説明

金子邦彦

①情報工学における創造力と未来志向

②情報工学演習I、IIと、卒業研究による学習の深化

③プログラミングとAIの理解と適用

④AIに対する理解と適用

アウトライン

１．情報工学の魅力

２．金子邦彦研究室紹介

３．①楽しもう

４．②AIのプログラム例

１．情報工学の魅力

デジタル社会と情報工学

オンラインでの交流やコミュニケーション

デジタルサービス（動画配信，オンラインショッピングなど）

莫大な情報の管理・処理

人間と AI の協働

情報工学は，**デジタル社会**を実現し発展させるための大切な学問分野.

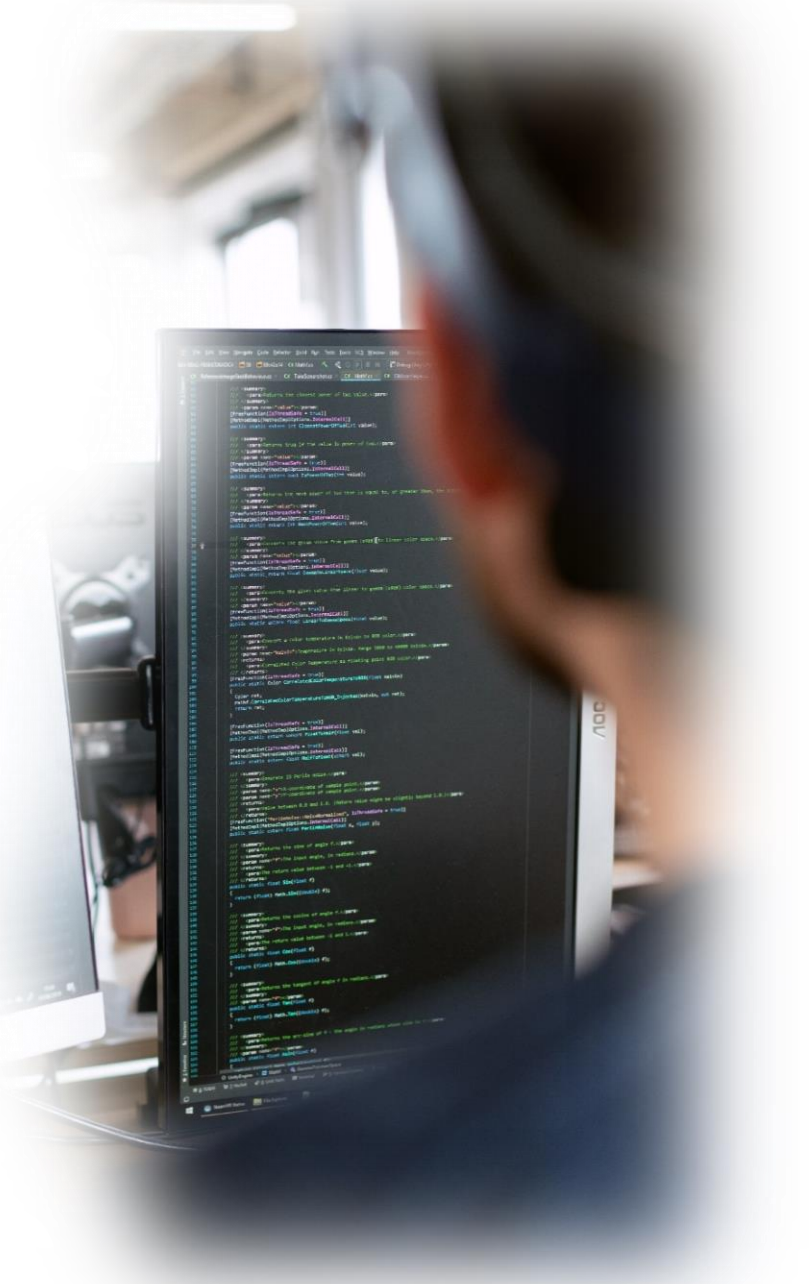
情報工学の面白さ

想像力を活かす楽しさ：自分のアイデアが形に.

未来の技術を学ぶワクワク感：AI，仮想現実，IoT など.

夢が広がる進路：IT企業や製造業，クリエイティブな職種などで活躍.

専門的な実力と，**未来を切り開く力**を身につけ，**社会に貢献**しながら，**自分の夢を追求する**ことができます.




２．金子邦彦研究室紹介

金子邦彦 研究室

◆ **現実世界とデジタルの融合**

街並み，住環境，人間の顔，姿勢、ものの形

◆ **社会の課題解決へ**

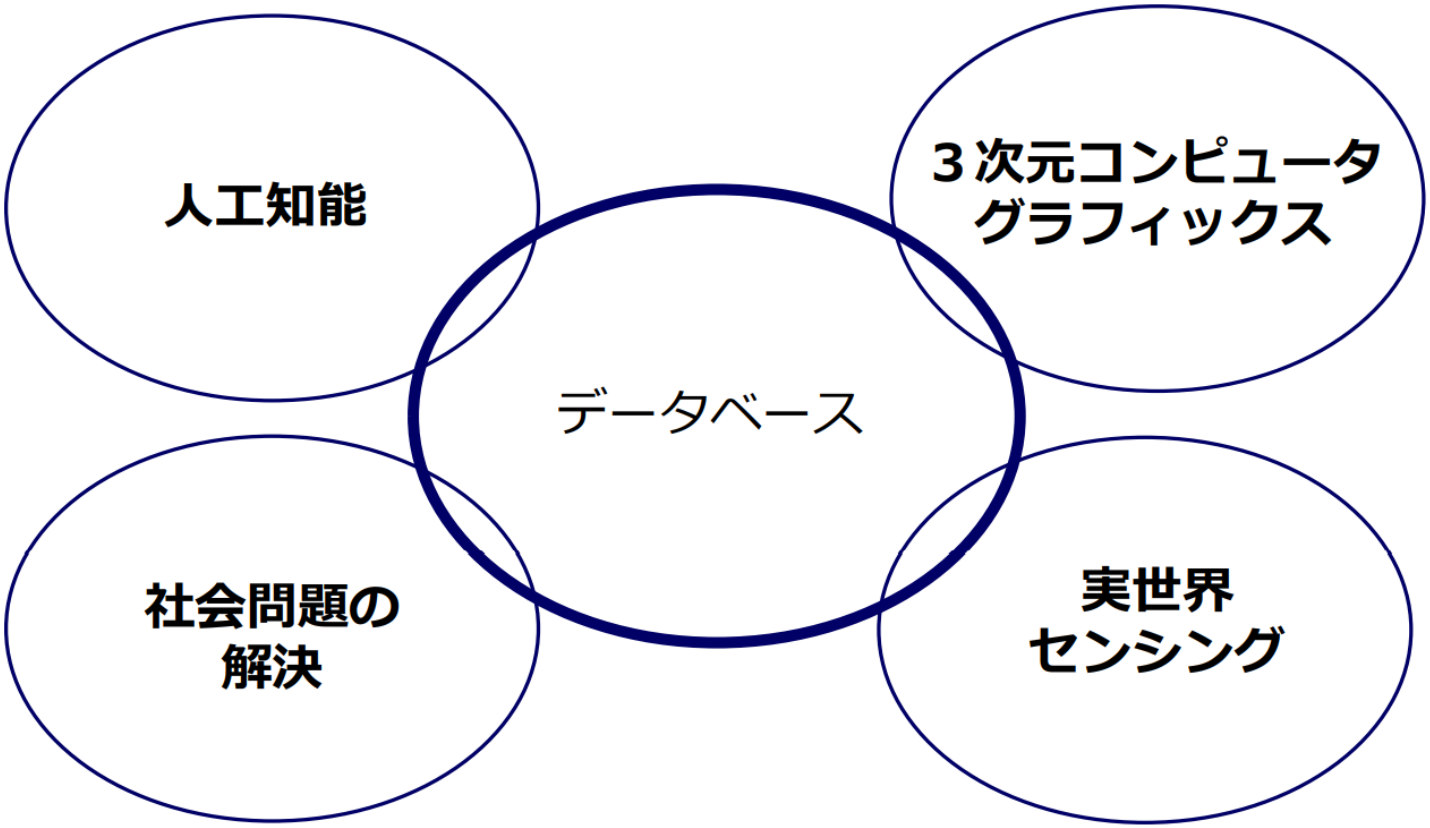
面白い，安全，安心，便利，快適

◆ **人工知能，データベース，３次元データの研究室**

学生は，最新技術を知る → 各自が創造力，問題解決力を発揮する

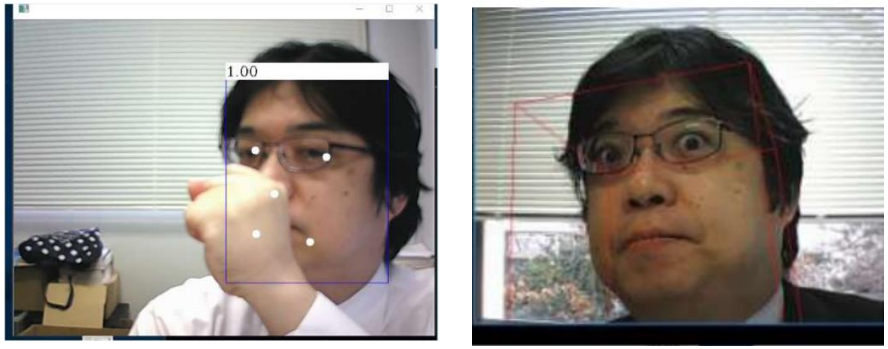
身近な機器（スマートフォン）とAIコンピュータを活用。
広く普及できる、社会や生活に根差した研究を追求。

データベースと関連する研究領域



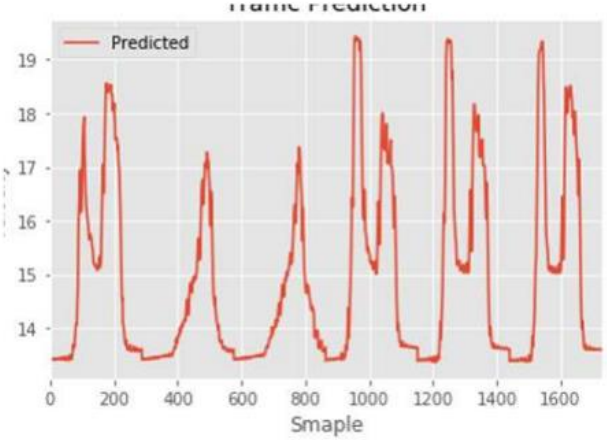
AI 活用．AI と人間の共同により不可能を可能へ

人間のきめ細かな把握



画像理解

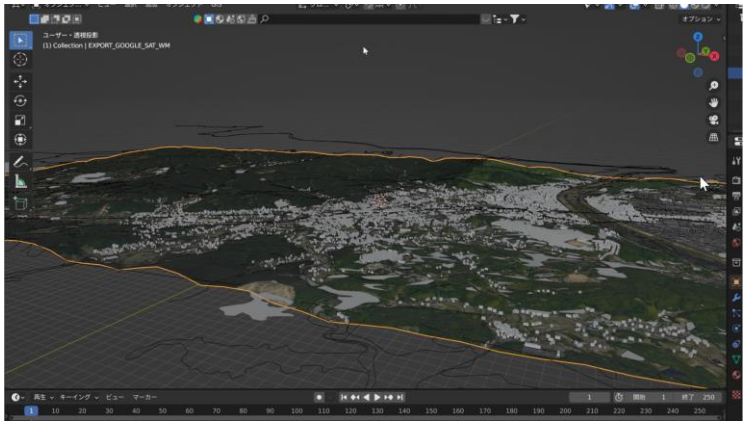
未来予測、変化要因の分析



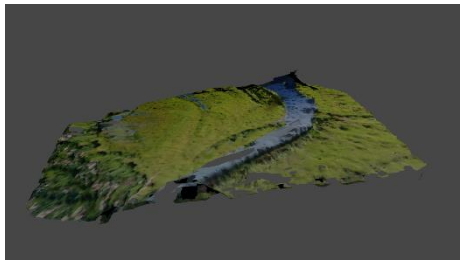
画像補正

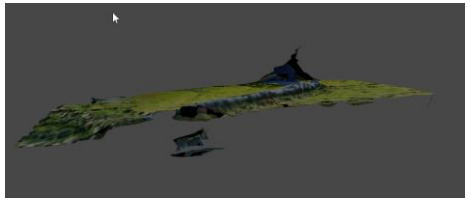
３次元．現実世界をデジタルと融合

３次元の福山市の再現

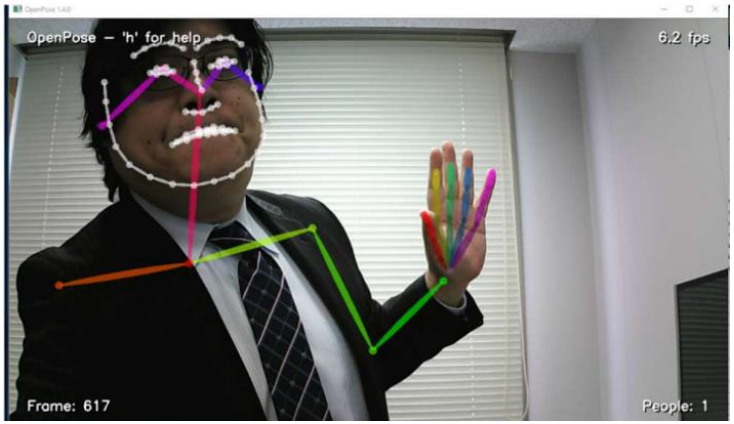


３次元再構成

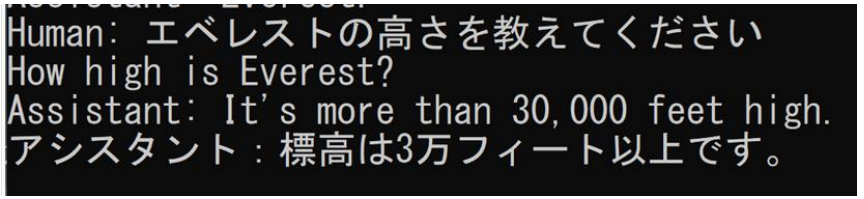




人体の姿勢推定（マーカーレス）



対話によるデータアクセス



卒業研究のメリット

- 各自が興味を持ち，熱中できることを選択できる．楽しく取り組むことができる.

- 研究に必要な専門知識，ITスキル，ロジカルシンキング，創造力，問題解決力を修得できる.

- 研究室での仲間や指導教員との交流を通じて，コミュニケーション能力の向上も期待できる.

- 自己の専門性を高めて，将来に役立ててください！

金子邦彦　（かねこくにひこ）

【研究領域】
データベース応用、人工知能応用，データベース基盤技術、ITシステム
【実績】
・学術論文等：２７編、査読付き国際会議：７５編、その他講演多数
・教科書等：３
・授業担当経験：のべ２４科目
・科学研究費：のべ１１件　概算のべ数千万円　他大学との共同多数
・共同研究、受託研究など：のべ１０件　概算のべ一億円　国際共同研究あり
・学部生、大学院生の指導経験多数

詳しくは　https://www.kaneko.jp/perf-j.html

人工知能、**画像処理**、**３次元コンピュータグラフィックス（VR含む）**、**Webシステム**、**知的システム**や**社会システム**の成功には、
データベースが必要　という気持ちで進めています

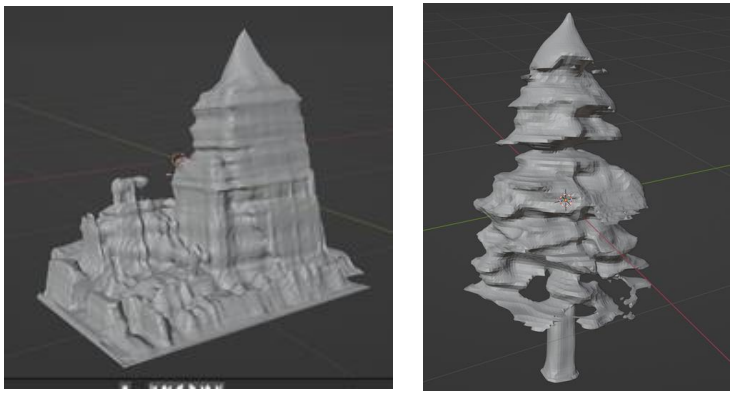
３．説明

金子邦彦研究室
①まずは楽しもう！

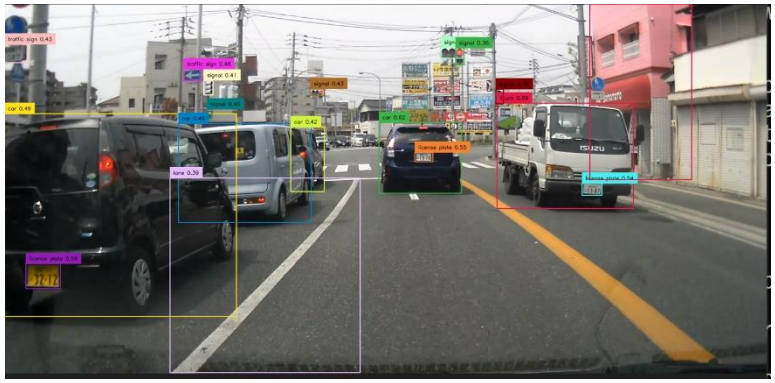
人工知能を学ぶ楽しさ

人工知能：人間の思考を模倣し、超えることを目指す挑戦。人工知能は、現在進行形の最先端技術であり、未来に向けてもその発展が続く、刺激と興奮に満ちた分野である。

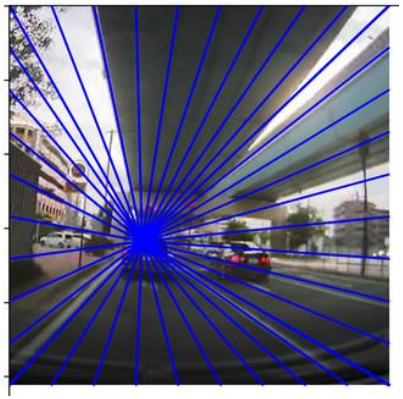
顔情報処理



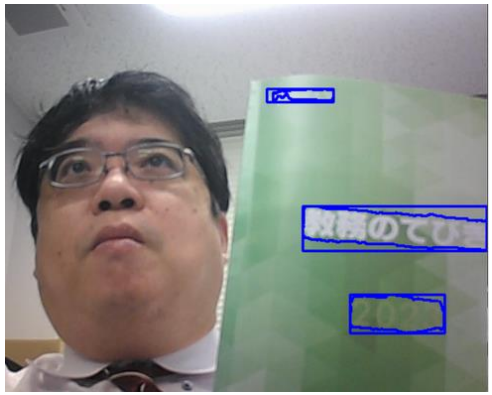
3 次元データの合成



物体検出、指定されたキーワードはAIには初見（ゼロショット）



消失点推定



テキスト検出

プログラミングの楽しさと達成感

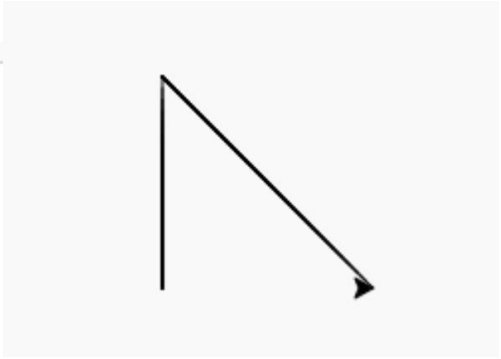
- **楽しさ**
  - **未来の技術を学ぶ**ことは楽しい。
  - プログラミングは**クリエイティブな行為。**
  - 視覚的なプログラムを書くことで、ゲーム感覚をもって楽しみながら学習することも可能。
- **達成感**
  - **自分のアイデアを形に**することで得られる達成感
  - 自分でデザインし、問題が生じたときは自分で解決していく。
  - **自分の手でプログラムを完成させるプロセス**は、大いに充実感をもたらすもの。

テーマ

① Python でのグラフィックス

プログラミングのクリエイティブな側面を実感し、ITエンジニアとしての自信を深める

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0,100)
4  t.goto(100,0)
```



視覚的な結果を得るプログラム

② 人工知能による画像認識

データから新しい価値を創造できるAIエンジニアに興味を深め、視野を広げる

画像が犬なのか、
猫なのか
を判別するAI を動作

プログラミング

- **プログラミング**は**人間の力を増幅**する
- **シミュレーション、大量データ処理、AI連携、ITシステム制作**など、さまざまな活動で、**プログラミング**は役立つ
- **プログラミングはクリエイティブな行為**
- **さまざまな作業を自動化したいときにも役立つ**
- **基本を押さえる**（パッケージ、オブジェクト生成、メソッド）。1年生のうちに学んだことが大切であり、超高度な知識がたくさん必要ということはありません

オブジェクトとメソッド

- **オブジェクト**：コンピュータでの**操作や処理の対象となるもの**

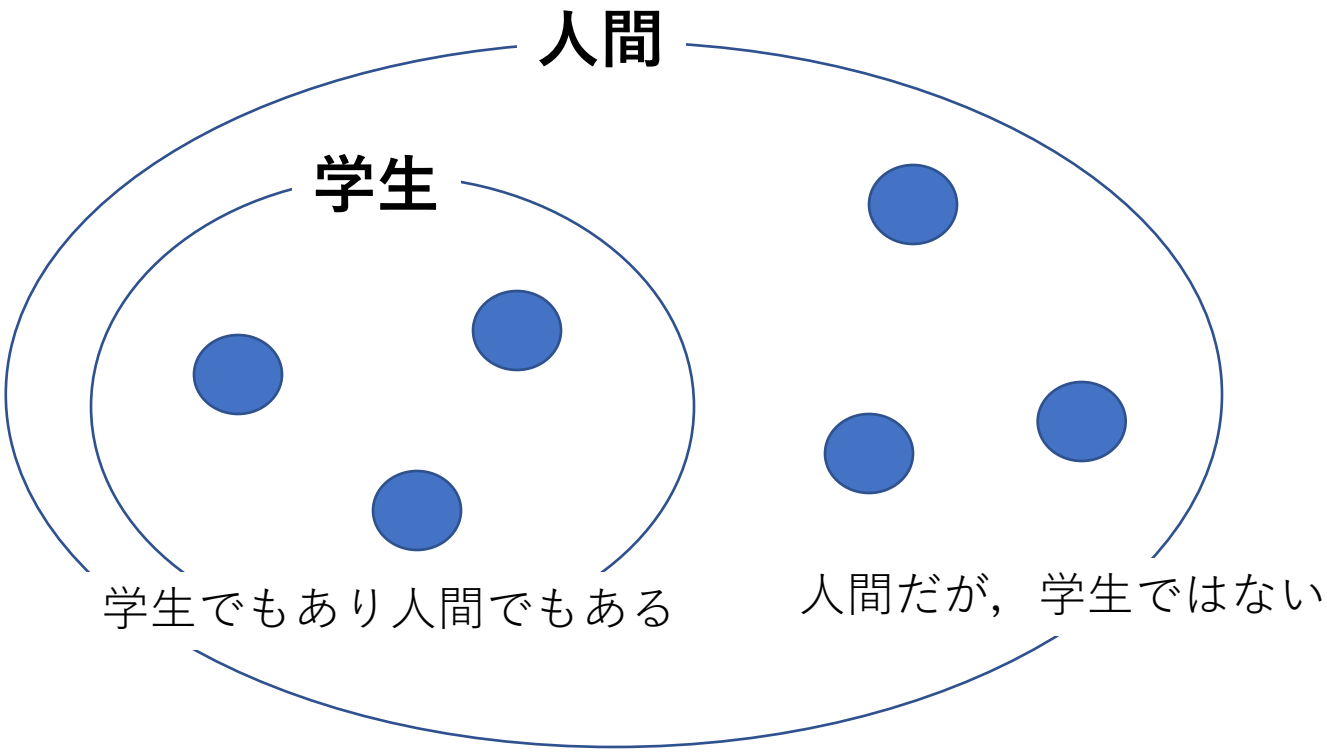
hero.moveDown()	**hero　オブジェクト** **moveDown()　メソッド** 間を「.」で**区切って**いる

- **メソッド: オブジェクト**に属する機能や操作．オブジェクトがもつ能力に相当する
- **引数：メソッド**が行う操作の詳細に関する情報，**メソッド**呼び出しのときに、引数を指定できる

hero.attack("fence", 36, 26)

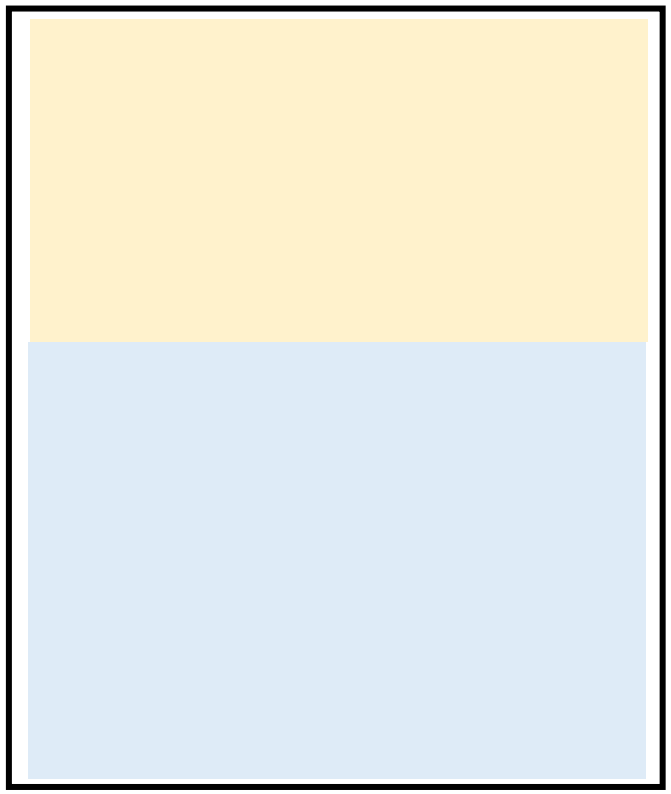
クラスとオブジェクト

クラスは，**同じ種類のオブジェクトの集まり**と考えることができる



インポート

インポートにより
ファイルを丸ごと取り込む



Python プログラムのファイル
（関数、変数、クラスなどの
オブジェクトについて記載）

Python プログラムのファイル

モジュールのインポートの方法

単一の Python プログラム
ファイル = モジュール

```
def foo(x):
    return x + 3
```

インポート

import bar でインポート

```
import bar
print(bar.foo(100))
```

bar.foo は bar モジュール内の foo オブジェクトの意味

Python プログラムのファイル

拡張子「.py」のファイル名でファイルに保存
（ファイル名を bar.py とする）

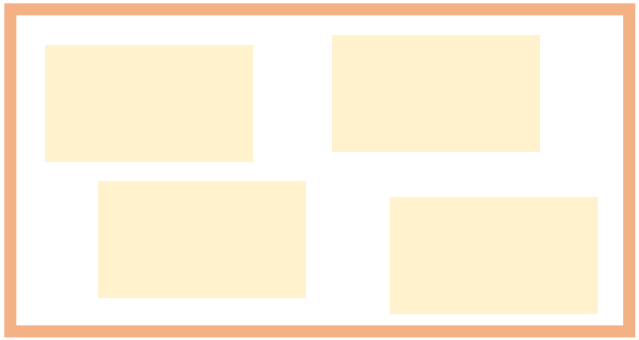
ファイル名は何でもよい

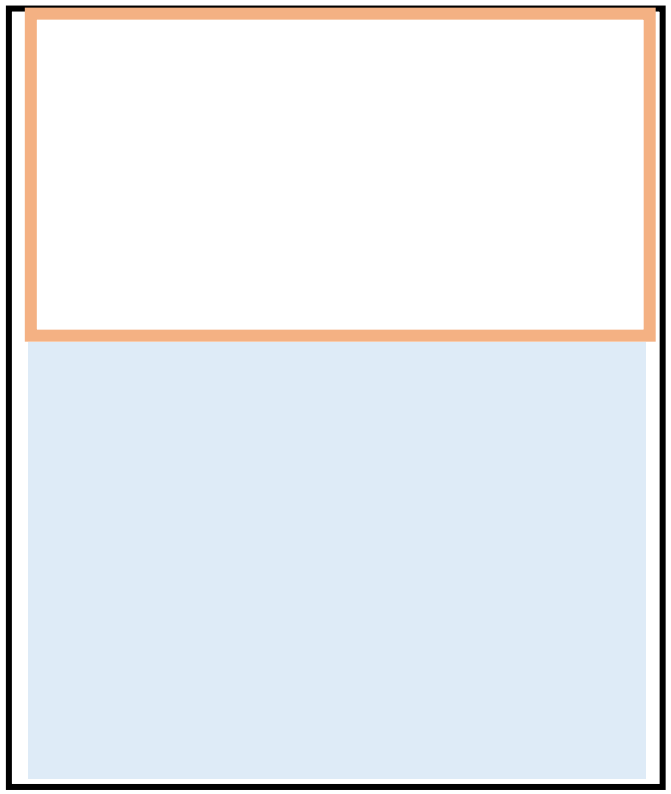
実行結果として
103 が表示される

```
>>> import bar
>>> print(bar.foo(100))
103
```

パッケージのインポート

インポートにより，パッケージのファイルを丸ごと取り込む





複数の Python プログラムをパッケージにすることも可能

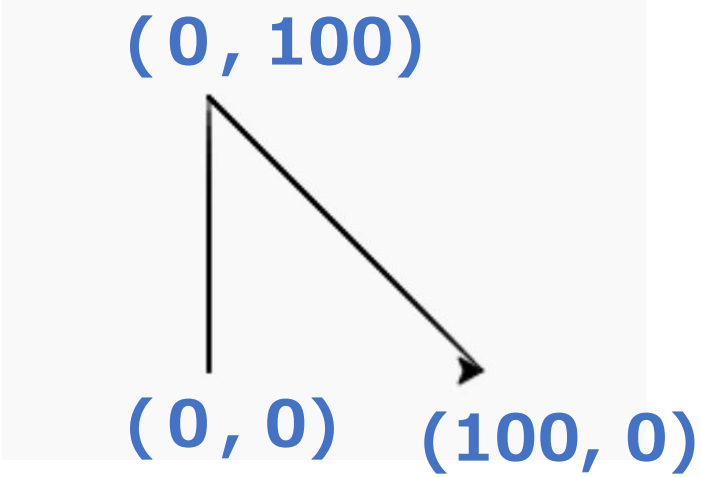
Python プログラムのファイル

タートルグラフィックス

カーソルを使って絵を描く

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0,100)
4  t.goto(100,0)
```

タートルグラフィックスの機能をインポートする「import turtle」が必要



タートルグラフィックス

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0,100)
4  t.goto(100,0)
```

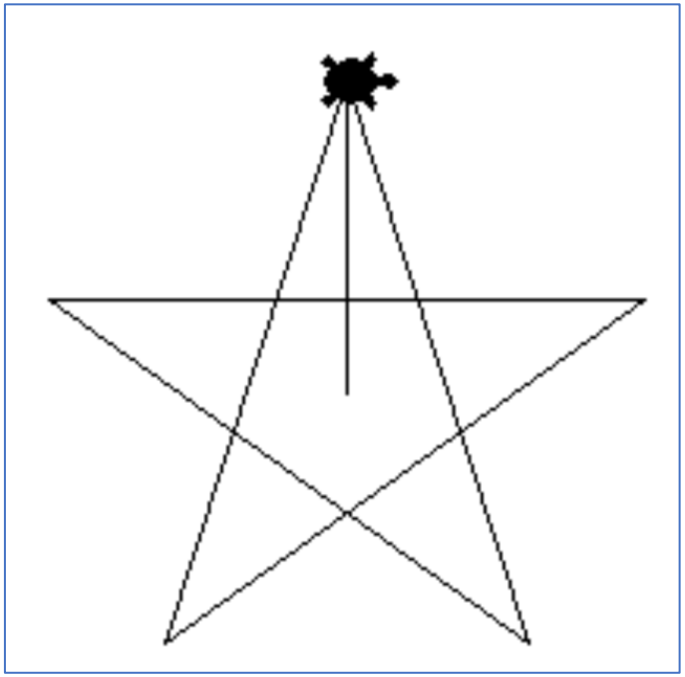
オブジェクト　メソッド

- **メソッド**は、オブジェクトが持つ機能を呼び出すためのもの
- 「**goto**」は**指定した座標への移動**

主なメソッド

- **goto**（＜横方向の値＞，＜縦方向の値＞）　**移動**
- **forward**(＜移動量＞)　**前進**
- **backword**(＜移動量＞)　**後退**
- **right**(＜角度＞)　**右回りに回転**
- **left**(＜角度＞)　**左回りに回転**

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0, 100)
4  t.goto(58, -80)
5  t.goto(-95, 30)
6  t.goto(95, 30)
7  t.goto(-58, -80)
8  t.goto(0, 100)
```



Python プログラム　　実行結果

基本を押さえる（パッケージ、オブジェクト生成、メソッド） ことで、プログラミングを楽しめるようになろう

Trinket のページ: https://trinket.io/python/5366def2f4

Python の基礎①　モジュール

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0, 100)
4  t.goto(58, -80)
5  t.goto(-95, 30)
6  t.goto(95, 30)
7  t.goto(-58, -80)
8  t.goto(0, 100)
```

- **モジュール**は、特定の機能を実現するための Python プログラムファイル
- 「**import turtle**」は**turtle モジュール**を現在のプログラムに**インポート**
- 「**turtle.Turtle()**」は**turtle モジュール内**の**Turtle** の機能を**呼び出し**

Python の基礎②　クラスとオブジェクト

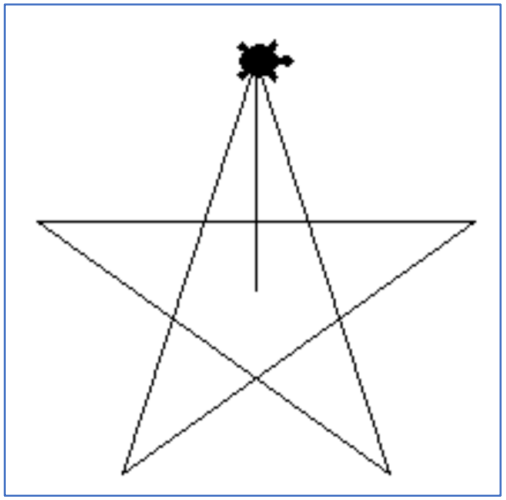
```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0, 100)
4  t.goto(58, -80)
5  t.goto(-95, 30)
6  t.goto(95, 30)
7  t.goto(-58, -80)
8  t.goto(0, 100)
```

- 「**turtle.Turtle()**」では、turtleモジュールの**Turtleクラス**の**オブジェクト生成**
- **Turtle クラスは**、図形の描画や移動などの操作の**機能**を実現するためのクラス

Python の基礎③　メソッド

```
1  import turtle
2  t = turtle.Turtle()
3  t.goto(0, 100)
4  t.goto(58, -80)
5  t.goto(-95, 30)
6  t.goto(95, 30)
7  t.goto(-58, -80)
8  t.goto(0, 100)
```

- **メソッド**は、オブジェクトが持つ機能を呼び出すためのもの
- 「**goto**」は**指定した座標への移動**



実行結果

演習
プログラミングはクリエイティブ

【構成】

①オブジェクト生成

②移動

③色、円

【トピックス】

- **モジュールのインポート**
- **オブジェクトの生成**
- **メソッド（移動）**

各自の自発的な演習、自己研鑽の時間

① **Pythonでグラフィックスを描く**

資料のプログラムを動かし理解を深める

② **Pythonの基本を押さえる**

オブジェクト、メソッド、引数

③ **発想力、創造力**

turtleモジュールを使用して、**あなた自身がデザインした図形を描く**。

④ **自主性、自己研鑽力、自分なりに工夫したこと**を振り返る

説明されなかった機能（他の図形の書き方）などを自主的に**調べ、理解し、自分で試してみる**。そして、**自分なりに工夫したこと**を振り返り、省察することで、さらに実力アップ。

trinket

- **Trinket** は**オンライン**の **Python、HTML** 等の**学習サイト**
- 有料の機能と無料の機能がある
- **自分が作成した Python プログラムを公開し、他の人に実行してもらうことが可能**（そのとき、書き替えて実行も可能）

- **Python の標準機能**を登載、その他、次のモジュールやパッケージがインストール済み

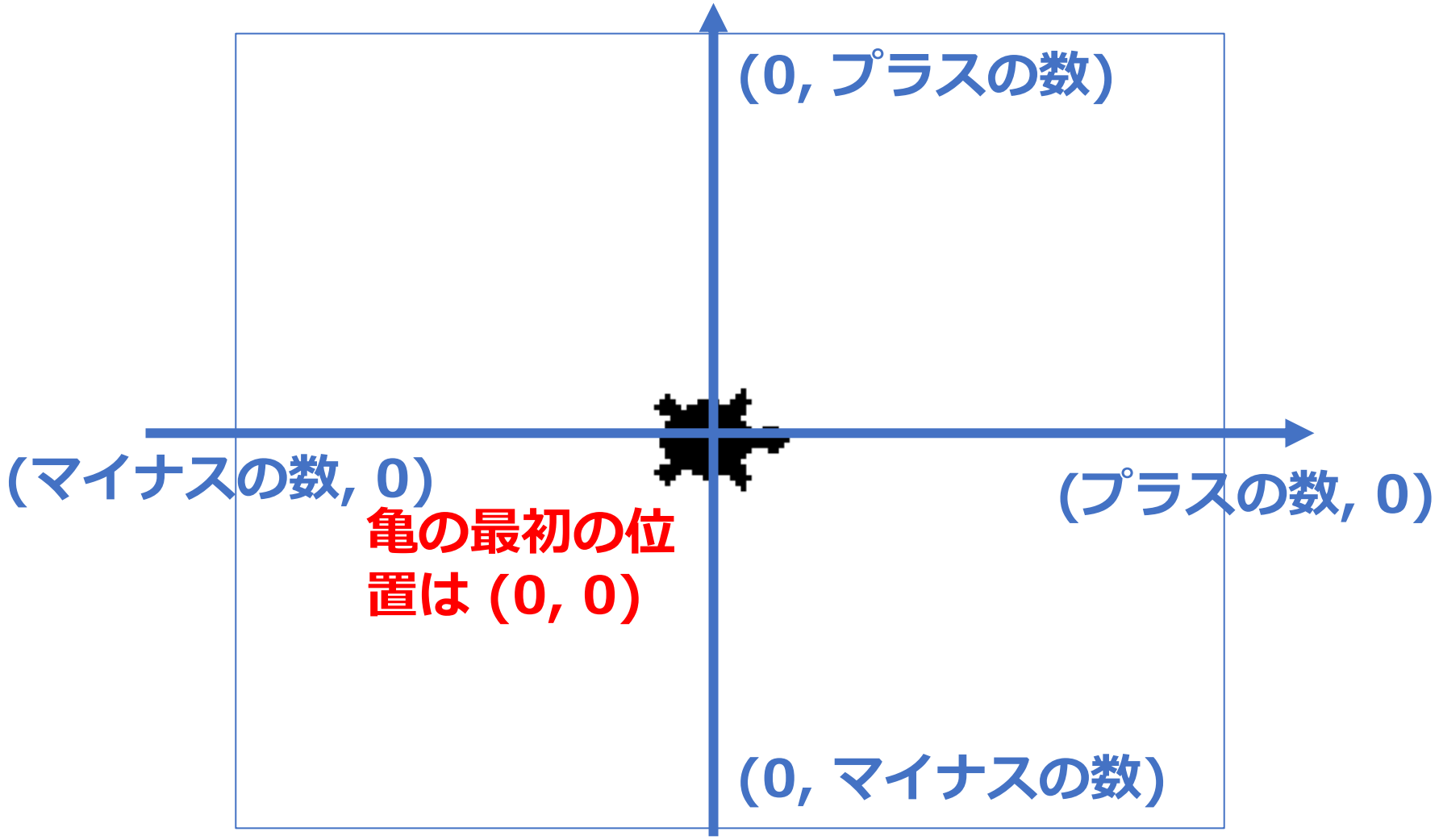
math, matplotlib.pyplot, numpy, operator, processing, pygal, random, re, string, time, turtle, urllib.request

trinket でのプログラム実行

- **trinket** は **Python, HTML などのプログラムを書き実行できる**サイト
- https://trinket.io/python/cdc4896571

のように、違うプログラムには違う URL が割り当てられる

- 実行が開始し直後のときは、「**実行ボタン**」で**実行**
- ソースコードを**書き替えて再度実行**することも可能


(0, プラスの数)
(マイナスの数, 0)
(プラスの数, 0)
亀の最初の位置は (0, 0)
(0, マイナスの数)

① 1つめ
https://trinket.io/python/f29bfe71cd

② 2つめ
https://trinket.io/python/5366def2f4

③ 3つめ
https://trinket.io/python/f8cd554693

①

```
import turtle
t = turtle.Turtle()
t.goto(0,100)
t.goto(100,0)
```

モジュールのインポート

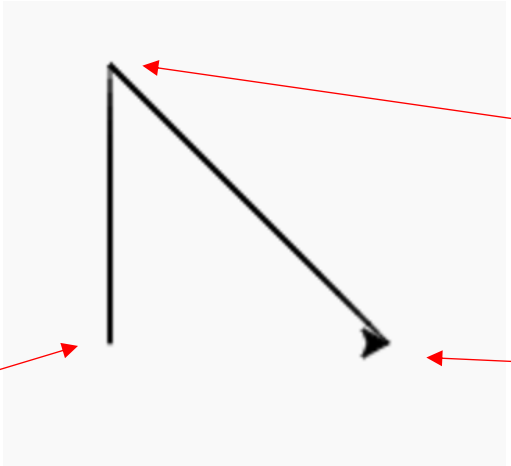
オブジェクト生成。t へのセット。

(0, 100) への移動

(100, 0) への移動

実行結果

最初の位置は
(0, 0)



②

```
import turtle
t = turtle.Turtle()
t.goto(0, 100)
t.goto(58, -80)
t.goto(-95, 30)
t.goto(95, 30)
t.goto(-58, -80)
t.goto(0, 100)
```

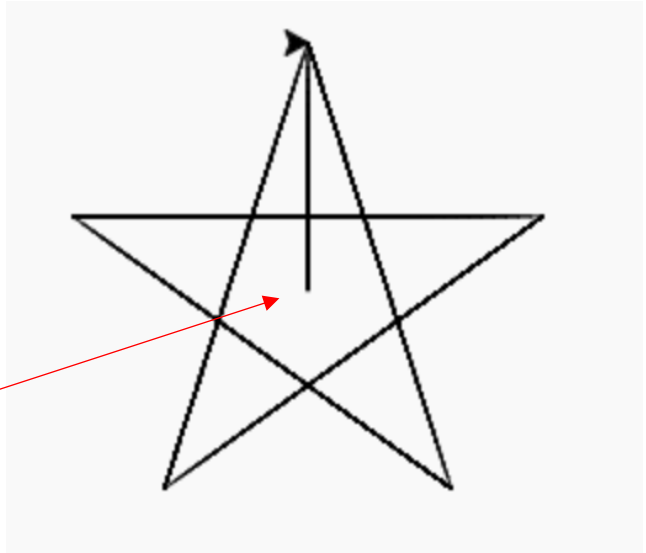
モジュールのインポート

オブジェクト生成。t へのセット。

移動

実行結果

最初の位置は
(0, 0)



③色、円

```
import turtle
t = turtle.Turtle()
colors = ["red", "green", "blue"]
for i in range(3):
    t.color(colors[i])
    t.circle(30)
    t.forward(50)
```

モジュールのインポート

オブジェクト生成。t へのセット。

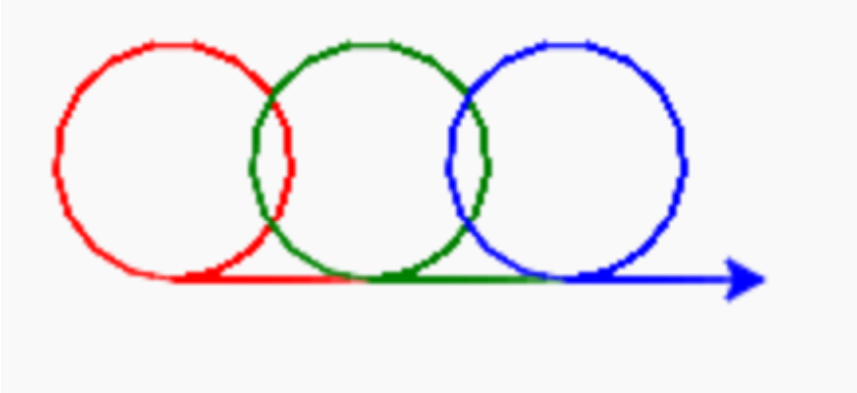
色は、赤、緑、青

色を変える

半径３０の円

前に50進む

実行結果



プログラミングまとめ

①卒業後は、ITエンジニアとして活躍しよう

②**基本的なプログラミングの知識**：モジュールの使用、オブジェクトの生成、メソッドの使用をおさえておくこと

③プログラミングの世界では、**失敗やエラーは成長のチャンス**になる。

④楽しみながらプログラミングに取り組もう。プログラミングは創造的な作業であり、自分のアイデアを形にすることができるもの。（「数学の問題を解く」ような勉強とは違う）

⑤**自分が興味を持つテーマやプロジェクト**を見つけ、プログラミングによる作品作りに**挑戦**してみよう。行動や体験が自信と成長につながる。

⑥**仲間**や**インターネット**、**先生**、対話型AI（**ChatGPT**など）から学び、教えることも重要。**共同作業やアイデアの共有**を通じて、**学び**、**新たな視点**を得ることができる

ChatGPT

- AI による対話システム
- 文章処理（要約、推敲、翻訳）のほか、問答により学び、新たな視点を得ることができる

- URL: https://chat.openai.com/
- 登録必要
- **AIの回答が間違っている場合がある**（必ず根拠を確かめよう）
- **秘密の情報やプライバシについて投稿してはいけない**
- 大学のレポートなどは、学生自身の成果物を求めているので、AIの回答をそのまま提出してはいけない

ChatGPT の登録手順（登録だけであればクレジットカード等は不要．スマホは必要）

①ChatGPT の公式ページにアクセス
https://chat.openai.com/

②「**Sign up**」をクリック

③ メールアドレスとパスワードを各自で設定

④「**Continue**」をクリック

⑤ ③のメールアドレスに確認メールが届くので、確認メール内の「**Verify email address**」をクリック

⑥氏名、誕生日、スマートフォンの電話番号を登録（画面は閉じないこと）

⑦⑥の**電話番号の SMS** に６桁の認証コードが届くので、画面に認証コードを入れる

⑧「**Verity**」をクリック

⑨ 使用目的を選択

ChatGPT との問答の例①　質問

KU　タートルグラフィックスで t.shape("turtle")を教わった。turtle 以外にあるか

はい、Pythonの`turtle`モジュールでは、`shape()`メソッドに指定できる形状がいくつかプリセットされています。これらは下記の通りです：

- "arrow"
- "blank"
- "circle"
- "classic"
- "square"
- "triangle"
- "turtle"

たとえば、"circle"や"triangle"を使って図形を描くことができます：

python　Copy code

```
import turtle

t = turtle.Turtle()

# Circle shape
t.shape("circle")
```

ChatGPT との問答の例②　プログラム作成を頼む

KU

次のプログラムを工夫して、花の形を描きたい　import turtle
turtle.Screen().clear()
t = turtle.Turtle()
t.shape("turtle")
t.goto(0,100)
t.goto(100,0)

以下のように、与えられたプログラムを工夫して花の形を描くことができます。

python

```python
import turtle

def draw_flower():
    window = turtle.Screen()
    window.bgcolor("white")

    # 花の色を設定
    colors = ["red", "blue", "purple", "yellow", "orange", "pink"]

    # Turtleを作成
    flower = turtle.Turtle()
    flower.shape("turtle")
    flower.speed(10)

    for _ in range(36):
        flower.color(colors[_ % len(colors)])

        # 花びらの描画
        flower.goto(0, 100)
        flower.goto(100, 0)
        flower.goto(0, -100)
        flower.goto(-100, 0)
        flower.goto(0, 100)
        flower.right(10)

    flower.hideturtle()
    window.exitonclick()

# プログラムの実行
draw_flower()
```

ChatGPT との問答の例③　プログラム内のミスを探すことを頼む

次は Python のタートルグラフィックスのプログラムです。間違いがあれば修正してください。import turtle
turtle.Screen().clear()
t = turtle.turtle()
t.shape("turtle")
t.goto(0,100)
t.goto(100,0)

プログラムの一部にいくつかの修正が必要です。以下に修正版を示します。

python　Copy code

```python
import turtle

screen = turtle.Screen()
screen.clear()

t = turtle.Turtle()
t.shape("turtle")
t.goto(0, 100)
t.goto(100, 0)
```

修正内容:

1. `turtle.turtle()` ではなく `turtle.Turtle()` を使用します。
2. `turtle.Screen().clear()` ではなく、`screen = turtle.Screen()` としてスクリーンオブジェクトを作成し、`screen.clear()` を呼び出します。

修正後のプログラムは、タートルグラフィックスのウィンドウを作成し、タートルを亀の形に設定して、座標 (0, 100) から座標 (100, 0) まで移動します。

ChatGPT との問答の例④　疑問に思ったことを相談する

Pythonの勉強をするときタートルグラフィックスで何を練習しますか

Pythonの勉強において、タートルグラフィックスは以下のような項目で練習することができます。

1. 基本的な動きの制御: タートルを前進させたり後退させたり、左右に回転させたりする方法を学びます。これにより、タートルがどのように動くのかを理解し、基本的なコマンドを使いこなすことができます。
2. 図形の描画: タートルを使用して、線や多角形、円などの図形を描画します。タートルがどのように動いて図形を描くのかを理解し、線の長さや角度を変えることで様々な形を作り出すことができます。
3. ループの活用: タートルグラフィックスを使って、繰り返し処理を行います。ループを活用することで、複雑なパターンや図形を効率的に描くことができます。
4. 条件分岐の応用: タートルが特定の条件に応じて動くように制御します。例えば、特定の位置に到達したら色を変える、特定の角度になったら方向を変えるなどの条件分岐を組み込むことができます。
5. アニメーションの作成: タートルの動きを連続的に更新することでアニメーションを作成します。ループや時間の概念を組み合わせることで、タートルが動く様子をリアルタイムで表示することができます。

これらの要素を組み合わせて、タートルグラフィックスを活用した様々なプログラムを作成することで、Pythonの基本的な構文や制御フローの理解を深めることができます。また、創造力を発揮して自分自身のアイデアを実現することもできます。

ChatGPT との問答の例⑤　勉強用に要点の作成を頼む

KU

Pythonの基礎を7行以内でまとめ、表形式で表示してください

以下にPythonの基礎を7行以内でまとめた表を示します。

項目	例
データ型	数値: `1`, `3.14` 文字列: `'Hello'`, `"Python"` ブール値: `True`, `False`
変数の代入	`x = 10` `name = 'Alice'`
演算子	算術演算子: `+`, `-`, `*`, `/` 比較演算子: `==`, `!=`, `<`, `>` 論理演算子: `and`, `or`, `not`
制御構造	条件分岐: `if 条件:` 繰り返し: `for 要素 in リスト:` ループの中断: `break` ループのスキップ: `continue`
リスト	`numbers = [1, 2, 3]` `fruits = ['apple', 'banana', 'orange']`
関数	関数の定義: `def 関数名(引数):` 関数の呼び出し: `関数名(引数)`
モジュール	モジュールのインポート: `import モジュール名` モジュールの機能の利用: `モジュール名.機能名`

この表にはPythonの基礎として代表的な要素がまとめられています。それぞれの項目には具体的な例が示されており、コードの一部を想像する助けとなるでしょう。ただし、Pythonの基礎を完全に網羅しているわけではありませんので、さらなる学習を進める際には詳細なリソースやドキュメントを参照することをおすすめします。

ChatGPT との問答の例⑥　勉強用に確認問題の作成を頼む

KU　Pythonの基礎を確認できる、下線補充の問題を 7 問作成してください。

1. データ型: 整数型の変数 `age` に値 `25` を代入するコードを書いてください。

```python
age = 25
```

1. 変数の代入: 文字列型の変数 `name` に値 `"John"` を代入するコードを書いてください。

```python
name = "John"
```

1. 演算子: 変数 `x` と `y` の和を計算し、結果を変数 `total` に代入するコードを書いてください。

```python
total = x + y
```

1. 制御構造: 条件式 `x > 10` が真の場合に `"x is greater than 10"` と表示する条件分岐のコードを書いてください。

```python
if x > 10:
    print("x is greater than 10")
```

1. リスト: 整数値 `1`、`2`、`3` を要素とするリスト `numbers` を作成するコードを書いてください。

```python
numbers = [1, 2, 3]
```

1. 関数: 引数 `name` を受け取り、`"Hello, name!"` というメッセージを表示する関数 `greet` を定義するコードを書いてください。

```python
def greet(name):
    print("Hello, " + name + "!")
```

1. モジュールの利用: `math` モジュールをインポートし、`math.sqrt()` 関数を使って変数 `x` の平方根を計算するコードを書いてください。

```python
import math

result = math.sqrt(x)
```

ChatGPT 利用上の注意点を再確認

- **AIの回答が間違っている場合がある**:

ChatGPTは、不正確な回答を提供する可能性がある。提供される回答は、参考にとどめ、**必ず他の信頼性の高い情報源などを確認**する。

- 秘密の情報やプライバシについての投稿:

個人情報や機密情報を投稿しないこと

- 自習などでの利用:

ChatGPTは、相談相手として、自習などに役立つ。しかし、**ChatGPT の回答をそのまま学校のレポートや宿題として提出してはいけません。**レポートや宿題は、学生自身の知識と理解を高めるためのもの。AIは参考として活用し、自分自身の考えや研究成果など、自分の成果物を提出することが大切。

４．説明

金子邦彦研究室
②AIを動作させる

画像分類　「犬」か「猫」か？

人工知能が、画像分類を行う。与えられた画像が特定のカテゴリに属するかどうかを判定するもの。

人工知能エンジニアのスキル

人工知能エンジニアは、**多様な幅広いスキル**を活用

- 機械学習、ディープラーニング（ニューラルネットワーク、CNN、Transformer など）
- プログラミング（Python、関連スキル）
- データの前処理や選別、データサイエンス
- 倫理的な理解（AIの社会への影響、プライバシーなど）
- ソフトスキル（チームで働く能力、スケジュール管理能力、コミュニケーション力、説明力）

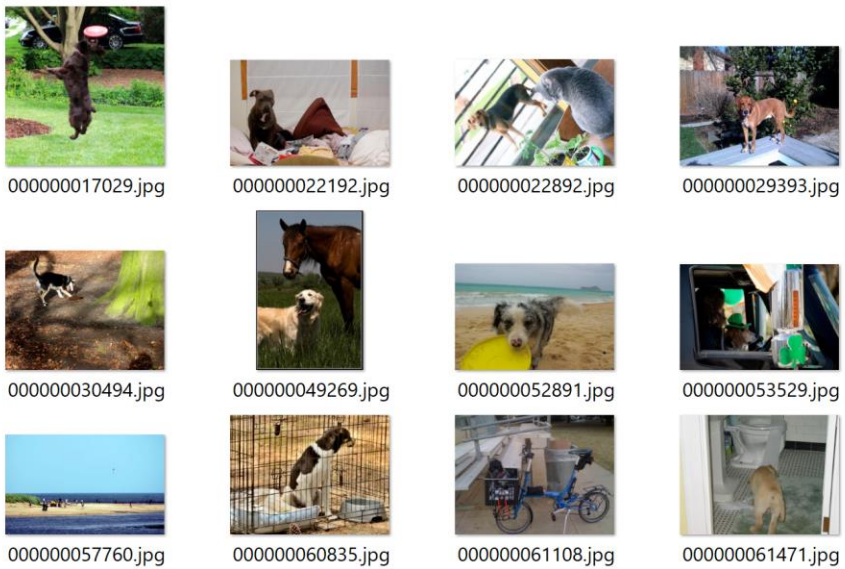
人工知能を制作し実行するためのプログラミングスキル

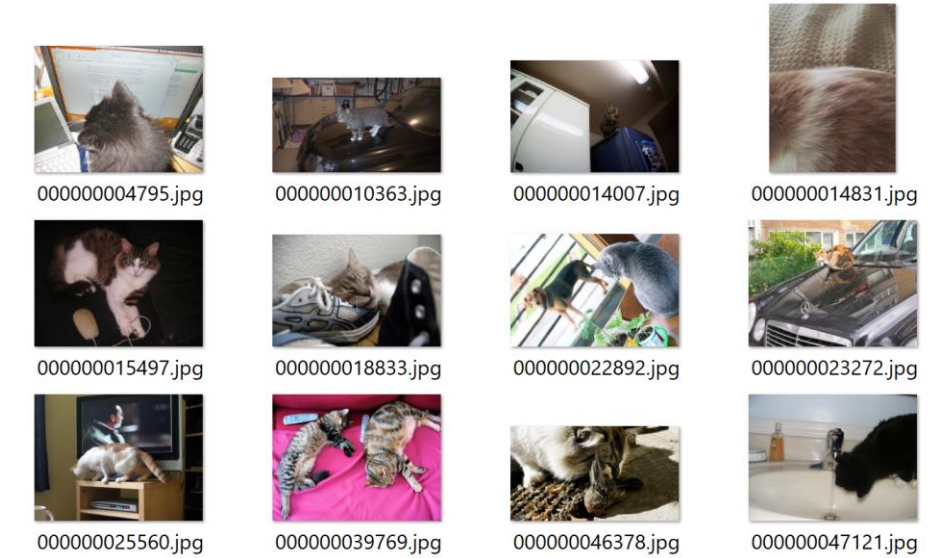
- **プログラミングは、基本を押さえる**（パッケージ、オブジェクト生成、メソッド、変数、関数、if、for）
- **関連スキル**を必要に応じて自主的に学ぶ
  - **システムデザイン**: ソフトウェアのアーキテクチャと設計
  - **試作とテスト**: コードが意図した通りに動作することを確認。テスト駆動開発（TDD）など。
  - **バージョン管理**: gitのようなバージョン管理ツールで、プログラムの変更履歴を管理。
  - **クラウドの活用**: AWSなどのクラウドサービスを利用。アプリケーションのデプロイ。
  - **データベース管理**: SQLなどのデータベース言語。
  - **高速数値計算**: NumPyやPandasのようなライブラリを使って、大量のデータを効率的に処理。
  - **ビッグデータ処理**:大規模なデータセットを扱うHadoopやSparkなど。

人工知能プログラムの構成

1. 学習データの読み込みと前処理
2. モデルの構築
3. モデルのコンパイル
4. 学習
5. モデルの検証
6. タスク（人工知能に本来させたい処理）の実行

「犬」か「猫」かの判別のために学習データを用いる





１７６の犬画像と１８３の猫画像で
人工知能が学習

人工知能エンジニアは「データから新しい価値を創造できる能力」を持つ

演習
カラー写真の犬と猫の分類

【構成】

①プログラムを実行

②ソースコードの確認

【トピックス】

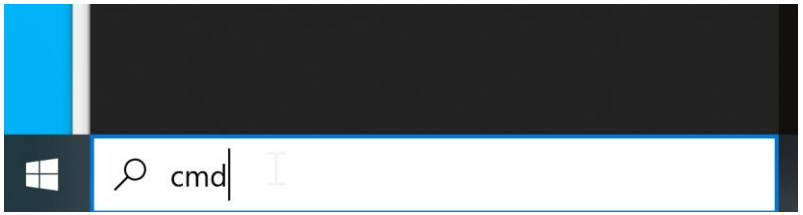
- 人工知能プログラムの構成：１．学習データの読み込みと前処理、２．モデルの構築、３．モデルのコンパイル、４．学習、５．モデルの検証、６．タスクの実行
- プログラミングの基本：パッケージ、オブジェクト生成、メソッド、変数、関数、if、for

Spyderの起動

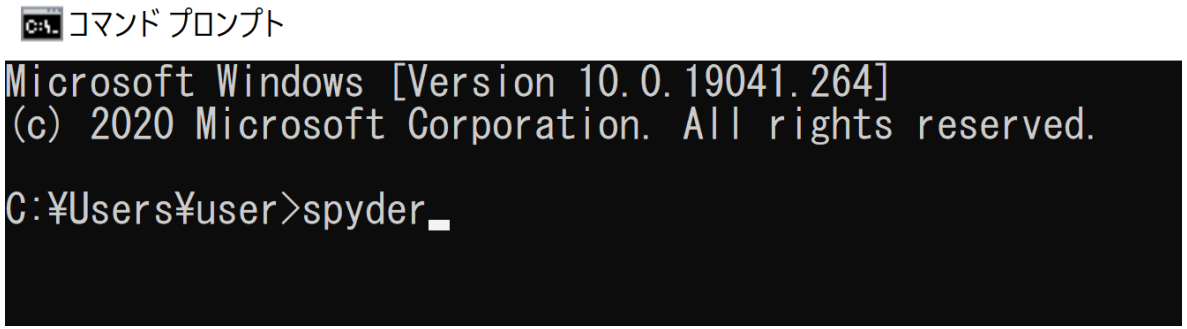
Spyder は Python の開発環境

起動

①　コマンドプロンプトを開く（検索窓で **cmd** が便利）

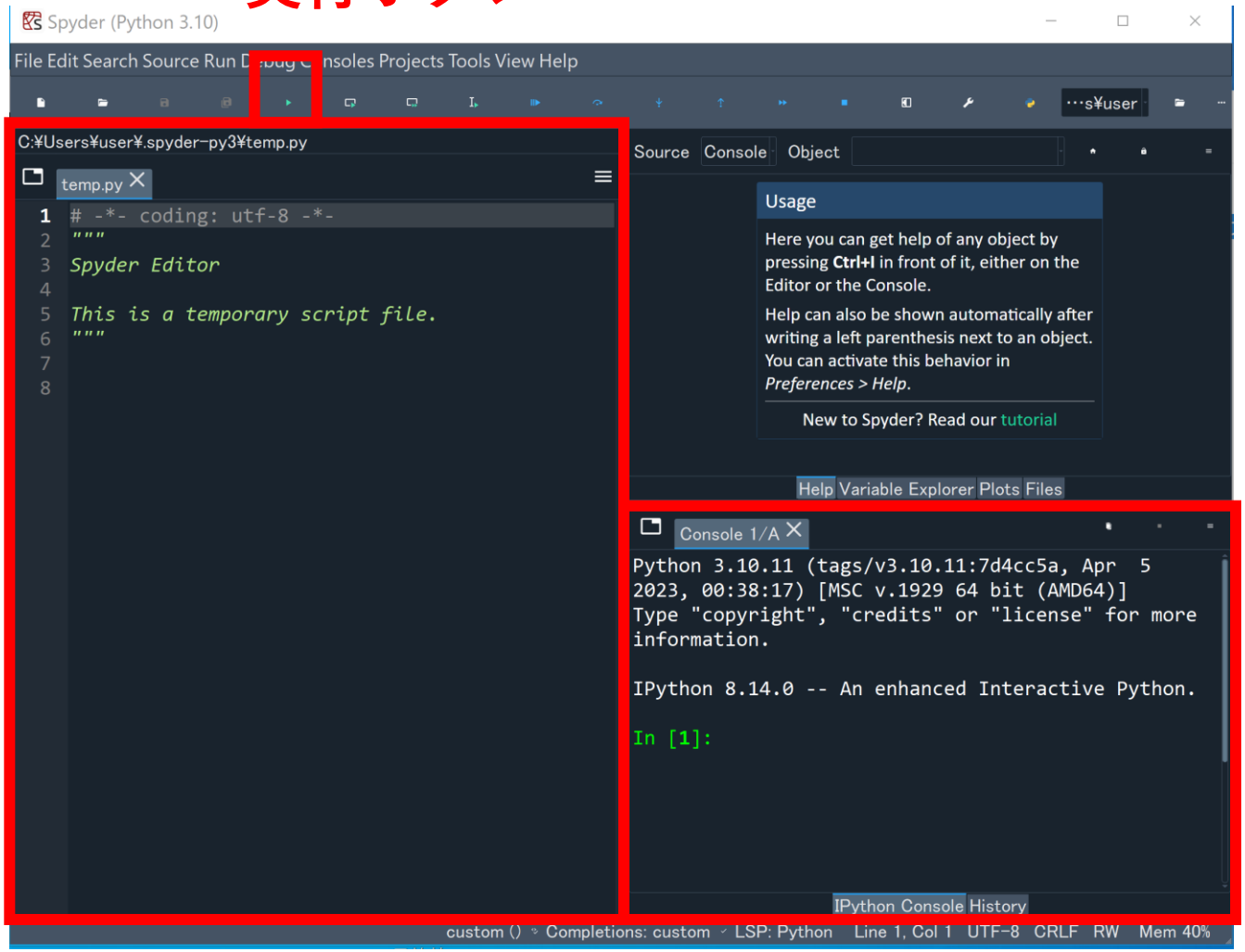


②　コマンドプロンプトで「**spyder**」，Enter キー



Spyder の画面

実行ボタン



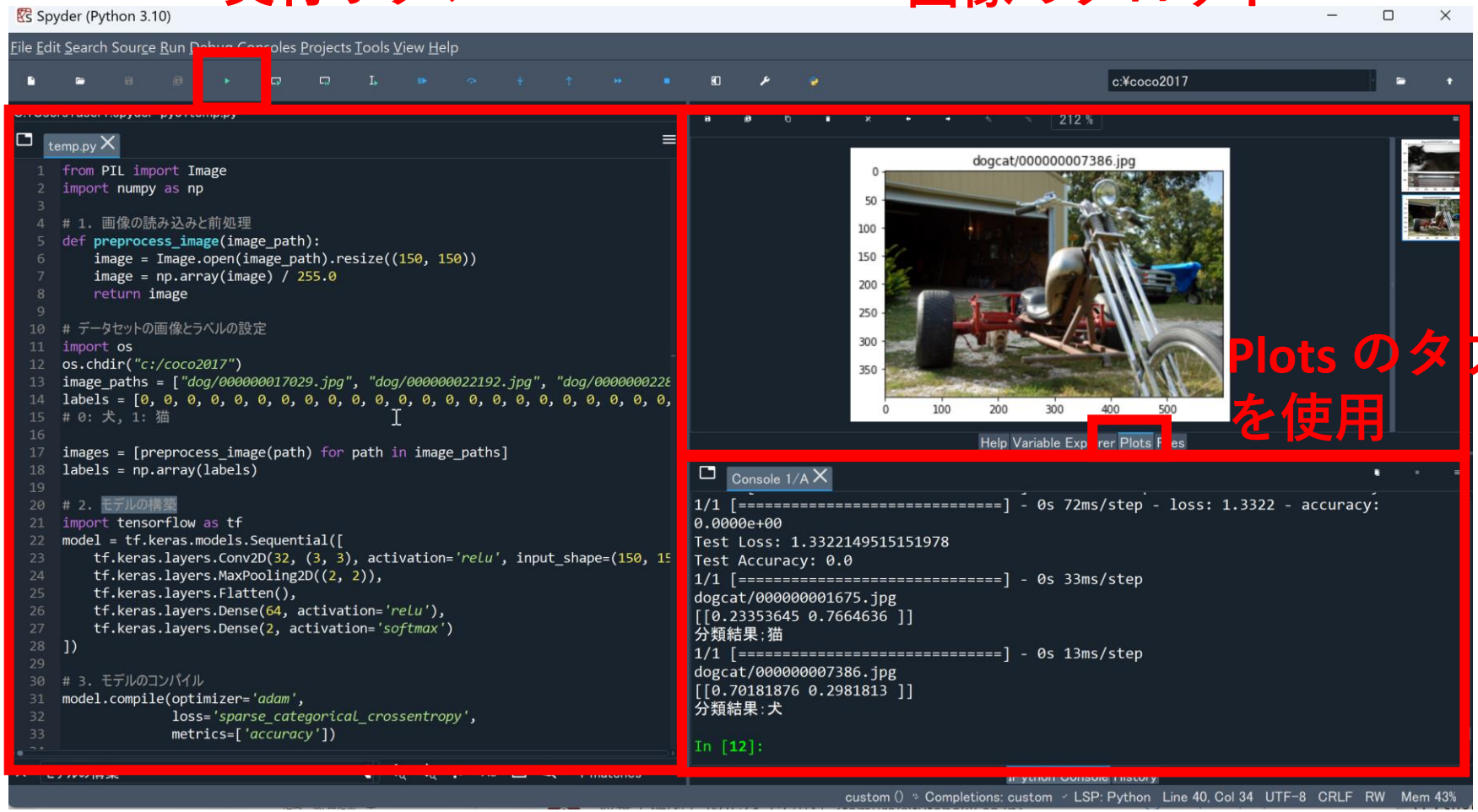
編集

コンソール

① プログラムを実行

実行ボタン

画像のプロット



Plots のタブを使用

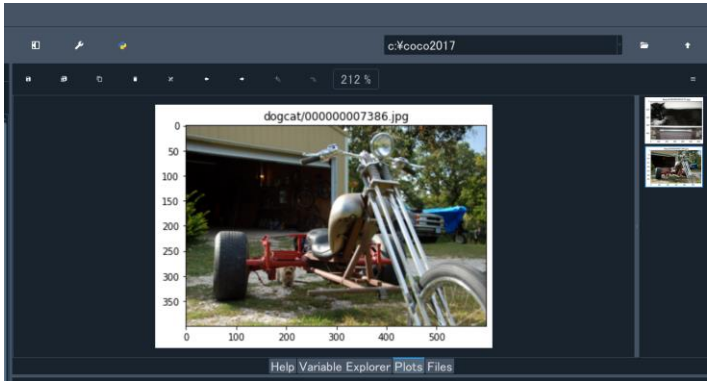
編集

コンソール

```
1/1 [==============================] - 0s 33ms/step
dogcat/000000001675.jpg
[[0.23353645 0.7664636 ]]
分類結果:猫
1/1 [==============================] - 0s 13ms/step
dogcat/000000007386.jpg
[[0.70181876 0.2981813 ]]
分類結果:犬
```

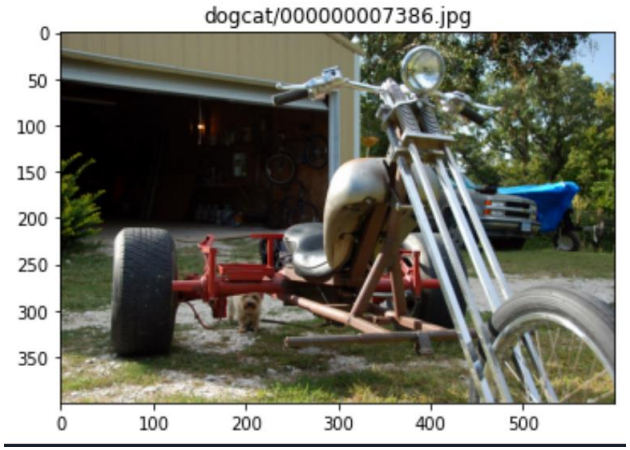
コンソール

１枚目の画像　**犬**である確率**２３％**
　　猫である確率**７７％**
２枚目の画像　**犬**である確率**７０％**
　　猫である確率**３０％**
（乱数を使っているので実行のたびに結果が変わる）



←選択可能

右上の画面に画像表示



１枚目の画像

２枚目の画像

②ソースコードの確認

```
# 1. 画像の読み込みと前処理
def preprocess_image(image_path):
    image = Image.open(image_path).resize((150, 150))
    image = np.array(image) / 255.0
    return image

# データセットの画像とラベルの設定
import os
os.chdir("c:/coco2017")
image_paths = ["dog/000000017029.jpg", "dog/000000022192.jpg", "dog/0000000228
labels = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
# 0: 犬, 1: 猫

images = [preprocess_image(path) for path in image_paths]
labels = np.array(labels)

# 2. モデルの構築
import tensorflow as tf
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(150, 15
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(2, activation='softmax')
])
```

□ モジュールやパッケージ

□ 関数やメソッドやクラス名

続き

```
# 3. モデルのコンパイル
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# 4. 学習
model.fit(np.array(images), labels, epochs=5)

# 5. モデルの評価
test_images = [preprocess_image(path) for path in ["dogcat/000000001675.jpg", "d
test_labels = np.array([0, 1])  # テスト用のラベル
test_loss, test_accuracy = model.evaluate(np.array(test_images), test_labels)
print("Test Loss:", test_loss)
print("Test Accuracy:", test_accuracy)

# 6. タスクの実行(学習済みモデルを使用して新たな画像に対して予測を行う)
import matplotlib.pyplot as plt
for test_image_path in ["dogcat/000000001675.jpg", "dogcat/000000007386.jpg"]: #
    test_image = preprocess_image(test_image_path)
    test_image = np.expand_dims(test_image, axis=0)  # 予測のために次元を追加
    predictions = model.predict(test_image)
    image = plt.imread(test_image_path)
    plt.imshow(image)
    plt.title(test_image_path)
    plt.show()
    print(test_image_path)
    print(predictions)
    predicted_label = np.argmax(predictions[0])
    if predicted_label == 0:
        print("分類結果:犬")
    else:
        print("分類結果:猫")
```

□ モジュールやパッケージ

□ 関数やメソッドやクラス名

人工知能による画像認識

画像認識でできることはたくさんある

- 画像分類
- 物体検出
- 顔検出
- 表情推定
- 文字検出
- セグメンテーション
- 追跡
- 姿勢推定

など

人工知能まとめ

- **画像分類**: 与えられた画像が「犬」か「猫」かを判定するなど
- **人工知能エンジニアのスキル**: 機械学習、ディープラーニング、プログラミング、データの前処理、倫理的な理解、ソフトスキル。
- **人工知能制作と活用のためのプログラミングスキル**: 基本の押さえ、関連スキルの自主学習（システムデザイン、試作とテスト、バージョン管理、クラウドの活用、データベース管理、高速数値計算、ビッグデータ処理)
- **人工知能プログラムの構成**: 学習データの読み込みと前処理、モデルの構築、モデルのコンパイル、学習、モデルの検証、タスクの実行。

①情報工学における創造力と未来志向

- 未来の技術（AI、データベース、３次元など）に対する理解や関心や探求心
- 将来の多様なキャリアパスへの視野（IT企業、製造業、クリエイティブな職種など）

②情報工学演習I、IIと、卒業研究による学習の深化

・自己選択による研究テーマの探求と深い学び

・専門知識、ITスキル、自己成長力、調査力、ロジカルシンキング、創造力、問題解決力の強化

③プログラミングとAIの理解と適用

- 自分のアイデアを形にする楽しさと達成感
- プログラミングの基本（パッケージ、インポート、オブジェクトとメソッド、クラス、オブジェクト生成）
- 画像分類（例：「犬」か「猫」か？）のAIを構築・使用するスキル

④AIに対する理解と適用

AI（特に対話型AI、Chatbot）の概念とその作成方法

AIの社会的影響の理解、その正確性や情報倫理、情報漏洩の理解

卒業後の活躍：学生は多岐にわたるIT分野で活躍する準備が整う。情報工学の持つ未来の可能性について深い理解を持つことで、新たな技術やトレンドにも柔軟にたいおうできるようになる

付録

【セットアップ手順】
自宅等で試したい時のため
１．Python のインストール
２．コマンドプロンプトを管理者として実行し、次のコマンドを実行

```
python -m pip install -U pip setuptools
python -m pip install -U jupyterlab jupyter jupyter-console jupytext PyQt5
nteract_on_jupyter spyder
python -m pip install -U PythonTurtle
python -m pip install tensorflow==2.10.1 pillow matplotlib
mkdir c:¥coco2017
cd c:¥coco2017
curl -O https://www.kkaneko.jp/a/dog.zip
curl -O https://www.kkaneko.jp/a/cat.zip
curl -O https://www.kkaneko.jp/a/dogcat.zip
powershell Expand-Archive -DestinationPath . dog.zip
powershell Expand-Archive -DestinationPath . cat.zip
powershell Expand-Archive -DestinationPath . dogcat.zip
del enshu1.py
curl -O https://www.kkaneko.jp/a/enshu1.py
```

【ChatGPT の URL】
https://chat.openai.com/