ソフトウェア
文書を作成するためのソフトウェアを**ワープロソフトウェア**という。➡p.156
また，表やグラフの作成，集計や統計などの計算ができるソフトウェアを**表計算ソフトウェア**という。➡p.158

情報の可視化

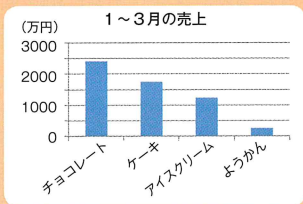
図3 情報の可視化と工夫の例

お菓子の売上　（単位：万円）

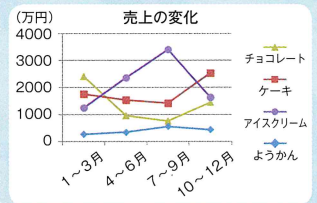
	1～3月	4～6月	7～9月	10～12月
チョコレート	2413	958	752	1456
ケーキ	1756	1531	1408	2529
アイスクリーム	1244	2355	3413	1632
ようかん	268	344	556	440
合計	5681	5188	6129	6057

適切なグラフの選択
表現したい内容にあわせて，適切なグラフを選択する。

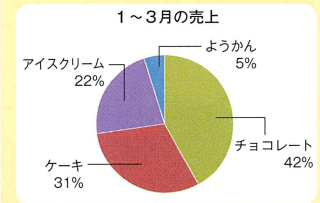
1～3月の売上を比べたい場合 ➡ **棒グラフ**
個別のデータの量を比べる



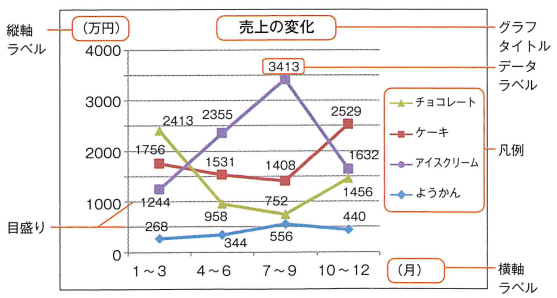
売上の変化を見たい場合 ➡ **折れ線グラフ**
データの時間的な変化を見る



1～3月の売上の割合を見たい場合 ➡ **円グラフ**
集めたデータがどういった割合であるかを比べる

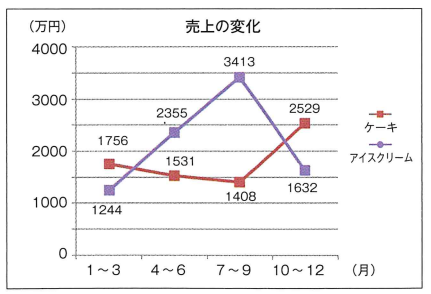


ラベルや凡例の表示
それぞれ何を表しているのか，わかるようにする。



凡例…データが何を表しているかということ。

適切なデータの量
関係するデータだけを選択して，簡潔なグラフにする。



2つの商品だけを比較したい場合

話しあってみよう　これまでに見たことのあるピクトグラムで印象に残っているものをあげ，正確な情報伝達のためにどのような工夫がされているのか話しあってみよう。

- 0928学習指導案 7 -

▶▶LINK

27 効果的な情報デザイン

a デザインの工夫

(1) **アフォーダンス** affordance　私たちはたくさんのものに囲まれて生活している。それぞれのものには形や色などの特徴があり，それによって私たちの行動は自然と影響されるため，はじめて目にしたものであっても，たいていはうまく扱うことができる。このような，ものと人の動作の関係性のことを**アフォーダンス** affordance という[1]。

[1] 情報デザインを考えるうえでは，望ましくない行動が可能となってしまうようなアフォーダンスをとり除く必要がある 図1。

(2) **ユーザインタフェース** user interface　人とコンピュータの間で情報の受け渡しを行うもの。

(3) **ユーザビリティとアクセシビリティ**　機器やサービスの利用のしやすさを**ユーザビリティ** usability という。ユーザビリティは利用する人によって異なることがある。それに対し，幅広い人々にとっての利用しやすさを表すのが，**アクセシビリティ** accessibility である。 図2 図3

(4) **バリアフリー** barrier free　道路や建物内の段差をなくして移動しやすくしたり，さまざまな操作に対応した機器をつくったりして，高齢の人や身体の不自由な人が生活する上での支障をとり除く工夫。 図4

(5) **ユニバーサルデザイン (UD)** Universal Design　年齢や身長，右利きと左利き，身体の障害の有無などによらず，すべての人が不自由なく使えることをあらかじめ想定して工夫されたデザイン。 図5 図6

キャリア▶▶

Note

インダストリアルデザイナー
エアコンや洗濯機など，工業製品のデザインを考えるデザイナー。おしゃれな形をデザインするだけではなく，製品の使いやすさや安全性をふまえて設計する。

アフォーダンスとユーザインタフェース

図1 アフォーダンス

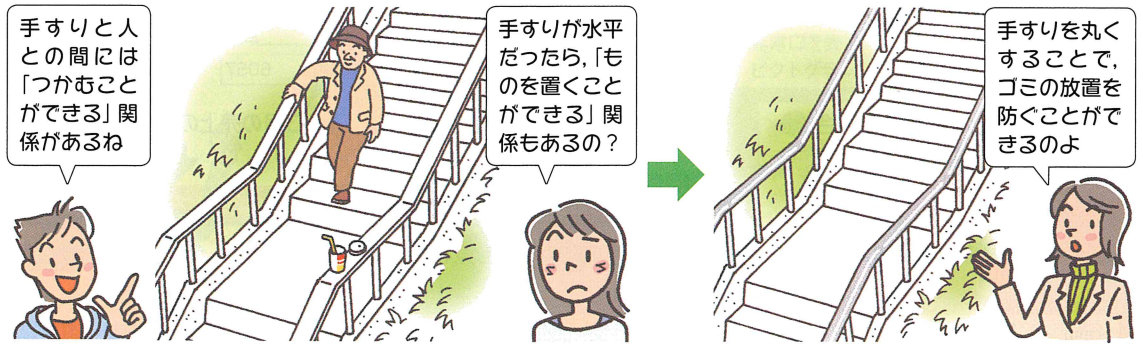


図2 ユーザインタフェースの例

◆CUI (キャラクタユーザインタフェース)
文字だけで情報を表示し，キーボードの文字入力で操作する方式。

◆GUI (グラフィカルユーザインタフェース)
文字やアイコン (絵柄) で情報を表示し，マウスやタッチパネルなどで操作する方式。

◆NUI (ナチュラルユーザインタフェース)
発話や身振り手振りによって操作する方式。人工知能技術の発展によって誕生した新しいユーザインタフェース。

デザインの工夫

図3 ユーザビリティへの配慮の例

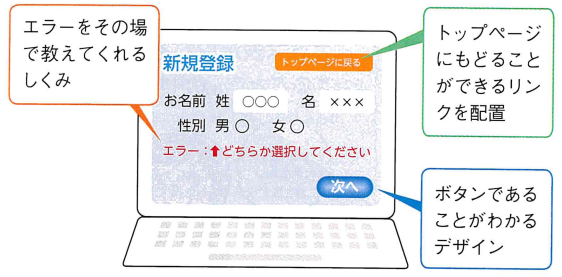


図4 アクセシビリティへの配慮の例

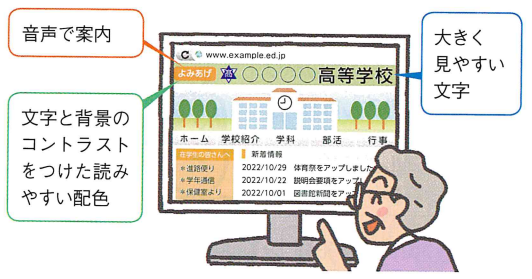


図5 バリアフリーの例

図6 ユニバーサルデザインの例

話しあってみよう　身近にあるものの中で，ユニバーサルデザインの考え方で設計されている例をあげ，どういったデザインがあらゆる人にとって利用しやすくなっているのか話しあってみよう。

- 0928学習指導案 8 -

情報科　学習指導案

単元名	第３編 コンピュータとプログラミング
単元の目標 （単元で育成する資質・能力）	① 知識及び技能 ・コンピュータや外部装置の仕組みや特徴，コンピュータでの情報の内部表現と計算に関する限界について理解すること。 ・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。 ・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。 ② 思考力・判断力・表現力等 ・コンピュータで扱われる情報の特徴とコンピュータの能力との関係について考察すること。 ・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。 ・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。

具体的な評価規準

知識・技能	思考力・判断力・表現力	主体的に学習に取り組む態度
・コンピュータや外部装置の仕組みや特徴，コンピュータでの情報の内部表現と計算に関する限界について理解している。 ・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，技能を身に付けている。 ・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している。	・コンピュータで扱われる情報の特徴とコンピュータの能力との関係について考察している。 ・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している。 ・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察している。	・問題の発見・解決にコンピュータを積極的に活用しようとしている。 ・問題解決の結果を振りかえり改善しようとしている。 ・身近な問題を解決するプログラムやシミュレーションの結果をもとに，試行錯誤しながら評価し改善しようとすることなどを通して，情報社会に主体的に参画しようとしている。

単元計画

次	時	評価規準と評価方法	学習活動
一	1 2 3	【評価規準】 コンピュータの基本的な仕組みや働き，内部における数の扱いについて，科学的に理解しているか。 【評価方法】 記述の観察	① コンピュータの基本的な仕組みと働き，CPU，メモリ，補助記憶装置，入出力装置，OS，ファイルの基礎について理解させる。 ② コンピュータの内部における数の表現方法と計算に関する限界について理解させる。

			③ コンピュータで扱われる情報の特徴やコンピュータの能力との関係について考えさせる。
二	4 5 6 7 8 9	【評価規準】 プログラムの作成を通してアルゴリズムやプログラミングの基本を理解している。 【評価方法】 行動の観察	④ 問題の解法をアルゴリズムを用いて表現する方法を身に付けさせる。 ⑤ プログラミングの基本的な考え方とコンピュータを活用する方法を理解させる。 ⑥ プログラミングの技能を身に付けさせるとともにプログラミングを評価し改善する活動を行わせる。
三	10 11 12 13	【評価規準】 シミュレーションの知識を身に付け，問題解決に活用する方法を理解している。 【評価方法】 行動の観察	⑦ モデル化やシミュレーションの考え方・手順を理解させ，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を理解させる。 ⑧ 目的に応じたモデル化やシミュレーションを行わせ，その結果から問題の解決方法を考えさせる。

宮崎県立妻高等学校　情報科（科目：情報Ⅰ）　指導教諭　河野正臣

1．日時　令和４年１１月１１日（金）３限目

2．場所　図書館棟１階コンピュータ室

3．対象クラス　１年２組　３３名

4．小単元及び本時の位置
第３編　コンピュータとプログラミング
第３章　プログラミング(計６時間)
33　アルゴリズム(１時間)
34～37　プログラミング(1)～(4)　(本時２/５時間)

5．本時の目標
基本的なプログラムの入力と実行及び修正を実際に行い，アルゴリズムの基本を理解させる。

6．展開

段階	具体的な内容・活動	指導上の留意・配慮事項	評価内容・評価方法
導入８分	1　タイピング練習を行う。 2　前時の復習（確認）	・進捗状況を確認し，進みの遅い生徒を把握する。	・タイピングの状況を目視で確認する。
展開３２分	3　本時の内容 ・前時に続き，プログラミング教材に沿って各自進める。	・半角英数（クォーテンションマーク内のみ全角 OK）で入力することを確認する。 ・エラーが出た場合は，デバッグをまず自分で行う。３分間考えて解決しなければ，周りに助けを求める。 ・周りと相談することを推奨する。	・机間指導にて，評価ルーブリックがＣ評価となりそうな生徒へアドバイスを行う。
まとめ５分	4　本時のまとめ ・本時のまとめとして「振り返りシート」への記入をする。 5　次時の予告	・振り返りシートへの記入することで，進捗状況を確認させる。 ・(再)提出ボタンを忘れずにクリックさせる。 ・次時の予告を聞く。	

33 アルゴリズム

a アルゴリズム

ある問題や課題を解決するための計算手順や処理手順を**アルゴリズム**(algorithm)という。アルゴリズムは，日常生活のさまざまな場面でも使われており，たとえば，料理の手順が書かれたレシピは，アルゴリズムの1つといえる[1]。図1

問題にはいろいろな解き方があり，アルゴリズムにも速く解ける手順や時間のかかる手順などがある。たとえば，英和辞書で「cat」という単語をひく場合，「辞書の最初のページから1つずつ順番にさがす」手順よりも，「見出しから『C』の最初のページを開いた後，1つずつ順番にさがす」手順のほうが，効率よく単語をさがすことができる。このように，それぞれの問題にあわせた効率のよいアルゴリズム図2を考える必要がある。

❶ レシピどおりの分量・手順でつくれば，初めての料理でも上手につくることができる。

b フローチャート

アルゴリズムを表現するときは，図によって手順の流れをわかりやすくした表現方法がよく用いられる。

フローチャート（**流れ図**）とよばれる表現方法では，ひとつひとつの手順を「箱」で表し，「箱」と「箱」を「矢印」でつなぐことによって，手順の流れを視覚化している[2]。図3

キャリア▶▶

Note

システムエンジニア（SE）

依頼人からの要望を受けてコンピュータのシステムを設計する職業。他の人が見てもシステムの構造がわかるように，アルゴリズムを表現する能力が求められる。

❷ フローチャートと同じように手順を「箱」と「矢印」で表したアクティビティ図とよばれる表現方法もある。アクティビティ図は，複数の人の手順を並行して示すことができる図3。

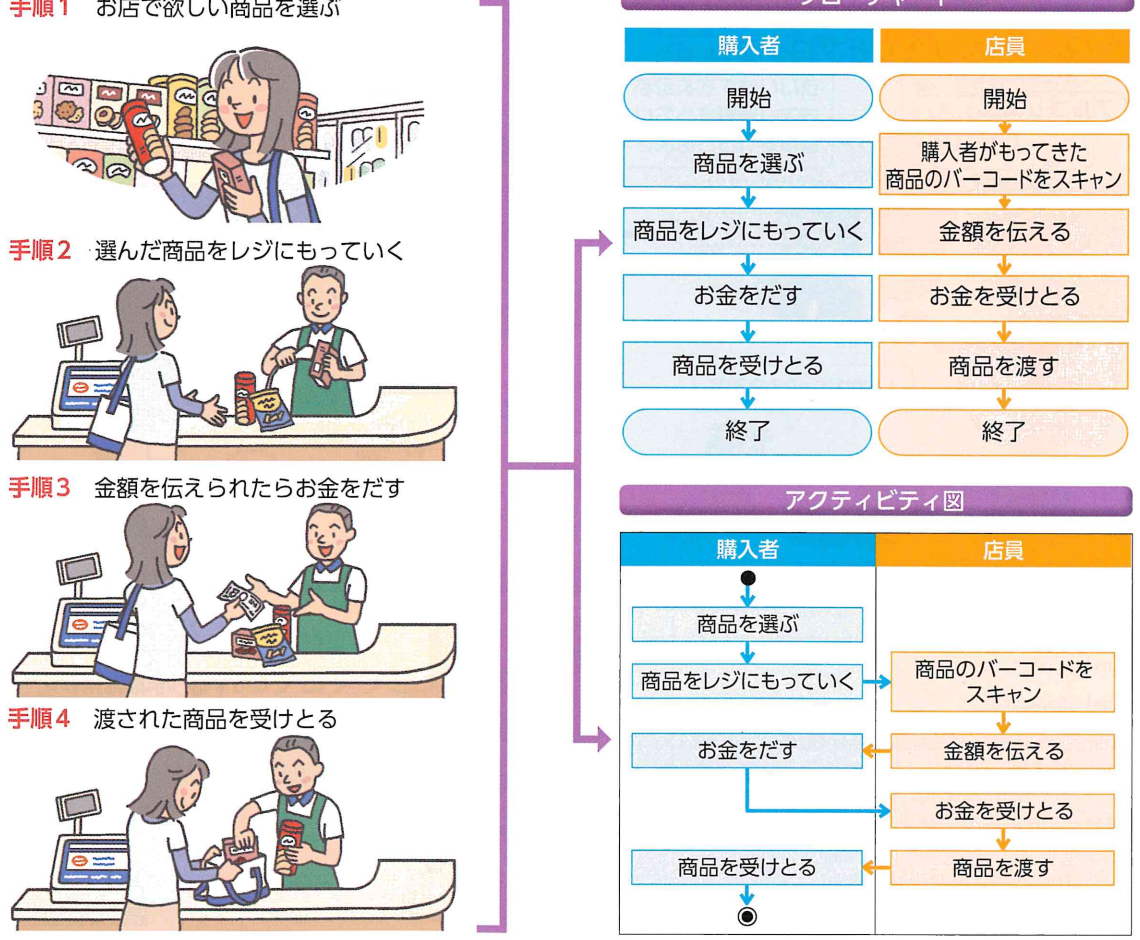
アルゴリズム

図1 アルゴリズムの例（料理のレシピ）

図2 辞書をひくときのアルゴリズム

アルゴリズムの表現

図3 商品を購入するときのアルゴリズム



話しあってみよう 自動販売機で商品を購入するときのアルゴリズムについて話しあってみよう。また，話しあいの結果のアルゴリズムをフローチャートで表してみよう。

LINK

34 プログラミング(1)
－プログラミング言語－

a プログラムとプログラミング言語

(1) **プログラミング**　コンピュータが処理できるようにアルゴリズムを記述すること。

(2) **プログラミング言語**　プログラミングのために考えられた専用の言語。文字を使って書くものやブロックを組みあわせて書くものがある。

(3) **プログラム**[図2][1]　プログラミング言語によって書かれたアルゴリズム。プログラムでは，アルゴリズムにあいまいな部分がないようにしなければならない[2]。

❶ あらかじめ (pro-)，ものごとの予定や計画を書き記したもの (-gram) という意味がある。

❷ コンピュータは，人とちがって正確な命令しか理解できない。そのため，あいまいな部分や誤字があると，プログラムは正しく動作しない[図3]。

b プログラミング言語の選択

(1) **プログラムの作成**　プログラムは通常，**エディタ**というソフトウェアで作成する。エディタで作成されたプログラムは，データとしてコンピュータに保存される。

(2) **プログラムの実行**　プログラムを実行するためには，書かれた文字や数字をコンピュータが理解できる形式[3]に変換する必要がある。この変換方法には，プログラミング言語によって，1行1行読みこみながら順番に変換・実行を行うものや，プログラム全体をまとめて変換するものなどがある。

(3) **プログラミング言語の選択**　プログラミング言語には，さまざまな種類があり，言語によって変換・実行のしかたや，得意とする処理・計算が異なる。そのため，どのような問題をどのような処理・計算で解くか[表1]を考え，適切なプログラミング言語を選択することが必要である。

❸ コンピュータは，0と1の2進法で表した形式しか理解できない。

アルゴリズムとプログラム

図1 アルゴリズムとプログラム

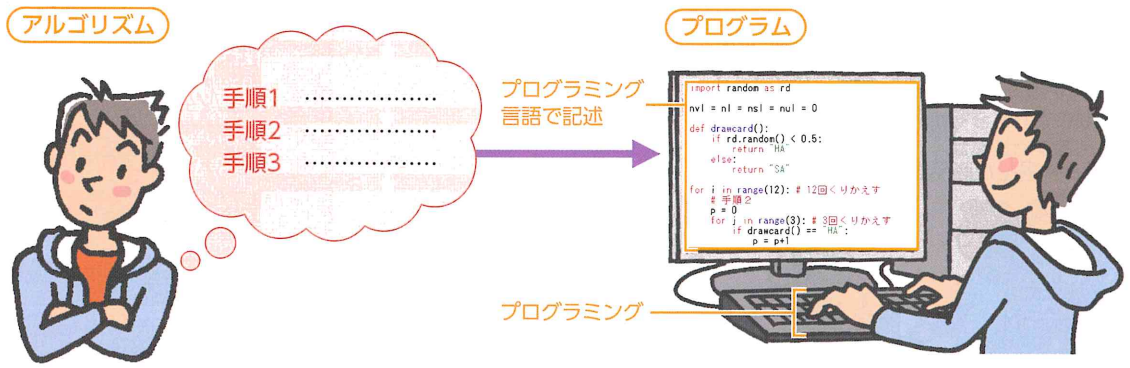


図2 プログラミング言語

文字を使って書く言語

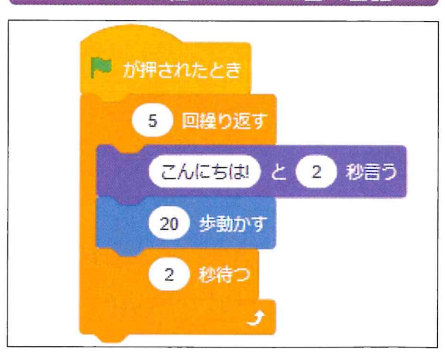
```
import random as rd

nvl = nl = nsl = nul = 0

def drawcard():
    if rd.random() < 0.5:
        return "HA"
    else:
        return "SA"

for i in range(12): # 12回くりかえす
    # 手順2
    p = 0
    for j in range(3): # 3回くりかえす
        if drawcard() == "HA":
            p = p+1
```

ブロックを組みあわせて書く言語



人の話す言語との違い

図3 人間とコンピュータのちがい

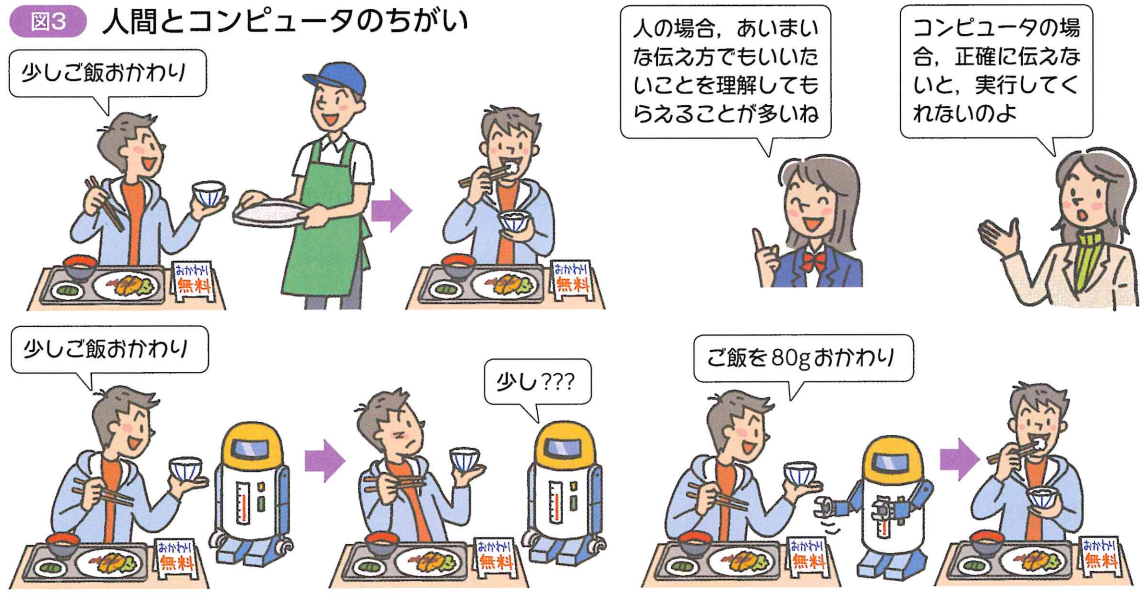


表1 よく使われるプログラミング言語

プログラミング言語	特徴	プログラミング言語	特徴
表計算マクロ言語	表計算ソフトウェアで処理を行うための言語	Java（ジャバ）	汎用性が高く，さまざまなアプリケーション開発に使われる言語
JavaScript（ジャバスクリプト）	おもにウェブページ上で動作する言語	C（シー）	OSなどの開発にも使われる，汎用性の高い言語
Python（パイソン）	人工知能などさまざまな分野で使われ，簡潔で読みやすい言語	Scratch（スクラッチ）	入門向けによく使われる，ブロックを組みあわせて書く言語。

話しあってみよう　小学校や中学校でこれまでに体験したプログラミングについて，どのようなことをするプログラムであったのか，話しあってみよう。

- 1111学習指導案 5 -

LINK

35 プログラミング(2)
－プログラミングの方法－

❶ ソースコードは，通常，上から順に実行される。

a ソースコード

プログラムは，各プログラミング言語のルールに従って，文字や数値，記号などの命令を1行ずつ上から順に書く。このように書かれた命令を**ソースコード**（**コード**）という[1]。
source code

b 変数

三角形の面積を計算するプログラムの場合，具体的な数値ではなく，公式（アルゴリズム）を書いておく。

三角形の面積 ＝ 底辺 × 高さ ÷ 2

この場合，「底辺」や「高さ」にあてはめる数値をかえるだけで，別の三角形の面積が計算できる。

三角形の面積の公式の「底辺」や「高さ」のように，いろいろな数値や文字をあてはめることができる「箱」のようなものを**変数**（へんすう）といい[2]，変数に数値や文字をあてはめることを**代入**（だいにゅう）という。図2

プログラムでは，代入して計算した結果をまた別の変数に代入するという手順をくりかえして，計算を実行する。

c 演算子

変数に数値や文字を代入したり計算を行ったりすることを演算といい，演算に用いる「＝」や「＋」などの記号を**演算子**（えんざんし）という。表1

❷ 変数に対し，常に同じ数値や文字をもつものを，定数という。

⚠ Attention

イコール（＝）の意味

数学では，「＝」という記号を「等号」とよび，たとえば「x=2」は「xと2は等しい」ことを意味する。しかし，プログラミング言語では，「＝」を「代入演算子」とよび，たとえば「x=2」は「xに2を代入する」ことを意味する場合が多い。プログラミング言語で「等しい」という意味を表す場合は，比較演算子（➡ p.106）を使う。

文字を表示するプログラム

図1 文字を表示するプログラムの例

表計算マクロ

```
1 Sub 文字の表示 ()
2     MsgBox "hello"
3 End Sub
```

JavaScript

```
1 alert("hello");
```

Python

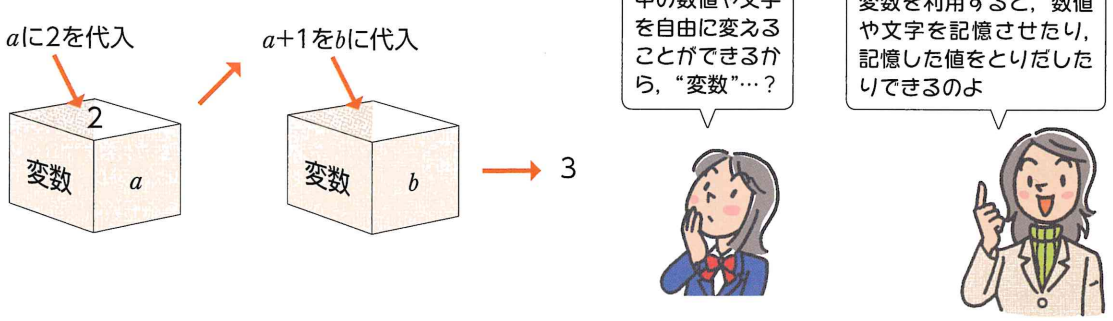
```
1 print("hello")
```

・表計算マクロでは，プログラムの開始と終了を示す命令が必要であり，「Sub」の後にプログラムの名前をつける（　部分）。

▶▶▶ TRY

❶ 上のプログラムを入力し，実行してみよう。
❷ "hello"を自分の好きな文字や文章に変更し，結果がかわることを確かめよう。

図2 変数と代入



計算を行うプログラム

表1 演算子

		表計算マクロ	JavaScript	Python
代入演算子		＝	＝	＝
算術演算子	足し算	＋	＋	＋
	引き算	－	－	－
	かけ算	*	*	*
	わり算	/	/	/
	累乗	^	＊＊	＊＊

図3 三角形の面積を計算するプログラムの例

アルゴリズム

手順1 「底辺」の変数に数値を代入
手順2 「高さ」の変数に数値を代入
手順3 「底辺」×「高さ」÷2の結果を「面積」の変数に代入
手順4 「面積」を表示する

表計算マクロ

```
1 Sub 三角形の面積 ()
2     Dim area, base, height
3     base = 10          …手順1
4     height = 7         …手順2
5     area = base * height / 2
6     MsgBox area        …手順4
7 End Sub                手順3
```

JavaScript

```
1 let area, base, height;
2 base = 10;             …手順1
3 height = 7;            …手順2
4 area = base * height / 2;
5 alert(area);           …手順4
                         手順3
```

Python

```
1 base = 10              …手順1
2 height = 7             …手順2
3 area = base * height / 2
4 print(area)            …手順4
                         手順3
```

・表計算マクロとJavaScriptでは，変数を使う前に，変数を設定する必要がある。
この命令は「変数の宣言」とよばれる。

▶▶▶ TRY

❶ 上のプログラムを入力し，実行してみよう。
❷ 変数baseとheightに代入する数値を変更し，計算結果がかわることを確かめよう。

話しあってみよう　ソースコードを無償で公開し，誰でも改良や再配布ができるようにすることをオープンソースという。オープンソースにするとどのような利点があるのか話しあってみよう。

LINK

36 プログラミング(3)
－条件分岐式とくりかえし命令－

❶ 条件式では，比較演算子とよばれる演算子が用いられる 表1。

a 条件式

条件にあっているかどうかを計算し，あっている場合は**真**，あっていない場合は**偽**という答えをだす式を**条件式**という❶。

b 条件分岐式

「ある数字が別の数字より大きい場合は引き算を行い，そうでない場合は，足し算を行うプログラム」のように，条件にあっている場合と，あっていない場合とで，実行する命令をかえる式(図1)を**条件分岐式**という。

条件分岐式では，条件式の計算結果が真の場合と偽の場合とで，それぞれの命令が実行されるように記述する。

c くりかえし命令

「『5』という数字が表示されるまで，『1』から1ずつ足した数字の表示をくりかえすプログラム」のように，条件を満たすまで何回もくりかえす処理を実行する命令(図2)を，**くりかえし命令**という。

くりかえし命令でのくりかえしが終了する条件は，条件式によって記述する。

⚠ Attention

Pythonのインデント

行のはじめに挿入する字下げのことをインデントという。プログラミングでは，ソースコードを見やすくするためにインデントをいれる場合が多いが，Pythonのように適切な位置にインデントをいれないとエラーとなるプログラミング言語もある。

条件分岐式

表1 比較演算子

	表計算マクロ	JavaScript	Python
より大きい	>	>	>
より小さい	<	<	<
以上	>=	>=	>=
以下	<=	<=	<=
等しい	=	==	==
等しくない	<>	!=	!=

図1 変数の大小を比較して計算式をかえるプログラムの例

アルゴリズム

手順 1　変数 a と変数 b を比較
　a の方が大きい場合(真の場合)
　⇒ 手順 2 を実行して手順 3 を省略
　それ以外の場合(偽の場合)
　⇒ 手順 2 を省略して手順 3 を実行
手順 2　a から b を引いた結果を変数 c に代入
手順 3　a と b を足した結果を変数 c に代入

表計算マクロ

```
1 If a > b Then        …手順 1
2     c = a - b        …手順 2
3 Else
4     c = a + b        …手順 3
5 End If
```

JavaScript

```
1 if( a > b ){         …手順 1
2     c = a - b;       …手順 2
3 } else {
4     c = a + b;       …手順 3
5 }
```

Python

```
1 if a > b :           …手順 1
2     c = a - b        …手順 2
3 else:
4     c = a + b        …手順 3
```

・この例では，条件分岐式の部分以外は省略している。
・条件式は，比較演算子を使って，変数を比較する(　　部分)。

▶▶▶ TRY

❶上のプログラムを入力し，実行してみよう。
❷手順2と手順3の式をかけ算やわり算に変更し，計算結果がかわることを確かめよう。

くりかえし命令

図2 1～5までの数字を表示するプログラムの例

アルゴリズム

手順 1　変数 i に 1 を代入する
手順 2　i と 5 を比較する
　i が 5 以下の場合(真の場合)
　⇒ 手順 3 を実行
　それ以外の場合(偽の場合)
　⇒ 終わり
手順 3　i に代入されている数字を表示
　i に 1 を足す⇒ 手順 2 にもどる

表計算マクロ

```
1 i = 1                …手順 1
2 Do While i <= 5      …手順 2
3     MsgBox i         …手順 3
4     i = i + 1        …手順 3
5 Loop
```

JavaScript

```
1 let i = 1;           …手順 1
2 while(i <= 5){       …手順 2
3     alert(i);        …手順 3
4     i = i + 1;       …手順 3
5 }
```

Python

```
1 i = 1                …手順 1
2 while i <= 5:        …手順 2
3     print(i)         …手順 3
4     i = i + 1        …手順 3
```

TOPIC

プログラムの基本的な構造

プログラムのアルゴリズムには，次の3つの基本的な構造があり，これらを組みあわせることで，複雑な処理を行うプログラムが作成されている。
・順次構造：上から順に命令を実行する構造。
・選択構造：条件によって実行する命令をかえる構造。
・くりかえし構造：処理をくりかえす構造。

・この例では，くりかえし以外のプログラムを省略している。

▶▶▶ TRY

❶上のプログラムを入力し，実行してみよう。
❷手順2と手順3を変更し，1～15までの奇数を表示するプログラムを作成してみよう。

話しあってみよう　「条件を満たすまで何かを繰り返す処理」について，日常生活の中でもあてはまるものはないだろうか。具体的な例を考え，話しあってみよう。

LINK

37 プログラミング(4)
－配列と関数，プログラムの修正－

❶ 曜日，クラスの名簿，テストの点数の一覧など。

❷ たとえば，「a[0]～a[9]の10個の数字が入った配列aの中から，3番目に入っている数字をとりだす」場合，『a[2]』と指定する。

❸ 複雑な問題をまとめて解くと，プログラムも複雑なものとなり，わかりにくい。

a 配列

同じ種類のデータ❶を複数含んだ変数のことを，**配列**(はいれつ)という。(図1)プログラムでは，変数に数値や文字を代入して計算するが，数値や文字の個数が多い場合は，配列を使うと処理しやすい❷。(図2)

b 関数

長いプログラムを書くときは，問題を分割し，それぞれの問題を解く小さなプログラムを作成する。そして，それらを組みあわせて問題全体を解く❸。この小さなプログラムを**関数**(かんすう)という。

たとえば，あらかじめ「三角形の面積を計算する関数」をつくっておけば，長いプログラムの中で関数を指定し，必要なときに何度でも計算させることができる。(図3)

c プログラムの修正

プログラムの不具合のことを**バグ**といい，バグを修正する作業を**デバッグ**という。

(1) **プログラムの書き方の誤りによるバグ**　入力ミスなどにより，プログラミング言語のルールにあわない書き方をしていた場合，コンピュータはエラーメッセージを表示する。このエラーメッセージを見ながらバグを修正していく。

(2) **計算の誤りによるバグ**　書かれている計算の内容が意図した計算とちがっていた場合，計算結果は表示され，エラーメッセージは表示されないため，バグに気づきにくい。そのため，「変数に○○を入力して計算したとき，××の結果が表示されるはず」というお手本を実行させて，答えあわせをすることで，バグを見つけ修正していく❹。

(3) **アルゴリズムがまちがっている**　アルゴリズム自体がまちがっている場合は，プログラムが正しくても，適切な結果は得られない。

Roots

バグ

英語の「虫(bug)」に由来する。プログラムを書いた人は，自分では不具合に気づきにくいことから，機械の中の見えないところに虫が入りこんで不具合を起こしたことにたとえて，バグとよばれるようになった。また，デバッグは「虫とり(debug)」に由来する。

❹ この過程をプログラムの**テスト**とよぶ。ふつう，プログラムを書くときは，テストとデバッグをくりかえしながらバグをなくしていく。

- 1111学習指導案8 -

配列

図1 配列

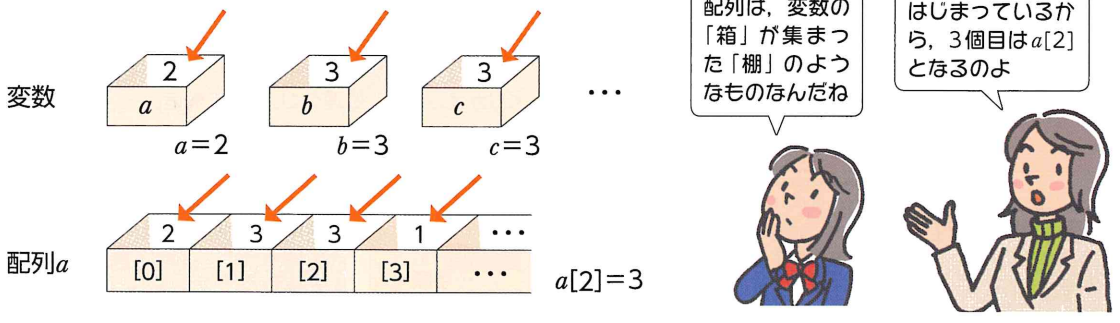


図2 配列に曜日のデータを入力する例

表計算マクロ
```
week = Array("月","火","水","木","金","土","日")
```

JavaScript
```
let week=["月","火","水","木","金","土","日"];
```

Python
```
week=["月","火","水","木","金","土","日"]
```

・たとえば，表計算マクロでは，week(1)とすることで，「火」を指定することができる。

関数

TRY

次のような10個の数字を配列に代入し，くりかえし命令を使って，1つずつ表示させてみよう。
2，5，8，9，30，45，55，102，183，209

図3 三角形の面積を計算する関数の例

表計算マクロ
```
1 Function triangle_area(base,height)
2       Dim area
3       area = base * height / 2
4       triangle_area = area
5 End Function
```

JavaScript
```
1 function triangle_area(base, height){
2       let area = base * height / 2;
3       return area;
4 }
```

Python
```
1 def triangle_area(base, height):
2       area = base * height / 2
3       return area
```

関数をつくっておけば，その関数に底辺と高さの数値をあたえて記述するだけで，三角形の面積を計算させることができるのよ

・この例では，関数の定義の部分以外は省略している。

話しあってみよう　同じ計算を行うプログラムでも，さまざまなアルゴリズムが考えられる。では，よいアルゴリズムとはどういったものだろうか。考えられる条件を話しあってみよう。

Google Colaboratoryを利用した Pythonの実習

2022.11.09 更新

宮崎県立妻高等学校
河野　正臣

1　Google Colaboratory（グーグル・コラボレイトリー）とは？

- Googleが機械学習の教育及び研究用に提供しているサービスの１つ
- ブラウザ上でPythonを記述，実行できる。
- 環境構築が不要
- GPUに料金なしでアクセス
- 簡単に共有
- Colab（コラボ）とも呼ばれます。

2　はじめに

このドキュメントは、Colab ノートブックという、コードを記述して実行できるものです。

たとえば次のコードセルには、値を計算して変数に保存し、結果を出力する短い Python スクリプトが記述されています。

```
hun_1day = 60 * 24  # 24時間を分に換算する。
print(hun_1day)
```

上記のセルのコードを実行するには、セルをクリックして選択し、コードの左側にある実行ボタン（黒丸に三角）をクリックするか、キーボード ショートカット「 Shift + Enter 」を使用します。実行して少し待つと結果が表示されます。

コードはセルをクリックしてそのまま編集できます。

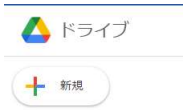
1つのセルで定義した変数は、あとで他のセルで使用できます。

```
byou_1day = hun_1day * 60  # 24時間を秒に換算する。
print(byou_1day)
```

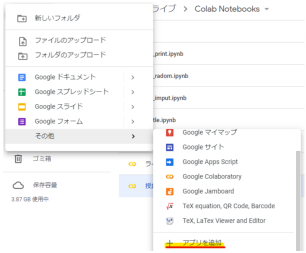
ちなみに作成したノートブックは、自動的にGoogle Driveに保存されます。

3　準備

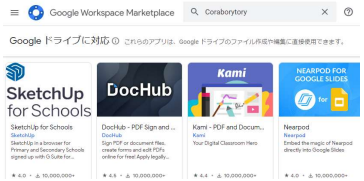
1. Google Driveの左上にある「＋新規」のボタンをクリックします。



2. 一番下にある「その他」→一番下にある「＋アプリを追加」をクリックします。



3. 出てきた画面の上側にある検索窓「apps内を検索」に**Colaboratory**と入力して検索し，出てきたColaboraytoryをクリックする（インストール）。

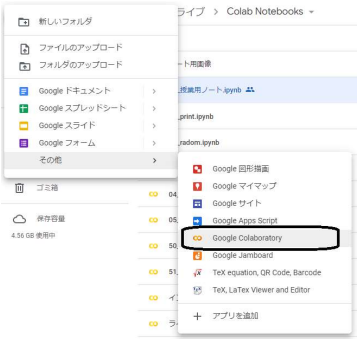


4　ファイルの新規作成

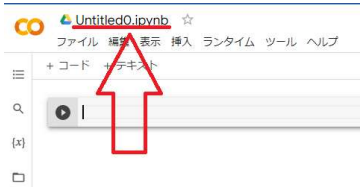
1. Google Driveの中にある「マイドライブ」→「Colab Notebooks」をクリックします（アプリを登録してすぐは作成されていないかもしれません。そのときは，2から進めてください）。

2. Google Driveの左上にある「＋新規」のボタンをクリックします。

3. 一番下にある「その他」→中程にある「**Google Colaboraytory**」をクリックすると，新しいノートブックが作成されます。



4. 作成したら，ファイルに名前を付けてください。左上にある「Untitled0.ipynb」の「Untitled0」の部分に新しい名前を付けます。



5　ファイル（ノートブック）を開く

ファイルは，Google Driveの中にある「マイドライブ」→「Colab Notebooks」内に保存されています。ダブルクリックして開きます。

しかし，iPad等の「ドライブ」アプリからは開くことができません。この場合は，ブラウザ（Safariなど）から開きましょう。

6　文字を表示する

1. 次の関数（命令）で表示することができます。

```
print(表示するもの)
```

「 print() 」の部分を関数，「 表示するもの 」の部分を引数といいます。 引数はクォーテンションマークで囲みます。ダブル(")でもシングル(')でもどちらでもよいですが，必ず対にします。作文を書くときに会話文を「 」で囲むのに近いイメージです。

(例)「おはよう」と挨拶した。

なお，**コードは半角英数で入力**します。**クォーテンションマークで囲まれた部分のみ、全角を使うことができます。**

```
print("おはよう")
print('Hello')
```

■練習01　「 こんにちは 」と表示させてください。
■練習02　近くの人と，画面に表示させて会話をしてみましょう。

2. プログラミングの世界では，数字と文字列を区別して扱います。
次の2つのコードを実行すると，どのようになるでしょう？

```
print("5 + 3")
print(5 + 3)
```

上のコードは「文字」として，下のコードは「数値」として，それぞれ認識されています。「数値」として計算させ，結果を表示させたいとき，引数はクォーテンションマークで囲みません。

では，次の2つのコードを実行すると，結果はどうなるでしょう？

```
print(5+3)
print(5 + 3)
```

実は，実行結果に違いはありません。
一般的には，読みやすさを優先して，半角スペースを入れて記述します。なお，半角

スペースを必ず入れなければならない場所もあります（後述）。
「＋」のような計算に必要な記号のことを算術演算子といいます。

演算子	読み	計算
+	プラス	足し算
-	マイナス	引き算
*	アスタリスク	かけ算
/	スラッシュ	割り算
%	パーセント	割り算の余り
//	スラッシュ スラッシュ	割り算の商
**	アスタリスク アスタリスク	べき乗（指数）

次のコードを実行すると，どのように表示されるでしょう？

```
print(5 + 2)
print(5 - 2)
print(5 * 2)
print(5 / 2)
print(5 % 2)
print(5 // 2)
print(5 ** 2)
```

■練習03　コードには数式を書いて「 13 」と表示させてください。
■練習04　練習03で使用していない算術演算子を使って数式を書いて「 13 」と表示させてください。

なお「＋」は，文字と文字をくっつける場合にも使用されます。

```
print("おはようございます。" + "今日もいい天気ですね。")
```


7　変数を使う

1. 次のコードを実行したら，どうなるでしょう？

```
hako = "たからもの"
print(hako)
```

このコードは，次のようになっています。

- [1行目]　「 hako 」と「 たからもの 」が等しいという意味ではありません。プログラミングの世界では，「 hako 」（左辺）に「 たからもの 」（右辺）を入れると考えます。
- [2行目]　「 hako 」を表示しますが，「 hako 」にはクォーテンションマークがないので，「 hako 」の中身である「 たからもの 」が表示されます。

2. 次のコードを実行したら，どうなるでしょう？

```
keisan = 6 + 7
print(keisan)
```

このコードは，次のようになっています。

- [1行目]　「 keisan 」には「 6+7 の計算結果 」つまり「 13 」が入っています。
- [2行目]　「 keisan 」を表示しますが，「 keisan 」にはクォーテンションマークがないので，「 keisan 」の中身である「 13 」が表示されます。

■練習05　「 box_1 」という変数を作成し，その変数に何か言葉を入れ，「 box_1 」を表示してください。
■練習06　「 box_2 」という変数を作成し，その変数に何か数式で表したものを入れ，「 box_2 」を表示してください。

3. ユーザーに文字を入力させ，表示させることもできます。

```
data = input("あなたの名前を入力してください。")
print("こんにちは，" + data + "さん。Pythonがんばっていきましょうね。")
```

このコードは，次のようになっています。

- [1行目]　`input()` 関数で，ユーザーに入力を求めます。入力された文字を `data` に入れます。
- [2行目]　`data` の部分に，[1行目]で入力した文字を入れた状態で表示します。

もし，ユーザーに数として入力させたい場合は，[1行目]を次のようにします。

```
data = int(input("整数を入力してください。"))　　←整数としたい場合
data = float(input("小数を入力してください。"))　←小数としたい場合
```

8　指定回数の繰り返し（反復処理）

1. 同じことを繰り返したいときは，どうするのでしょう？
例えば，「 こんにちは 」を4回繰り返したいときは，次のようにすればいいですね。

```
print("こんにちは")
print("こんにちは")
```

```
print("こんにちは")
print("こんにちは")
```

でも，1万回繰り返したいとなったらどうしましょう？1万回コピペを繰り返せばできるのですが，大変です。

そこで，次のような便利な構文（for 構文）があります！

```
for 変数 in range( 回数 ):
□□□□繰り返したい処理
```

ここで注意しなければならない点が2つあります。

- [1行目]の一番最後にある「：」（コロン）を忘れないことです。
- [2行目]の先頭にある□□□□（半角スペースを4つ）入れることです。
  半角スペースを4つ以外のことをすると，正しく実行されません。　※ 半角スペースが分かるように□を使って表示しています。

先ほどの「 こんにちは 」を50回繰り返し表示させるには，次のようにします。

```
for x in range( 50 ):
□□□□print( "こんにちは" )
```

■練習07　for 構文を用いて「 おはよう 」を5回繰り返し表示させてください。

2. for 構文について，実は，もう少し細かく繰り返しを指示できます。
   次のようにコードを書くと，変数の値を「 値1 」から「 値2 」の1つ前まで増減量で変化させながら繰り返すことができます。

```
for 変数 in range( 値1 , 値2 , 増減量 ):
□□□□繰り返したい処理
```

次のコードを実行すると，どのように表示されるでしょう？

```
for x in range( 2 , 10 , 2):
□□□□print( x )
```

9　もし～ならば（条件分岐）

1. 条件によって，処理を変えたいときもありますよね。そのときに使える構文になります。

```
if 式や関数などの真/偽で答えられる条件:
□□□□(真である場合の)処理①
else:
□□□□(偽である場合の)処理②
```

このコードは，次のようになっています。

- [1行目]　if は，英語で学んだとおり「 もし～ 」のことです。真/偽は，数学で学んだとおり「 真＝正しい 」「 偽＝正しいとはいえない 」のことです。一番最後にある「：」（コロン）を忘れないでください。
- [3行目]　else と出てきますが，これより前の行までが，真の場合の処理となります。そして，これより後の行が，偽の場合の処理となります。
- [2行目][4行目]　先頭にある□□□□（半角スペースを4つ）入れることを忘れないでください。半角スペースを4つ以外のことをすると，正しく実行されません。　※ 半角スペースが分かるように□を使って表示しています。

式や関数などの真/偽で答えられる条件とは，どのように書けばよいのでしょう。ここでは，次のような比較演算子を使って書きます。

演算子	意味
==	左辺と右辺が等しい
!=	左辺と右辺が等しくない
<	左辺が右辺よりも小さい
<=	左辺が右辺以下
>	左辺が右辺よりも大きい
>=	左辺が右辺以上

次のコードを実行すると，どのように表示されるでしょう？

```
data = int(input("小テストの点数(10点満点)を入力してください。"))
if data >= 8:
□□□□print("あなたは合格です。")
else:
□□□□print("あなたは不合格です。")
```

■練習08　年齢を入力させ，次の条件で表示させてください。

・18才未満であれば「 あなたは未成年です 」と表示させる。
・18才以上であれば「 あなたは大人です 」と表示させる。

2. if 構文を入れ子式で組み合わせることで3段階に分岐させることもできます。

```
data = int(input("小テストの点数(10点満点)を入力してください。"))
if data >= 8:
```

```
    print("おめでとう。合格です！")
else:
    if data >= 6:
        print("おしい。あと少しで合格です。")
    else:
        print("あなたは不合格です。しっかり復習しよう。")
```

このコードは，次のように考えることができます。

- [2行目]　条件（8点以上）を満たせば[3行目]に進みます。条件を満たさない場合は，[4行目]以降に進みます。
- [5行目]　条件（8点未満であり，6点以上）を満たせば[6行目]に進みます。条件を満たさない場合は，[7行目]以降に進みます。
- まとめると，①8点以上，②6点以上，③6点未満の3つに分けられたことになります。

■練習09　年齢を入力させ，次の条件で表示させてください。
・18才未満であれば「 あなたは未成年です 」と表示させる。
・18才以上20才未満であれば「 あなたは大人ですが，お酒は飲めません 」と表示させる。
・20才以上であれば「 お酒も飲める大人です 」と表示させる。

さらに組み合わせれば，4段階以上に分けることも可能です。

3. （おまけ）3段階以上に分岐させる場合の別の方法もあります。

```
if 式や関数などの真/偽で答えられる条件①:
    (真である場合の)処理①
elif 式や関数などの真/偽で答えられる条件②:
    (真である場合の)処理②
else:
    (偽である場合の)処理③
```

先ほど，小テストの点数を3つに分けましたが，この構文を使って書き直してみると，次のようになります。

```
data = int(input("小テストの点数(10点満点)を入力してください。"))
if data >= 8:
    print("おめでとう。合格です！")
elif data >= 6:
    print("おしい。あと少しで合格です。")
else:
    print("あなたは不合格です。しっかり復習しよう。")
```

4段階以上に分けたいときは，「 elif 」とその次の行をセットで増やしていけばよいです。

10　条件が正しいときに繰り返し（反復処理2）

1. 条件が正しいときに繰り返すという反復処理として，while構文というのがあります。

```
while 変数を用いた真/偽で答えられる条件:
    (真である場合の)処理
    変数を変化させる処理
```

このコードは，次のようになっています。

- [1行目]　whileは，英語で学んだとおり「 ～の間 」のことです。一番後ろに：（コロン）を忘れないでください。
- [3行目]　とても重要な行です。[1行目]の変数を変化させる処理を必ず行います。そうでないと，永遠に繰り返し終了しません。

次のコードを実行すると，1桁の奇数が表示されます。

```
suu = 1
while suu <10:
    print(suu)
    suu += 2
```

[4行目]を見てください。これは，「 suu 」に「 2 」を加えるという意味の式になります。

■練習10　while構文を用いて，10以上50以下にある偶数を表示してください。

11　リスト（配列）

1.　次のようにすると，1つの変数の中に複数のデータを入れることができます。

```
変数 = [ 値1 , 値2 , 値3 , ・・・ ]              ←変数に数を入れる場合
変数 = [ "文字1" , "文字2" , "文字3" , ・・・ ]   ←変数に文字を入れる場合
```

表示させる場合には，前から何番目を表示させるかを指定してします。なお，コンピュータは0番目から始まることに注意してください。

```
aisatsu = [ "おはよう", "こんにちは", "こんばんは" ]
print( aisatsu[0] )
```

上のコードを実行すると「 おはよう 」が表示されますね。

■練習11　上のコードを実行して「 こんにちは 」を表示させたい場合，aisatsu[0]の部分をどのようにすればよいでしょう。

2. リストを順番に使いたい場合には，for 構文と組み合わせることでできます。

```
for 変数 in リスト:
    繰り返したい処理
```

例えば，次のコードを実行すると，日曜日から順に土曜日まで表示されます。

```
youbi = [ "日曜日", "月曜日", "火曜日", "水曜日", "木曜日", "金曜日", "土曜日" ]
for x in youbi:
    print( x )
```

■練習12　１月，２月，･･･，１２月のリストをつくり，順に表示させてください。

3. リストの中からランダムに使いたい場合には，次のようにするとできます。

```
import random
変数 = [ "文字1", "文字2", "文字3", ･･･ ]　　←文字ではなく，数でも可
print( random.choice( 変数 ) )
```

このコードは，次のようになっています。

- [1行目]　pythonでは，よく使われるコードについてパッケージ化がされており「 モジュール 」と呼ばれています。今回は「 random 」という乱数に関するモジュールを利用します。
- [3行目]　「 random 」というモジュールの中にある「 choice 」という関数（ランダムにリストの中から選ぶ）で選ばれたものを表示します。
  例えば，次のコードを実行すると，おみくじの結果が表示されます。実行してみてください。

```
import random
omikuji = [ "大吉", "中吉", "小吉", "吉", "末吉", "凶", "大凶" ]
print( random.choice( omikuji ) )
```

いかがだったでしょうか。おみくじがランダムに表示されてきます。

■練習13　グー，チョキ，パーのリストをつくり，ランダムに表示させてください。

4. （おまけ）次のようにすると，サイコロのように1～6の中から，ランダムに1つ表示させることができます。

```
import random
print( random.randint( 1 , 6 ) )
```

このコードは，次のようになっています。

- [2行目]　「 random 」というモジュールの中にある「 randint 」という関数（1～6の中からランダムに整数を1つ選ぶ）で選ばれたものを表示します。

■練習14　1～100の中からランダムに整数を表示させてください。

12　応用実習Ⅰ（図形描画）

1. 次のようにモジュール（関連性のある「プログラム部品」を1つのファイルにまとめたもの）を使うことで図を描くことができます。

```
!pip3 install （モジュール名）
```

ここでは，図を描くために，モジュール名のところには「 ColabTurtle 」を入力します。

2. モジュール「 ColabTurtle 」では，カメを指示どおりに動かしながら，図形を描きます。次のようにすると，正三角形を描くことができます。

```
from ColabTurtle.Turtle import *
initializeTurtle(initial_speed = 2)
color('orange')
bgcolor('white')
width(1)
for i in range(3):
    forward(200)
    left(120)
```

このコードは，次のようになっています。最後の3行が特に大事です。

- [1行目]　モジュール「 ColabTurtle 」の中の「 ColabTurtle.Turtle 」からすべての機能を使うことが示されています。

- [2行目]　「 initial_speed 」とは，図を描くカメの表示スピード（1～10）を表しています。
- [3行目]　カメが描く線の色を示しています。
- [4行目]　背景の色を示しています。
- [5行目]　カメの描く線の幅を示しています。
- [6～8行目]　セクション7で学んだ「 繰り返し 」を行う for 構文です。
- [7行目]　「 forward(200) 」とは，カメの進行方向に向かって，まっすぐに200ピクセル進むことを示しています。
- [8行目]　「 left(120) 」カメの進行方向に向かって，左に120度方向を変えることを示しています。

■練習15　1辺が300ピクセルの正方形を描いてください。
■練習16　1辺が200ピクセルの正五角形を描いてください。
■練習17　1筆書きの星（5辺）を描いてください。ただし，辺の長さはいくらでもかまいません。

2. 次のようにリスト（配列）を利用すると，描く辺の色を変えることができます。

```
initializeTurtle(initial_speed= 6)
col = [ "orange", "limegreen" , "gold" , "plum" , "tomato" ]
bgcolor('white')
width(1)
for i in range(4):
    color( col[ i ] )
    forward(200)
    left(90)
```

■練習18　1筆書きの星（5辺）の各辺を「 orange 」「 limegreen 」「 gold 」「 plum 」「 tomato 」で描いてください。ただし，辺の長さはいくらでもかまいません。
■練習19　隣の人とは異なる図形を想像して1つ描いてください。

13　応用実習Ⅱ（ソート）

リストなどの中を昇順(a,b,c,...,z)，降順(z,y,x,...,a)に並べ替えることをソートといい，いろいろなソートの仕方があります。ここでは，３つやってみましょう。

バブルソート

1. バブルソートとは，隣り合う要素の大小を比較しながら並べ替えを実現するアルゴリズムです。この動画をみて，アルゴリズムを理解しましょう。
2. このアルゴリズムをコードで表現すると，次のようになります。

```
def bubble_sort(a):
    for i in range(0, len(a), 1):
        for j in range(0, len(a) - i - 1, 1):
            if a[j] > a[j + 1]:
                temp = a[j]
                a[j] = a[j + 1]
                a[j + 1] = temp
```

このコードは，次のようになっています。

- [1行目]　処理するための関数を「 bubble_sort 」（変数は a ）という名前で定義する，と言っています。
- [2・3行目]　ここは反復処理ですね。「 8 指定回数の繰り返し（反復処理）」の2を参照しましょう。なお，「 len(a) 」は，変数 a （実際にはリストになります）の数値の数を表します。
- [4行目]　隣り合う a[j] と a[j + 1] を比較します。
  もし a[j + 1] が a[j] よりも小さいならば， a[j] と a[j + 1] を入れ替えるために，[5行目]で a[j] の中身を temp という変数に一時的に移し，[6行目]で a[j + 1] の中身を a[j] に移し，[7行目]で temp の中身を a[j + 1] に移します。

では，実際にこのアルゴリズムでソートしてみましょう。上のコードの後に，次の４行分のコードを加えて実行してください。

```
a = [7, 22, 22, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]
print("ソート前 → ", a)
bubble_sort(a)
print("ソート後 → ", a)
```

なお，このコードは，次のようになっています。

- [1行目]　並び替えたい数字のリストを変数 a としています。
- [2・4行目]　ソート前，ソート後のリストを表示させます。
- [3行目]　定義した関数「 bubble_sort 」に代入し，ソートの処理を行います。

■練習20　数字のリストを変更して，正しくソートされるかを確認してください。
■練習21　次のコードを実行したら，「 ●回 」と表示されます。何の回数が表示されているか，考えましょう。また，実行してそれを確認しましょう。（ヒント：数字のリストの数を減らすと分かりやすくなります。）

```
def bubble_sort(a):
    c = 0
    for i in range(0, len(a), 1):
        for j in range(0, len(a) - i - 1 , 1):
```

```
        if a[j] > a[j + 1]:
            c = c + 1
            temp = a[j]
            a[j] = a[j + 1]
            a[j + 1] = temp
    print(c, "回")

a = [7, 22, 22, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]
print("ソート前 → ", a)
bubble_sort(a)
print("ソート後 → ", a)
```

選択ソート

1. 選択ソートとは，リスト内のデータから最小値を探索し，最小値から順に取り出すことで並べ替えを実現するアルゴリズムです。この動画をみて，アルゴリズムを理解しましょう。
2. このアルゴリズムをコードで表現すると，次のようになります。

```
def selection_sort(a):
    for i in range(0, len(a), 1):
        for j in range(i+1, len(a), 1):
            if a[j] < a[i]:
                temp = a[i]
                a[i] = a[j]
                a[j] = temp
```

このコードは，次のようになっています。

- [1行目]　処理するための関数を「 selectin_sort 」（変数は a ）という名前で定義する，と言っています。
- [2・3行目]　ここは反復処理ですね。「8 指定回数の繰り返し（反復処理）」の2を参照しましょう。なお，「 len(a) 」は，変数 a （実際にはリストになります）の数値の数を表します。
- [4行目]　a[i] が最小値の候補です。
  もし a[j] が a[i] よりも小さいならば， a[i] と a[j] を入れ替えるために，[5行目]で a[i] の中身を temp という変数に一時的に移し，[6行目]で a[j] の中身を a[i] に移し，[7行目]で temp の中身を a[j] に移します。

では，実際にこのアルゴリズムでソートしてみましょう。上のコードの後に，次の４行分のコードを加えて実行してください。


```
a = [7, 22, 22, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]
print("ソート前 → ", a)
selection_sort(a)
print("ソート後 → ", a)
```

なお，このコードは，次のようになっています。

- [1行目]　並び替えたい数字のリストを変数 a としています。
- [2・4行目]　ソート前，ソート後のリストを表示させます。
- [3行目]　定義した関数「 selectin_sort 」に代入し，ソートの処理を行います。

■練習22　数字のリストを変更して，正しくソートされるかを確認してください。
■練習23　選択ソートのコードを修正して，入れ替えを行った回数が表示されるようにしてください。（ヒント：バブルソートでやったような…）

クイックソート

1. クイックソートとは，リスト内の1つのデータを軸として，大小2つに分割した後，分割したデータに対して同じ処理を再度行うことにより並べ替えを実現するアルゴリズムです。この動画をみて，アルゴリズムを理解しましょう。
2. このアルゴリズムをコードで表現すると，次のようになります。

```
def quick_sort(a, hajime, owari):
    m = int((hajime + owari) / 2 )
    i = hajime
    j = owari
    while(i < j):
        while a[i] < a[m]:
            i = i + 1
        while a[j] > a[m]:
            j = j - 1
        if i >= j:
            break
        temp = a[i]
        a[i] = a[j]
        a[j] = temp
        if i == m:
            m = j
        elif j == m:
            m = i
        i = i + 1
        j = j - 1
```

```
    if hajime < i - 1:
        quick_sort(a, hajime, m-1)
    if owari > j + 1:
        quick_sort(a, m+1, owari)
```

このコードは，次のようになっています。

- [1行目]　処理するための関数を「 quick_sort 」（変数は a ， hajime ， owari の3つ）という名前で定義する，と言っています。
- [2行目]　比較するための基準（ここでは中央値）を m とします。
- [3・4行目]　比較する対象 i ， j の初めの値を決めました。
- [5～20行目]　ここは反復処理ですね。「１０　条件が正しいときに繰り返し（反復処理2）」を参照しましょう。
- [11行目]　ここは[10行目]の条件「 i >= j 」において「 正しい（真） 」の場合，「 break 」します（[5～20行目]の反復処理から抜けます）。つまり[20行目]までを飛ばし[21行目]の処理をします。
- [21・22行目]　 m を基準に分けた小さい方の集まりにおいて，「 quick_sort 」を実行します（[1行目]に戻り，再び処理が始まります）。
- [23・24行目]　 m を基準に分けた大きい方の集まりにおいて，「 quick_sort 」を実行します（[1行目]に戻り，再び処理が始まります）。

では，実際にこのアルゴリズムでソートしてみましょう。上のコードの後に，次の４行分のコードを加えて実行してください。

```
a = [7, 22, 22, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]
print("ソート前 → ", a)
selection_sort(a)
print("ソート後 → ", a)
```

なお，このコードは，次のようになっています。

- [1行目]　並び替えたい数字のリストを変数 a としています。
- [2・4行目]　ソート前，ソート後のリストを表示させます。
- [3行目]　定義した関数「 selectin_sort 」に代入し，ソートの処理を行います。

■練習24　数字のリストを変更して，正しくソートされるかを確認してください。

14　発展実習（人工知能(AI)）

1. まず，人工知能(AI)の歴史について，学んでおきましょう。(出典：https://www.tryeting.jp/column/1192/)

   最初のAIブームは1960年代に興りました。第1次AIブームで研究されたのは、コンピューターを使って推論・探索をすること。パズルや明確なルールがあるゲームなど、特定の問題をコンピューターが次々の解く姿に世間を驚かせていました。第1次AIブームは、ルールが不明確で複雑な問題を解けないと分かると、次第に下火になっていきました。このとき人工知能が解くことのできた問題は、「おもちゃの問題（トイ・プロブレム）」と呼ばれることになります。

   一度冬の時代に入ったAIですが、1980年代で再び勢いを取り戻します。第2次AIブームを起こす引き金になったのが「エキスパートシステム」の実現です。エキスパートシステムとは、専門家の判断を代行するシステム。ルールに基づいたデータを入力し、その答えを条件反射で答えるプログラムです。ビジネスでの導入例も出現するなど好調に見えたエキスパートシステムですが、再び限界の壁にぶつかることになります。問題点は2つありました。1つは、人間の持つ「一般常識」レベルの膨大な知識を記述しなければならないこと。2つ目の問題点は、例外処理、矛盾したルールに対応できないことでした。

   2006年から現在に至るまで、第3次ブームの真っ只中にいます。第3次ブームが興った背景として挙げられるのが、機械学習の登場、ディープラーニングの登場、ビッグデータの普及などです。第3次ブームの注目点は、AIが自ら学習し推測する点にあります。それ以前では、人間がルールを定義し、AIが問題を解決したり知識を取り出す手法が主流でした。

   機械学習とは、人間の学習能力を模倣し、大量のデータを処理しながら「分け方」を自動的に習得する技術です。機械学習のコンセプト自体は第2次ブームにもありましたが、そのころはコンピュータの計算速度や容量が追いついておらず、実現できませんでした。

   ディープラーニングは機械学習の一手法で、分けるための判断軸（特徴量）をAIで見つけることができる技術です。上記の機械学習は、「どの特徴量に注目して、情報を取り出して、分けるか」に関して、人間の力が必要でした。特徴量に何を選ぶかによって、分け方の精度は大きく変わります。

このセクションでは，「手書きの数字を認識するプログラム」を作成する実習をしていきます。

2. それでは，画像を認識・表示するための準備をします。

   大量の「数字の画像」と「それが何の数字か」のペアを与えて，「数字の特徴」を学習させ，「手書きの数字」が何の数字なのかを判定させます。

   ここでは，「 sklearn 」というモジュールを利用します。 datasets.load_digits() でデータを読み込みます。この中には，「手書きの数字画像(data，images)」と「それが何の数字か(target)」がペアでたくさん入っています。

   まずは，sklearnにどんなデータが用意されているかを確かめることから始めます。

```
import sklearn.datasets
digits = sklearn.datasets.load_digits()
print("データの個数＝", len(digits.images))
```

```
print("画像のデータ＝", digits.images[0])
print("何の数字か＝", digits.target[0])
```

このコードは，次のようになっています。

- [1行目]　モジュール「 sklearn 」の中の関数「 datasets 」を読み込みます。
- [2行目]　「 datasets 」の中に用意されているデータを読み込みます。
- [3行目]　「 datasets 」の中に用意されているデータの総数を数えています。
- [4行目]　読み込んだデータ(8x8の数値)を表示します。濃淡を数値で表しており，0が1番明るい白で，16が1番暗い黒です。
- [5行目]　読み込んだデータが何の数字であるかを示しています。

■練習25　上のコードを実際に実行してみてください。

[4行目]で数値データとして表示していますが，正直なところ分かりづらいです。そこで，画像で表示させ体と思います。そのためには，モジュール「 matplotlib 」を使います。

```
import matplotlib.pyplot as plt
digits = sklearn.datasets.load_digits()
plt.imshow(digits.images[0], cmap="Greys")
plt.show()
```

このコードは，次のようになっています。

- [1行目]　モジュール「 matplotlib 」の中の関数「 pyplot 」を追加で読み込みます。ここでは，「 plt 」という略語で使うことにしています。
- [2行目]　「 datasets 」の中に用意されているデータを読み込みます。
- [3行目]　読み込んだデータをグレースケールの濃淡画像にしています。
- [4行目]　グレースケールの濃淡画像になったデータを表示します。

■練習26　上のコードを実際に実行して画像を表示させてください。

今は1つのデータだけを確認しましたが，もっと他のデータはどうなっているのでしょうか。セクション7で学んだ「 繰り返し 」を行う for 構文を使って，まとめて30個表示させてみましょう。

```
for i in range(30):
    plt.subplot(3, 10, i + 1)
    plt.axis("off")
    plt.title(digits.target[i])
    plt.imshow(digits.images[i], cmap="Greys")
plt.show()
```

このコードは，次のようになっています。

- [1行目]　for 構文により30回繰り返します。
- [2行目]　3x10に順番に表示させます。
- [3行目]　表示の際に枠線を非表示にします。
- [4行目]　表示した数字が何であるかをタイトルとして表示します。
- [5行目]　読み込んだデータをグレースケールの濃淡画像にしています。
- [6行目]　グレースケールの濃淡画像になったデータを表示します。

3.　「手書きの数字を認識するプログラム」を作成します。
ちょっと長いですが，がんばって入力しましょう。

```
import sklearn.svm
import PIL.Image
import numpy
def imageToData(filename):
    grayImage = PIL.Image.open(filename).convert("L")
    grayImage = grayImage.resize((8, 8), PIL.Image.ANTIALIAS)
    numImage = numpy.asarray(grayImage, dtype = float)
    numImage = numpy.floor(16 -16 *(numImage / 256))
    numImage = numImage.flatten()
    return numImage
def predictDigits(data):
    digits = sklearn.datasets.load_digits()
    clf = sklearn.svm.SVC(gamma = 0.001)
    clf.fit(digits.data, digits.target)
    n = clf.predict([data])
    print("この画像は "+str(n)+" です！")
data = imageToData("what.png")
predictDigits(data)
```

このコードは，次のようになっています。

- [1～3行目]　3つのモジュールや関数を追加で使います。
- [4～10行目]　「 imageToData 」という関数を定義しています。
- [11～16行目]　「 predictDigits 」という関数を定義しています。
- [17行目]　プログラムに判定させたい手書きの数字(「 what.png 」というファイル名)をdataという変数で読み込みます。
- [18行目]　関数「 predictDigits 」で「 what.png 」を処理し，判定します。

もう少し詳しく見ていきましょう。
まず，[4～10行目]　「 imageToData 」という関数を定義している部分についてです。

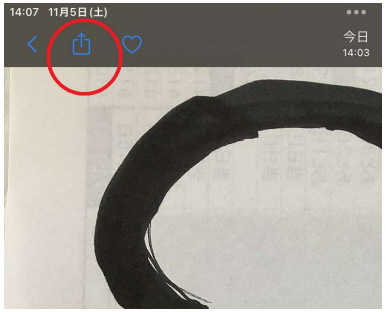
- [5～6行目]　与えられた画像を8x8のグレースケール画像に変換します。
- [7行目]　8x8のグレースケール画像を数値リストに変換します。濃淡データは255(白)～0(黒)となっている。
- [8行目]　255(白)～0(黒)の濃淡データを0(白)～16(黒)になるように変換します。
- [9行目]　0(白)～16(黒)になった濃淡データを数値リストに変換します。

次に，[11～16行目]　「 predictDigits 」という関数を定義している部分についてです。

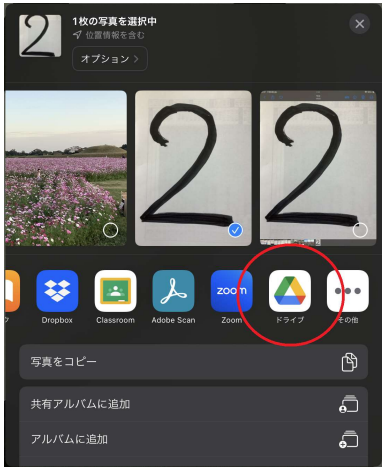
- [12行目]　学習用のデータを読み込みます。
- [13～14行目]　機械学習をします。
- [15行目]　学習したデータをもとに判定をします。
- [16行目]　判定結果を表示します。

4. 「手書きの数字を認識するプログラム」を作成して，実際に判定してみましょう。
まずは，上のコードを入力してください。
次に，プログラムに判定させたい「手書きの数字」を準備します。

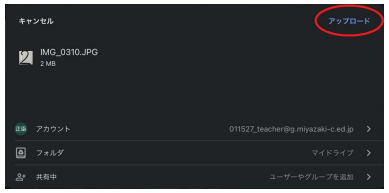
1. 紙に太い字で，０から９の中から１つ書きます。それをタブレットで，画面いっぱいになるように拡大して，パシャリと写真で撮ります。（前の教卓の上に紙はあります。）
2. 撮った写真をGoogle Driveへ保存するために，写真を開いて，画面左上にある赤丸部分をタップします。



3. 画面にある赤丸部分(Google Drive)をタップします。

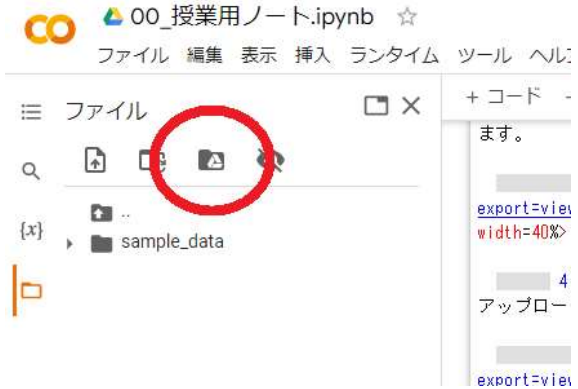


4. 選択した写真に間違いがなければ，右上の「 アップロード 」をタップします。

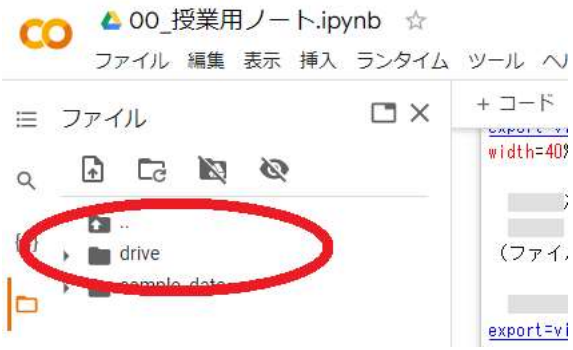


5. Colabノートの左端にあるフォルダのアイコン（ファイルと表示される）をクリックします。

6. 表示されたら，赤丸部分のgoogle Driveをクリックします（ドライブをマウントします）。

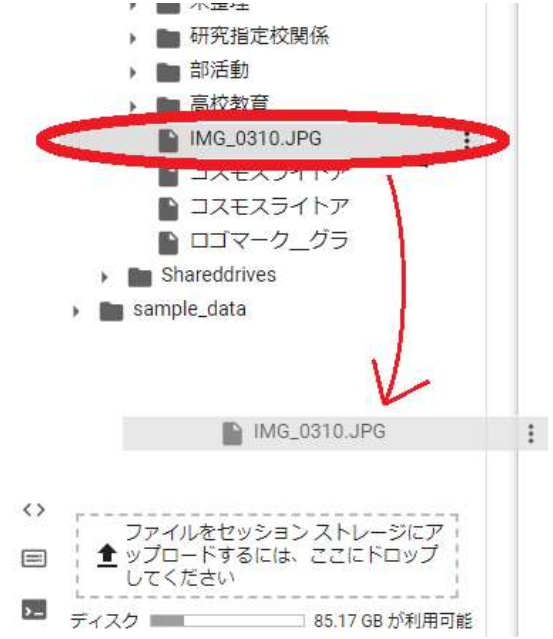


7. 途中で「このノートブックにGoogleドライブのファイルへのアクセスを許可しますか？」と聞かれるので，「Googleドライブに接続」をクリックします。次の写真のように「drive」(赤丸内)が表示されればOKです。



次に，プログラムに判定させる準備をします。

8.「drive」の左側の▼をクリック，続けて「MyDrive」の左側の▼をクリックします。プログラムに認識させたい画像（下の写真では「IMG_0310.JPG」）を矢印の場所へドロップします。



9. 下の写真のようにsample_dataの下に「IMG_0310.JPG」があれば完了です。

さあ，準備は整いました！プログラムを実行させてください。ちゃんと判定できるかな。

Colab の有料サービス - 契約解除はこちら