1. 人工知能の歴史，種類，応用分野，Python の基礎

（ディープラーニング，Python を使用）

（全１５回）

https://www.kkaneko.jp/ai/ae/index.html

金子邦彦

CC BY NC SA

アドバイス

1. **簡単なことからスタート**し、**段階的に複雑な内容**へ
2. **自分自身の成長（さまざまな体験、知識の深まり、スキルの向上、視野の広がり）**を感じとる
3. 大事な**基礎**は**繰り返し**学習する
4. 知識を増やすため、パソコン操作を**辛抱強く**続ける
5. Pythonプログラミングへの**チャレンジ精神**を持つ
6. **計画的に行動**する：**授業はすべての回に出席**し、**課題はすべて提出**

①多様な技術の理解

②コンピュータとプログラミングの基礎知識

③AIとプログラミングの深い関連性

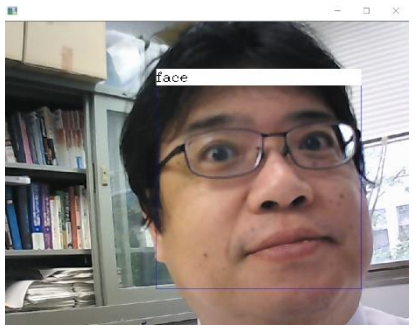
技術者としての能力を高め、新しい発見やアイデアの創出にもつながる

アウトライン

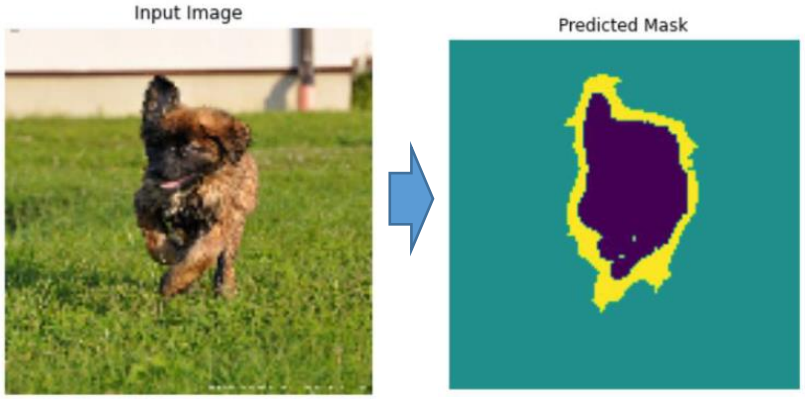
1. 人工知能の応用分野
2. 人工知能の種類，機械学習
3. ニューラルネットワークとその歴史
4. コンピュータを用いた問題解決、Python
5. 人工知能とプログラミングの関連性
6. 15回の全体計画（次のステップへ）

1-1. 人工知能の応用分野

人工知能の応用例



顔検知、顔識別



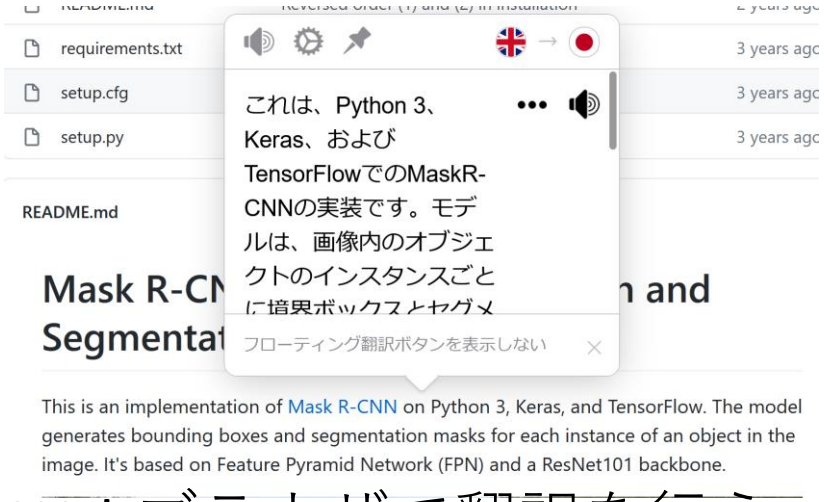
画像のセグメンテーション

金子の顔

有名人の声，表情，語り

金子がその有名人そっくりで語りだす

生成

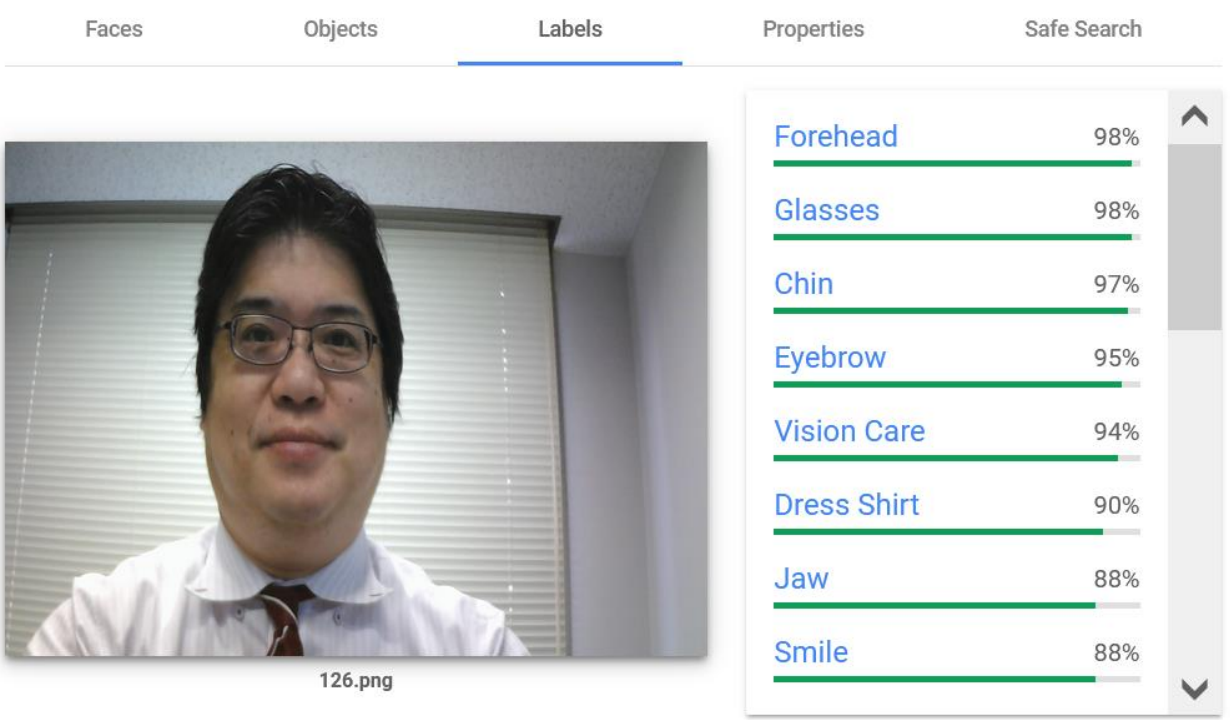


Webブラウザで翻訳を行う
Mate Translate (Web ブラウザ Firefox のアドオン)

画像分類を行うオンラインサービス

Web ブラウザで動く

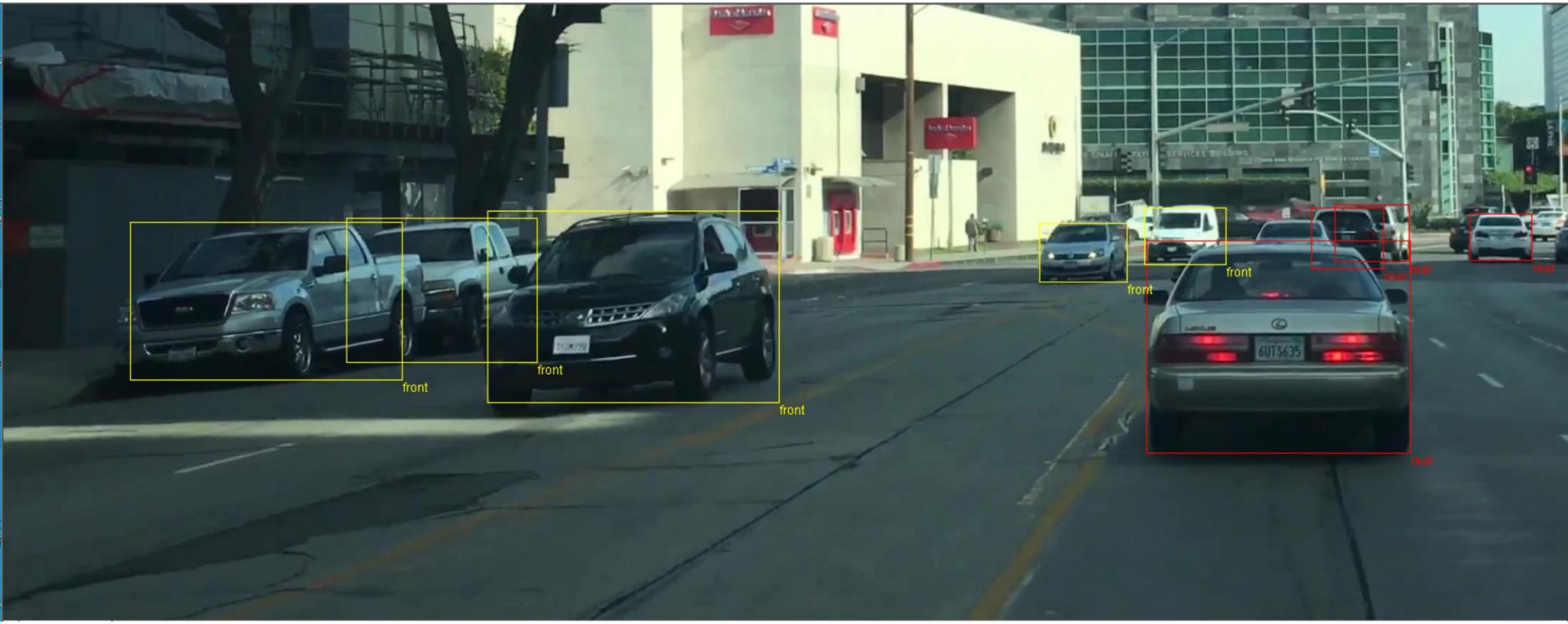
元画像



画像分類の結果

URL：https://cloud.google.com/vision/docs/drag-and-drop

車両の発見・検知



人工知能は，車両の場所と向き（前なのか後ろなのか）を素早く発見できるようになってきた
（Dlib を使用）

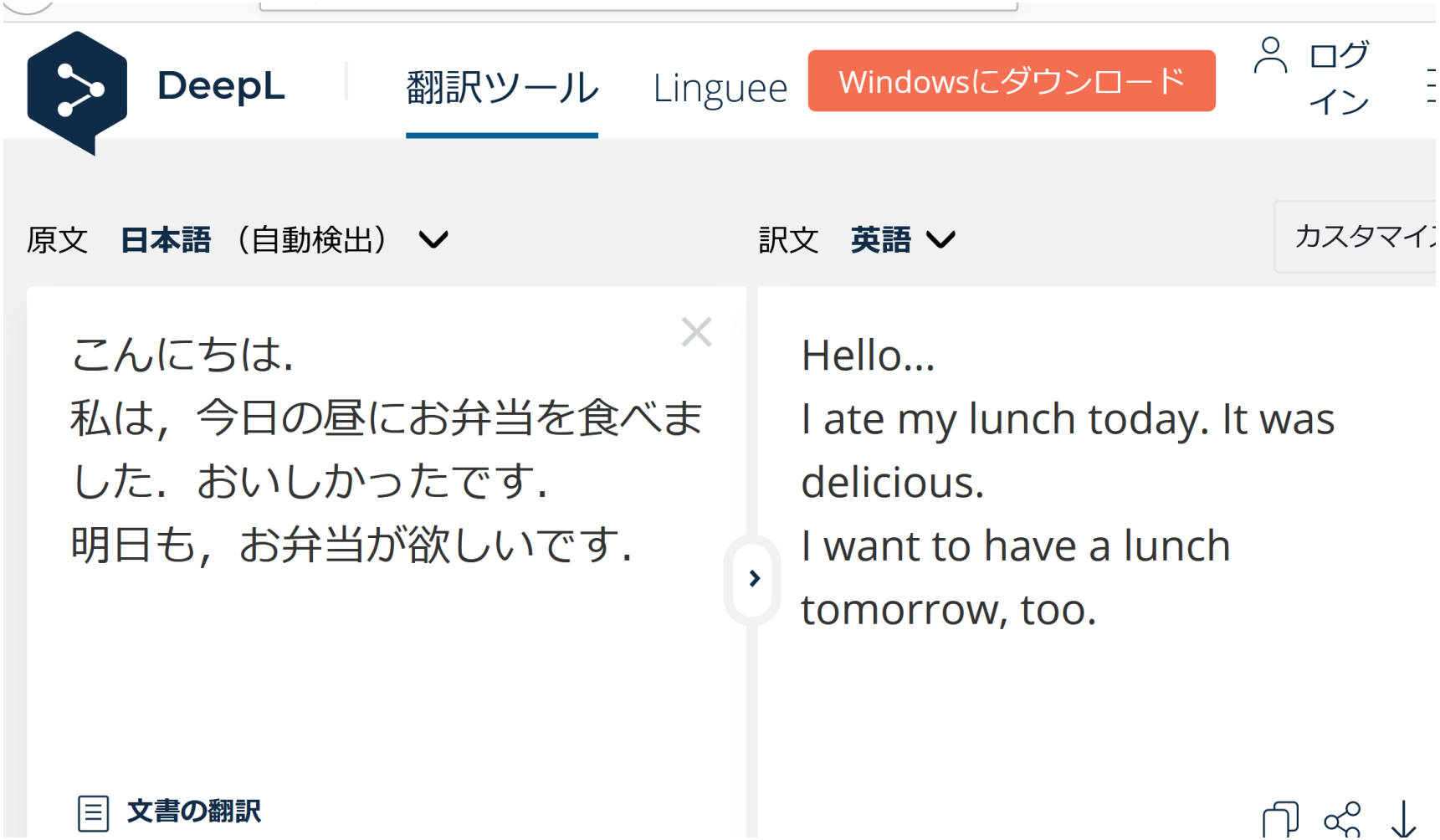
人や自転車などの，オブジェクトの発見・検知

元画像

人工知能による読み取り結果
(DeepLabv3+ を使用)

- 人間の「目」の一部機能をコンピュータで再現.
  画像の中のオブジェクトを，**人工知能**が発見・検知

人工知能による翻訳



https://www.deepl.com/

フェイクビデオ

写真

＋

ビデオ

→

人工知能により
生成されたビデオ

プロンプト（言葉での指示）による画像生成

人間が与えるプロンプト（言葉での指示）

beautiful garden, small house,
many flowers, blue sky, clouds,
realistic, cinematic, landscape vista
photography, Ghibli

人工知能が生成する画像

人工知能利用での倫理、注意点

人工知能による生成能力は目を見張るものがある。

《倫理》

- 人工知能を**悪用してはいけない**。
- 大学の課題などで，**人工知能が生成した文章や画像などを丸写ししてはいけない**

《注意点》

- 人工知能は**不正確な情報**を回答する場合がある
- 人工知能は**著作権に違反するコンテンツ**を回答する場合がある
- **プライバシや秘密にしたい情報を人工知能に与えない**

対話AIのサービス

ChatGPT の派生サービス（登録不要で無料の場合もある．期間限定で無料という場合もある）

TalkAI　https://talkai.info/ja/

翻訳、文章中の間違い探し、アイデア出し、まとめ、プログラム作成支援が得意

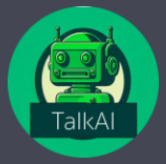
TalkAI の実行例①

- 間違い探しを人工知能に頼む

あなた

次の文章から誤字を探してください。

（基本原則）
第一条　私権は、公共の福祉に適合しなければなない。
２　権利の行使及び義務の履行は、信義に従い誠実に行わなければならない。
３　権利の濫用は、これを許さない。



ChatGPT

誤字はありません。ただし、第一条の「なない」という部分は「ならない」の誤りです。

TalkAI の実行例②

- プログラム作成のアシスタントとして利用

あなた

Javaで整数をキーボードから読み込み、それが4の倍数かどうかを調べたい

```
import java.util.Scanner;
public class MultipleOfFourChecker {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("整数を入力してください: ");
      int number = scanner.nextInt();
      // 4の倍数かどうかをチェック
      if (number / 4 = 0) {
         System.out.println(number + " は4の倍数です。");
      } else {
         System.out.println(number + " は4の倍数ではありません。");
      }
      scanner.close();
   }
}
```

ChatGPT

コードには間違いがあります。4の倍数かどうかを調べるためには、剰余演算子（%）を使う必要があります。

正しいコードは以下のようになります。

```
import java.util.Scanner;

public class MultipleOfFourChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("整数を入力してください: ");
        int number = scanner.nextInt();
        // 4の倍数かどうかをチェック
        if (number % 4 == 0) {
            System.out.println(number + " は4の倍数です。");
        } else {
            System.out.println(number + " は4の倍数ではありません。");
        }
        scanner.close();
    }
}
```

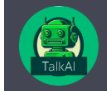
修正されたコードでは、4で割った余りが0であれば、入力された整数は4の倍数です。剰余演算子（%）は、左オペランドを右オペランドで割った余りを返します。

TalkAI の実行例③

- 家庭教師になってもらう

あなた

私は Python でゲームを作りたいのですが、やり方が分かりません。ゲームのプログラムの見本を見せてください



ChatGPT

以下に、簡単なコンソールベースの数当てゲームのPythonプログラム例を示します。

```
import random

def guess_number():
    number = random.randint(1, 100)
    tries = 0

    print("1から100の範囲内で、秘密の数を当ててください！")

    while True:
        try:
            guess = int(input("予想した数を入力してください: "))
            tries += 1

            if guess < number:
                print("もっと大きい数を予想してください！")
            elif guess > number:
                print("もっと小さい数を予想してください！")
            else:
                print(f"おめでとうございます！{number}が正解です！")
                print(f"あなたは{tries}回目で正解しました！")
                break

        except ValueError:
            print("正しい形式で数値を入力してください。")

guess_number()
```

このプログラムは、1から100の範囲内の整数を当てるゲームです。プレイヤーはコンソールに予想した数を入力し、プログラムは正解かどうかを判定し、ヒントを出力します。正解すると、正解までの試行回数が表示されます。

このように、Pythonでゲームを作るときは、まずゲームの仕様を決めて、それに基づいたプログラムを作成します。具体的なゲームの種類や要件に応じて、機能を拡張したり、グラフィックや音声を追加したりすることもできます。

演習
生成AI

ページ２０～２３

【トピックス】

- **翻訳サイト**
- **生成AIによる画像合成**
- **生成AIと対話する**

1．翻訳サイト DeepL による翻訳

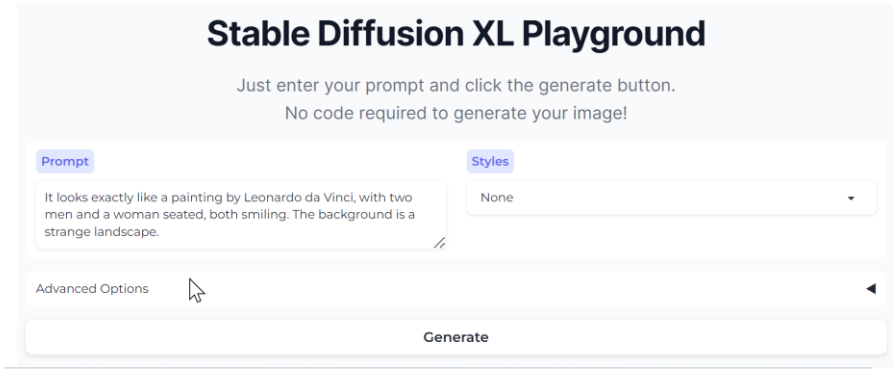
① 翻訳サイト DeepL

https://www.deepl.com/

② 右側に日本語の文章を入れると

左側に翻訳結果が出る

2．Stable Diffusion Playground による画像生成



① Stable Diffusion Playground のサイト

https://stablediffusionweb.com/

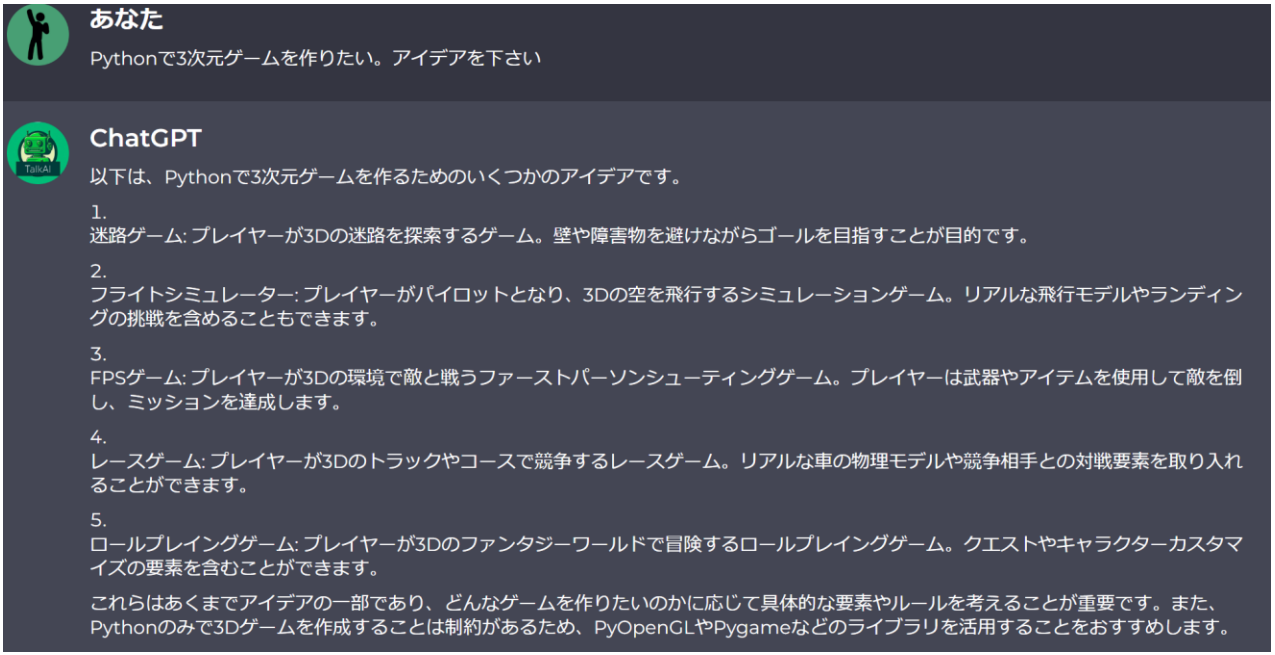
② 対話する。創造力、発想力を試してみる

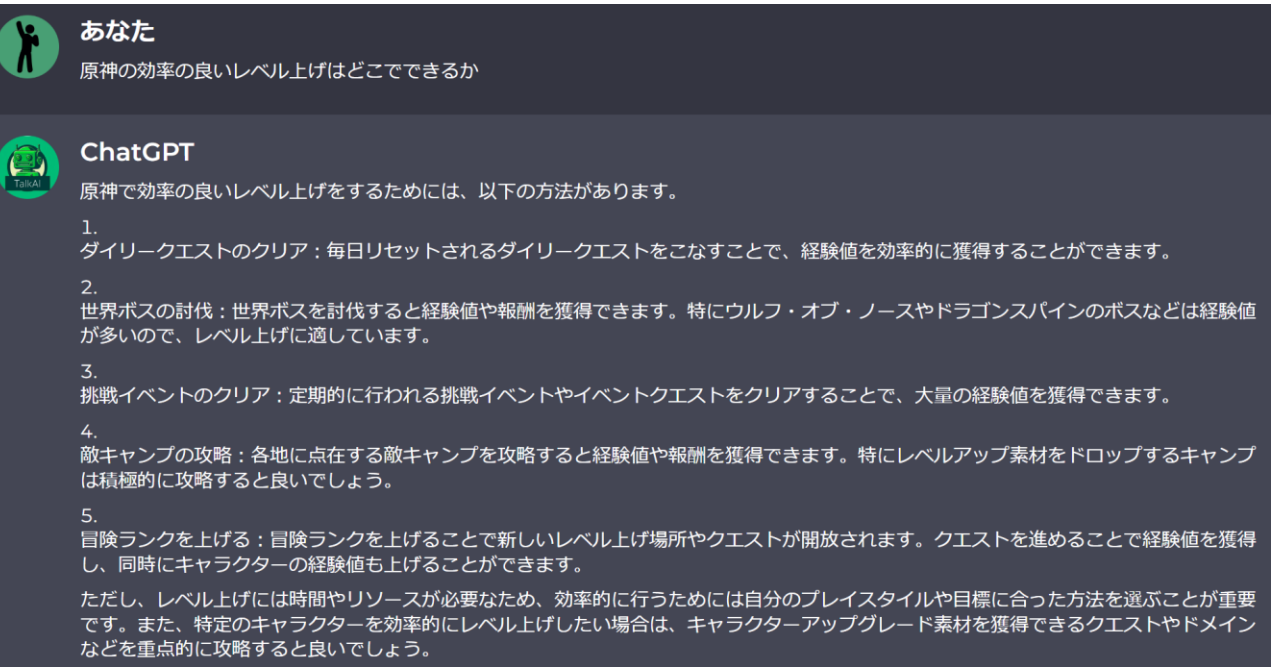
③ 「**Prompt**」の下に、プロンプトを**英語**で入れて、「Generate」をクリック

結果が出るまで**１分以上**待つ。

思い通りの結果を得るためにプロンプトを工夫する．プロンプトは具体的に。

3．TalkAI と対話する





① TalkAI のサイト

https://talkai.info/ja/

② **対話する。創造力、発想力、行動力のアシスタント**になってもらう

③今の時点では、調べごと、調べ学習のサポートは得意ではない

人工知能に期待される役割

- **人間の仕事の補助や代行。人間との協働。**

人工知能は、単純なタスクや反復的な作業など、人間にとって退屈な仕事を代行できる。

- **迅速な判断**

膨大な量のデータの処理、高速な計算が得意。人間のミスを減らすことにつながる。

- **新たな創造性や価値の創出の可能性**

人間の知覚を超えたセンサー、情報ネットワークの能力を組み合わせて、新たな分野の開拓や価値の創出が可能に。

- 人工知能の原理を探求することで、**人間の知性についての理解が深まる**ことも期待できる。

1-2. 人工知能の種類，機械学習

人工知能の種類

人工知能
（知的なITシステム）

機械学習

コンピュータが**データ**を使用して**学習**することより**知的能力を向上**させる技術

ルール，知識による人工知能

人間が**ルール，知識**を**コンピュータに与える**ことによる人工知能

機械学習の特徴

機械学習は，**コンピュータ**が**データ**を使用して**学習**することにより**知的能力を向上**させる技術

- **情報の抽出**：データの中からパターンや関係性を自動で見つけ出す能力
- **簡潔さ**：人間が設定しなければならなかったルールを、自動で生成できるようになる
- **限界の超越**：他の方法では難しかった課題でも、機械学習を用いることで解決策や視点を得られる可能性がある

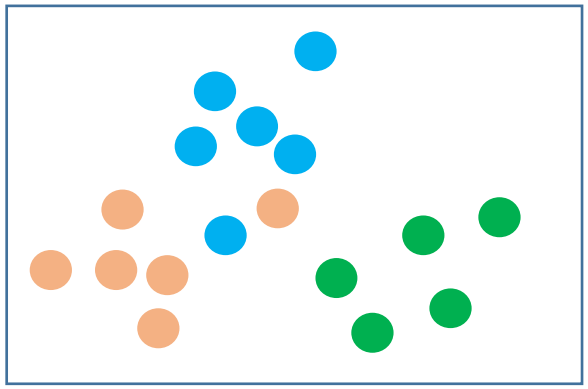
機械学習の特徴

- **データを用いた学習**： コンピュータは**訓練データ**を用いて**学習**し、その結果として知的能力が向上。
- **パターン認識**： **訓練データ**を基にして、**新しいデータ**に対する**パターン**や**関連性**を認識できるように。
- **自動化**： 学習によって得られた知的能力を用いて、**さまざまなタスクを自動実行**。
- **情報の抽出と予測**： 学習を通じて、新しいデータから**情報の抽出**、**予測**の能力を獲得
- **能力の進化**： **新しい訓練データが追加**されることで、知的能力のさらなる向上を期待できる

機械学習と訓練データ

機械学習は，**コンピュータ**が**データ**を使用して**学習**することより**知的能力を向上**させる技術

訓練データ



3 種類に分類済み

学習

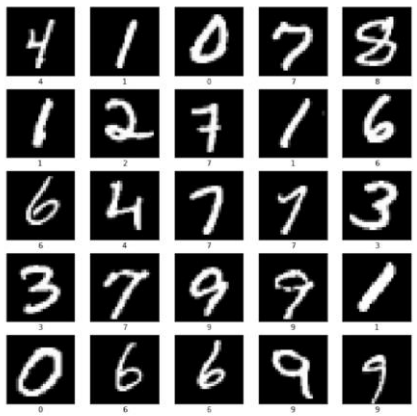
学習者

大量の訓練データを用いて
学習を行う

機械学習と訓練データとプログラム

機械学習のプログラム

学習に使用する訓練データ
（抜粋）



4 1 0 7 8
1 2 7 1 6
6 4 7 7 3
3 7 9 9 1
0 6 6 9 9

画像 60000枚
（うち一部）

正解 60000個
（うち一部）

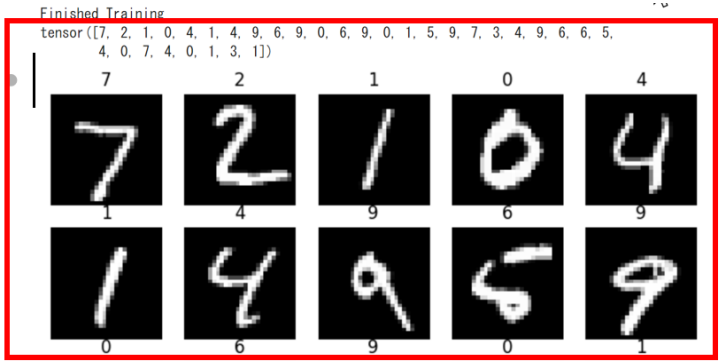
プログラム

データを用いて学習を行う
学習ののち，画像分類を行う

```
[4] !pip install -U scikit-learn matplotlib

    import torch
    import torch.nn as nn
    import torch.optim as optim
    from sklearn import datasets
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    import matplotlib.pyplot as plt

    # データの取得と前処理
    iris = datasets.load_iris()
    X = iris.data
    y = iris.target

    # データの標準化
    scaler = StandardScaler()
    X = scaler.fit_transform(X)

    # 訓練データとテストデータの分割
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    X_train = torch.tensor(X_train, dtype=torch.float32)
    y_train = torch.tensor(y_train, dtype=torch.long)
    X_test = torch.tensor(X_test, dtype=torch.float32)
    y_test = torch.tensor(y_test, dtype=torch.long)

    # ニューラルネットワークの定義
    class Net(nn.Module):
        def __init__(self):
            super(Net, self).__init__()
            self.fc1 = nn.Linear(4, 10)  # 入力は4次元（Irisの特徴量）
            self.fc2 = nn.Linear(10, 3)  # 出力は3クラス

        def forward(self, x):
            x = torch.relu(self.fc1(x))
            x = self.fc2(x)
            return x

    net = Net()
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(net.parameters(), lr=0.01)
```

学習の結果、文字認識の能力を獲得

```
Finished Training
tensor([7, 2, 1, 0, 4, 1, 4, 9, 6, 9, 0, 6, 9, 0, 1, 5, 9, 7, 3, 4, 9, 6, 6, 5,
        4, 0, 7, 4, 0, 1, 3, 1])
```



機械学習まとめ

機械学習の特徴

- データを用いて知的能力を向上
- 自動でデータのパターンを抽出
- さまざまなタスクを自動実行

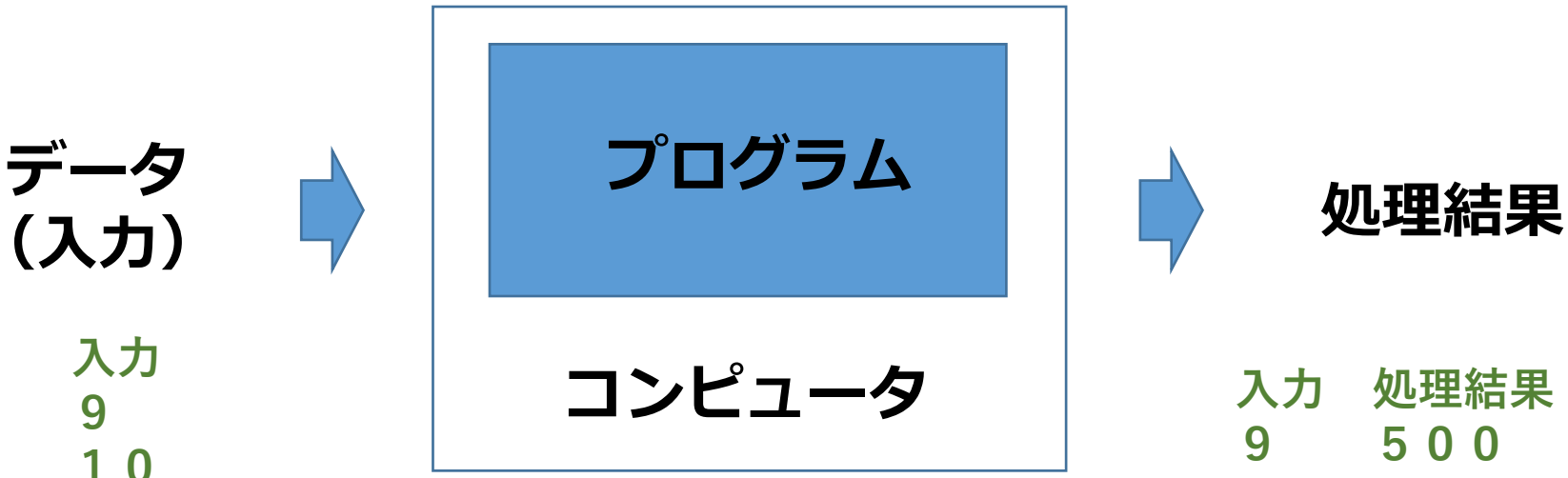
応用事例

画像理解、自然言語処理、予測など多数

機械学習の定義：

- **訓練データ**を用いて**学習**し、その結果として知的能力が向上
- 訓練データの追加により、さらに知的能力が向上する可能性

一般のプログラミング

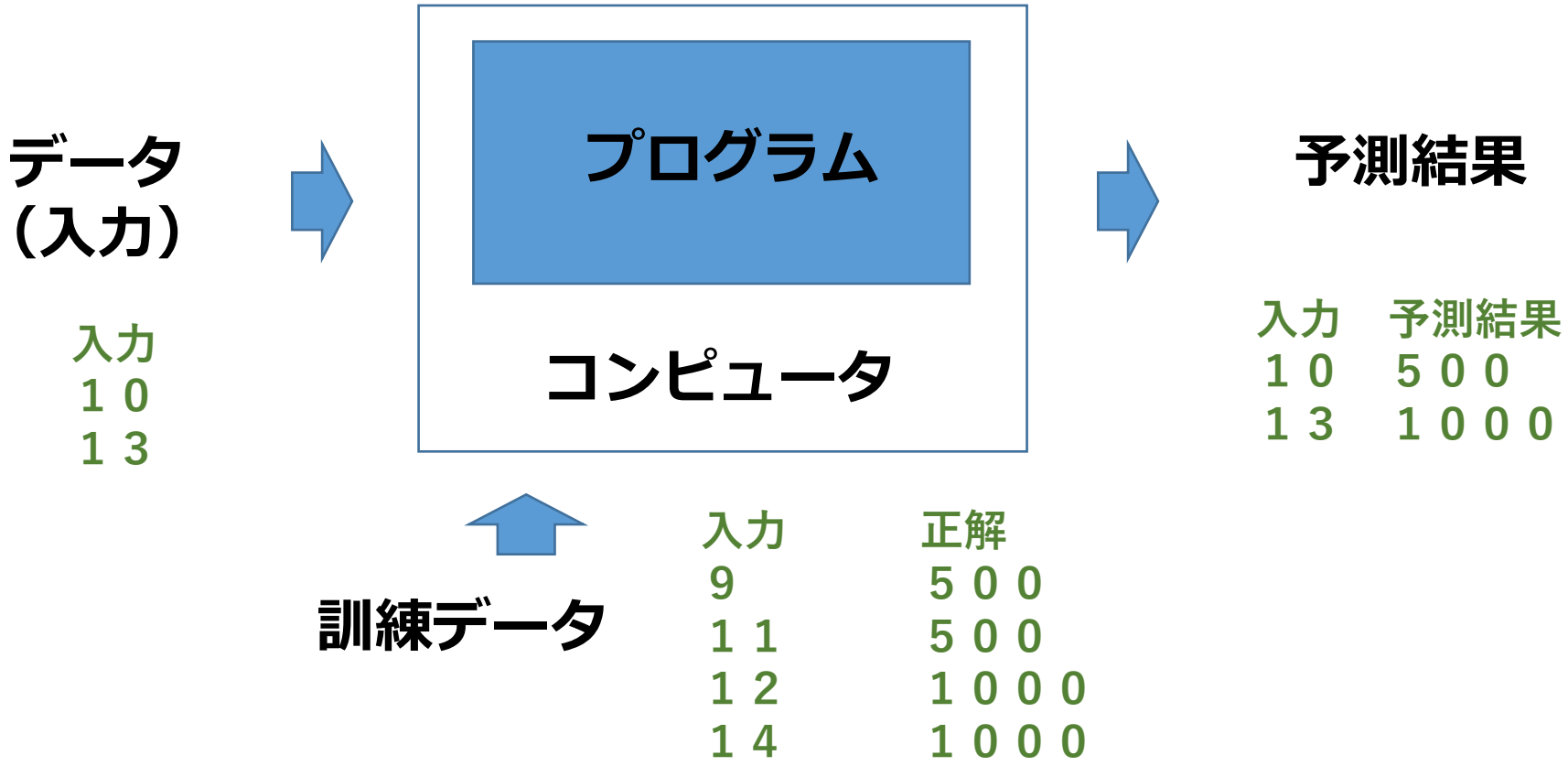


入力
9
10
11
12
13
14

入力	処理結果
9	500
10	500
11	500
12	1000
13	1000
14	1000

あらゆる入力について
正しい処理結果が得られるように，
プログラムを作成し，テストする

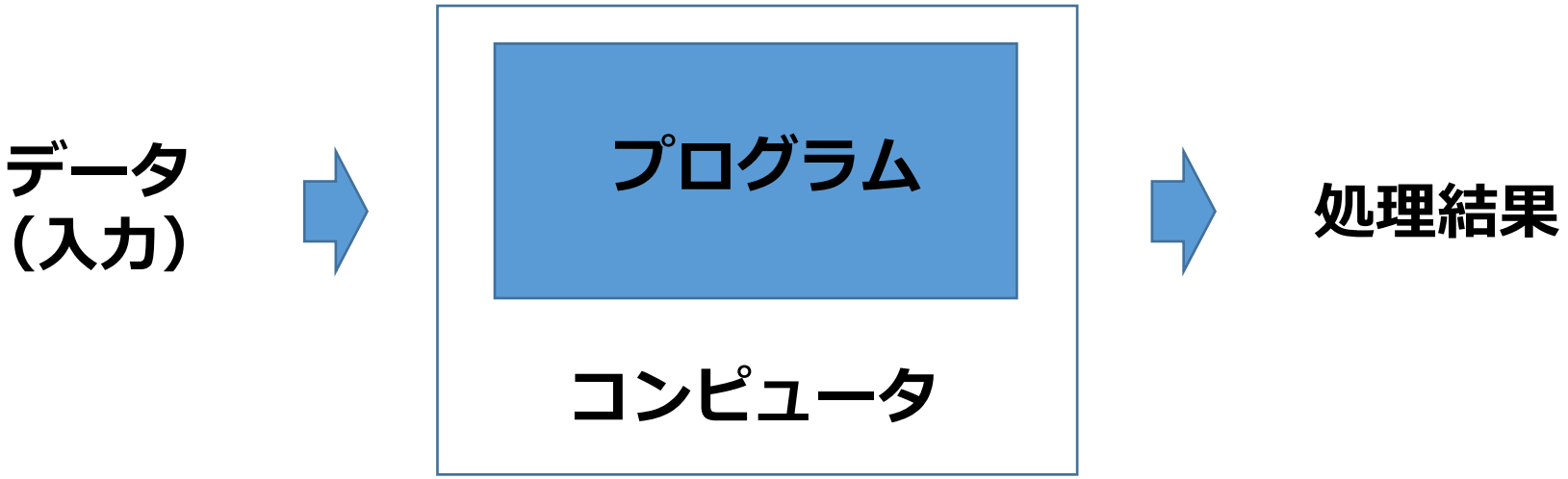
機械学習での予測



訓練データにより，学習が行われる。
新しいデータに対しては、自動で処理が行われる。

① 一般のプログラミング

・プログラムは人間が作成しテストし，調整する．



② 機械学習

・データを利用して知的能力を向上させる

汎化

- **汎化**は，**訓練データに基づいて学習**し，**未知のデータに対しても適切に処理**できる能力

- **機械学習**の一つの大きな特徴は、この**汎化能力を持つ**こと。

汎化

訓練データ

入力	正解
9	500
11	500
12	1000
14	1000

訓練データと未知のデータ

入力	予測結果
7	**500**
8	**500**
9	500
10	**500**
11	500
12	1000
13	**1000**
14	1000
15	**1000**
16	**1000**

訓練データに基づいて学習し，未知のデータに対しても適切に処理

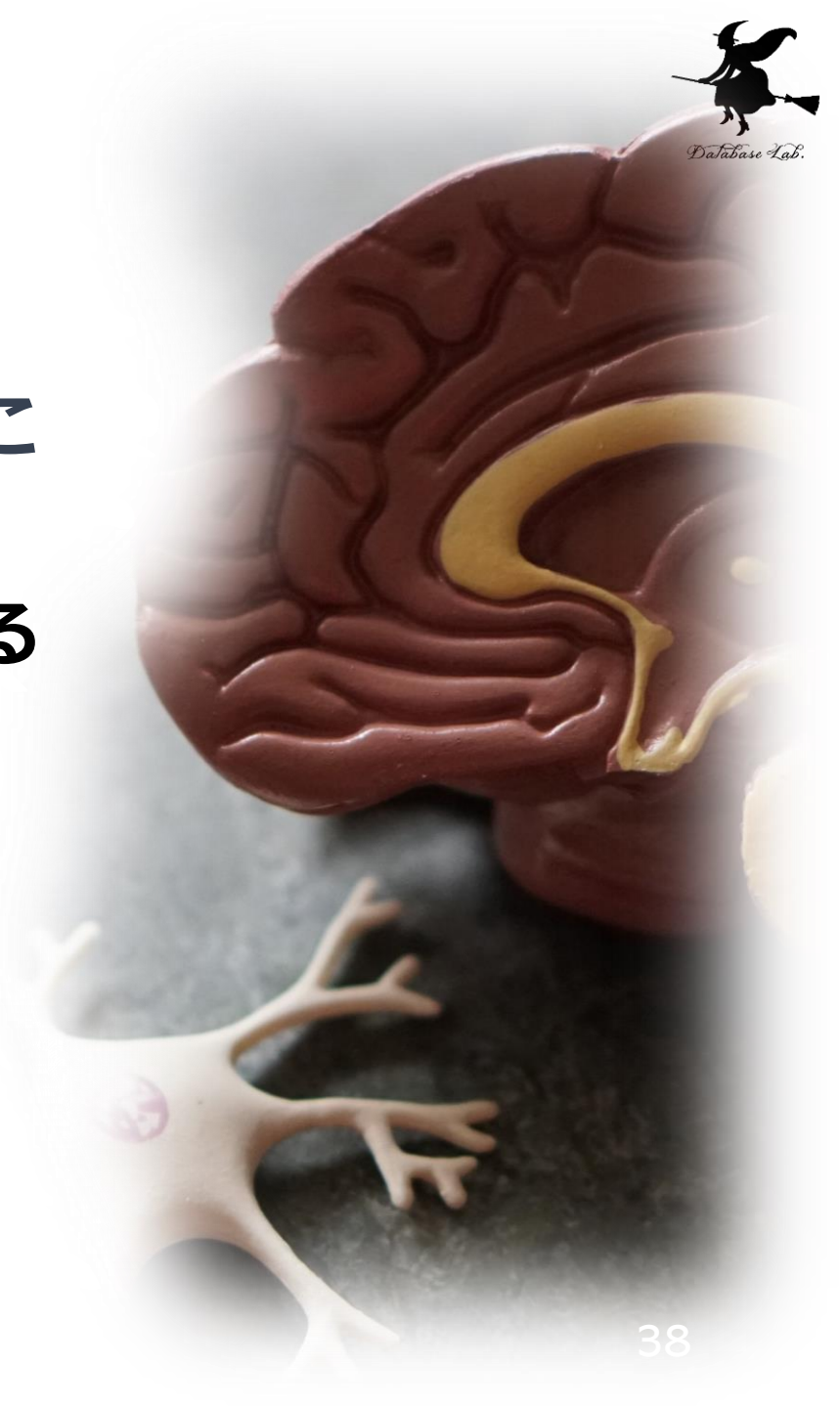
ここまでのまとめ

- **機械学習：**コンピュータは**訓練データ**を用いて**学習**し、その結果として知的能力が向上
- **汎化**は、**訓練データ**を基にして**学習**し、**未知のデータ**でも**適切に対応する能力**を指す
- **機械学習**の一つの大きな特徴は、この**汎化能力を持つ**こと
- **汎化**による結果は**常に正解とは限らない。**

1-3. ニューラルネットワークとその歴史

ニューラルネットワーク

- **ニューラルネットワーク**は、**人工知能**の一種
- **生物の脳の働きを模倣すること**を目指している
- **すでに広く活用が進んでいる**

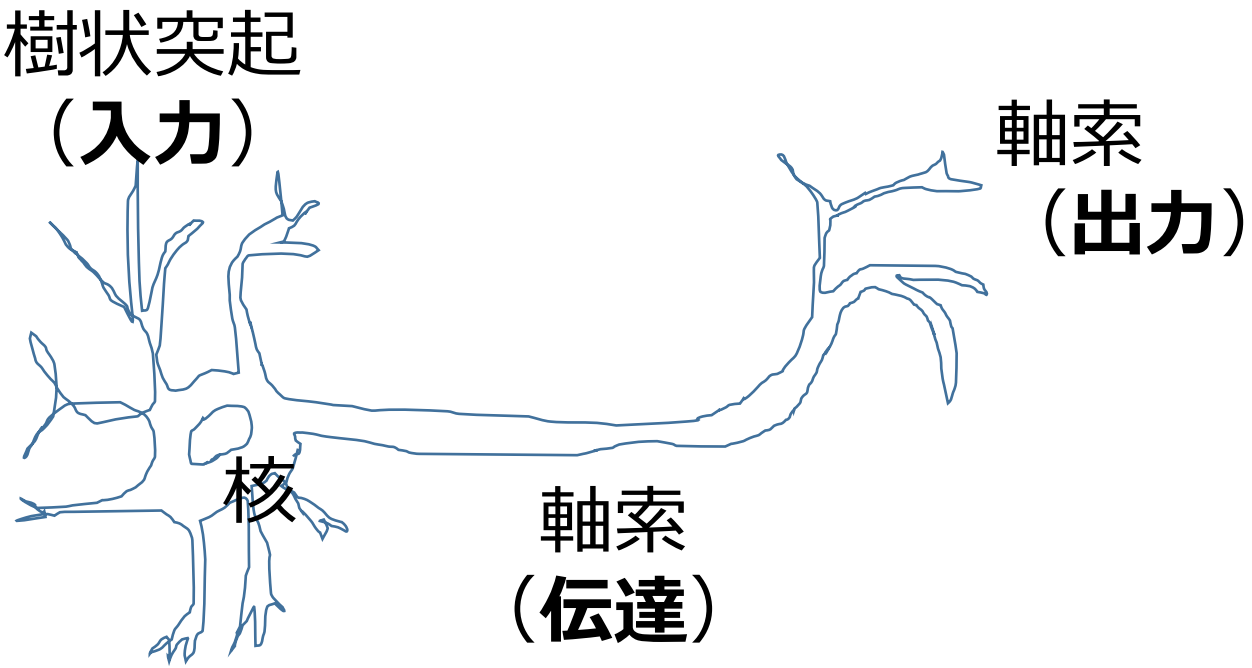


ニューラルネットワークとニューロン

- **ニューラルネットワーク**は，**脳や神経系のニューロン（神経細胞）**を参考に作られている
- **ニューロン**が**信号を受け取り，処理**し，**伝達**する**仕組みを模倣**している

ニューロン

脳や神経系の**ニューロン**は，生物の脳や神経系で、**情報伝達，記憶，情報処理**を行う．脳や神経系の大切な要素.



脳や神経系の研究の進展

1900年頃： **脳や神経系**は，**ニューロンの集まり**と考えられるようになった．ニューロンが情報伝達を行うと分かってきた.

1980年頃：生きた脳を対象とした測定が行われる、脳のどの領域がどのような機能を持つかを解明する**脳機能マッピングが進んだ**.

2010年頃：脳の異なる領域が連携して情報処理を行っていることが観察できるようになってきた

脳の仕組みと機能について**深い洞察**が得られるように

脳のニューロン

生物のニューロンの数（推定値）

- カタツムリ　11,000
- ロブスター　100,000
- アリ　250,000
- カエル　16,000,000
- ハツカネズミ 71,000,000
- タコ　500,000,000
- ネコ　760,000,000
- **ヒト　86,000,000,000**
- アフリカゾウ 257,000,000,000

Wikipeida の記事：
https://ja.wikipedia.org/wiki/%E5%8B%95%E7%89%A9%E3%81%AE%E3%83%8B%E3%83%A5%E3%83%BC%E3%83%AD%E3%83%B3%E3%81%AE%E6%95%B0%E3%81%AE%E4%B8%80%E8%A6%A7 より

- 生物の脳は**非常に多くのニューロン**から成り立つ
- 人間の脳には**約860億個**のニューロンが存在
- 複雑なネットワークを形成

ニューラルネットワークの歴史

1940年代： ニューラルネットワークの誕生

1980年代： バックプロパゲーション，ボルツマンマシン（機械学習の基本的な仕組み）

2010年代： 多層化，正規化，ReLU，ドロップアウトなど，ニューラルネットワークの新技術

（機械学習の問題である勾配消失や過学習の緩和）

コンピュータの進展とニューラルネットワーク進展

年	コンピュータで扱えるニューロン数の規模
2010年	100,000個
2020年	2,000,000個
2030年	50,000,000個
2040年	1,000,000,000個
2050年	20,000,000,000個

所説あります

2055年ごろには，860億を超えるかも？

【過去】

人間の脳の**約860億個**のニューロンを模倣することは不可能

【現状】

GPU，クラウドコンピューティングで，**コンピュータの性能向上が続く**

【将来】

生物の脳に近いニューラルネットワークの実現へ進む

1-4. コンピュータを用いた問題解決，Python

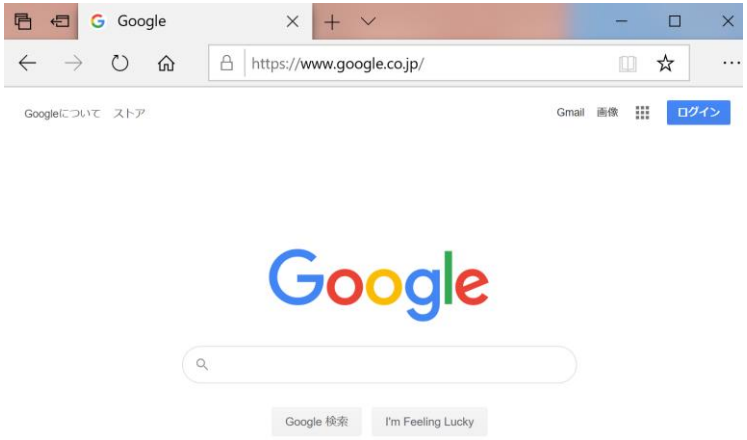
コンピュータとプログラム

- **コンピュータ**は，**プログラム**に従って動作
- **プログラム**は，**コンピュータ**に指示を出し，所定の作業を遂行させる

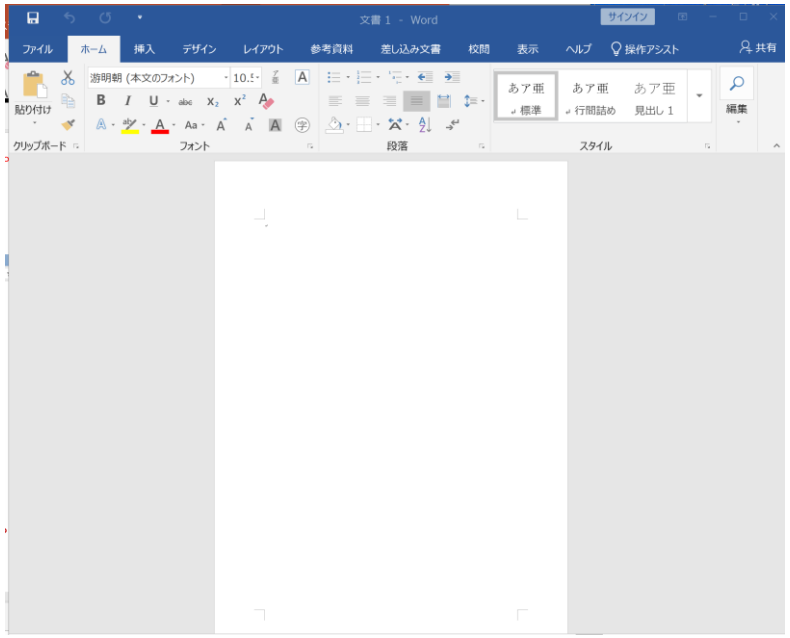
プログラミング

- **プログラム**を設計し作成するプロセス（プログラミング）は，**創造的な活動**
- **アイデアを形にできる**ことが，**プログラミング**の魅力

① プログラムとアプリケーション



Web ブラウザ



ワープロ
（マイクロソフト・ワード）

プログラムが動作し，**アプリケーション**の機能を実現

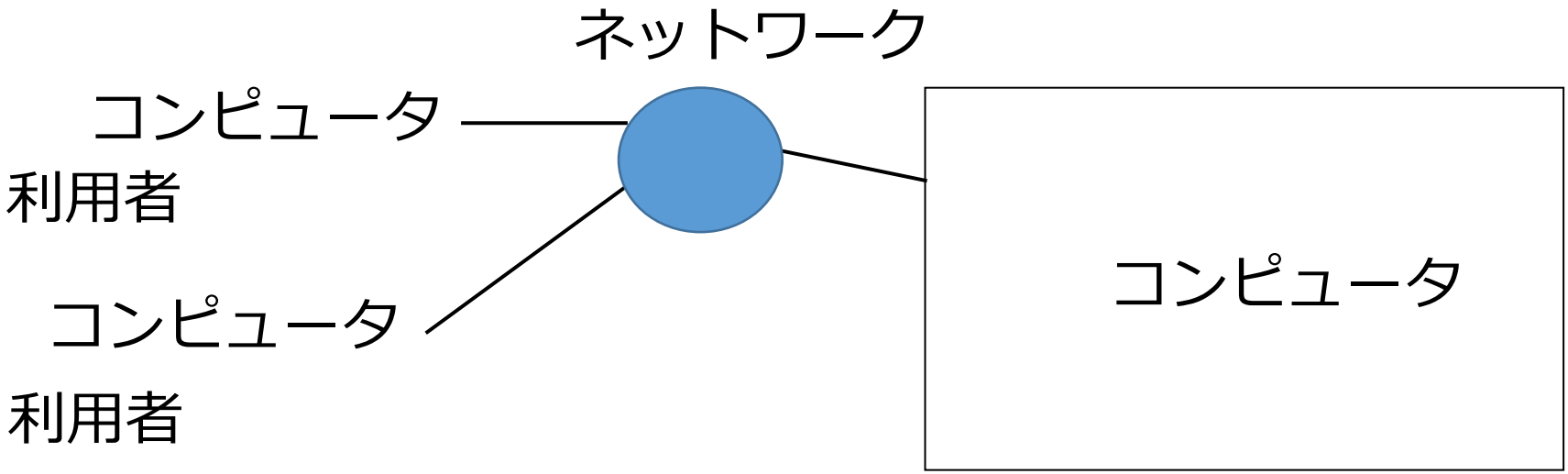
② プログラムは，コンピュータの動作をコントロール

```
In [7]: from keras.models import Sequential
   ...: model = Sequential()
   ...: from keras.layers import Dense, Activation
   ...:
   ...: model.add(Dense(units=64, input_dim=len(x_train[0])))
   ...: model.add(Activation('relu'))
   ...: model.add(Dense(units=max(set(y_train)) - min(set(y_train)) + 1))
   ...: model.add(Activation('softmax'))
   ...: model.compile(loss='sparse_categorical_crossentropy',
   ...:               optimizer='sgd',
   ...:               metrics=['accuracy'])
   ...: model.fit(x_train, y_train, epochs=200)
   ...: score=model.evaluate(x_test, y_test, batch_size=1)
   ...: print(score)
   ...: model.predict(x_test)
   ...: model.summary()
Epoch 1/200
3/3 [==============================] - 0s 5ms/step - loss: 1.0583 - accuracy:
0.3200
Epoch 2/200
3/3 [==============================] - 0s 0s/step - loss: 1.0530 - accuracy:
0.3200
Epoch 3/200
3/3 [==============================] - 0s 0s/step - loss: 1.0485 - accuracy:
0.3200
```

Python 言語を使って
ニューラルネットワーク
を作成． AIシステムを構築

プログラムは，**コンピュータ**の動作を細かくコントール

③ プログラムは，コンピュータ間の連携にも役立つ



サーバは，サービスを提供するITシステム

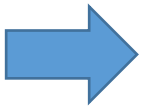
コンピュータ同士の接続でも**プログラム**が必要.

プログラミングの目的

- **プログラム**は，**コンピュータ**に指示を出し，所定の作業を遂行させる
- 複雑な作業も**自動化**し，効率化することが可能

```
a = [200, 400, 300]
for i in a:
    print (i * 1.08)
```

Python プログラムの
ソースコード



```
216.0
432.0
324.0
```

プログラムの
実行結果

プログラミングの利点

① **プログラム**の内容によって，**コンピュータ**は**さまざまな作業を実行**できる

② **プログラム**を利用することで，多くの作業を**自動化**できる

③ **プログラム**で行った作業を**いつでも再現できる**.

④ **プログラム**は柔軟性がある.変更により，**プログラムの動作を簡単に調整できる**

Python

- **Python** は多くの人々に利用されている**プログラミング言語**の１つ
- **読みやすさ，書きやすさ，幅広い応用範囲**が特徴

```
from keras.models import Sequentia
model = Sequential()
from keras.layers import Dense, Ac

model.add(Dense(units=64, input_di
model.add(Activation('relu'))
model.add(Dense(units=max(set(y_tr
model.add(Activation('softmax'))
model.compile(loss='sparse_categor
              optimizer='sgd',
              metrics=['accuracy']
model.fit(x_train, y_train, epochs
score=model.evaluate(x_test, y_tes
print(score)
model.predict(x_test)
model.summary()
poch 1/200
'3 [==============================] - 0s
3200
och 2/200
3 [==============================] - 0s
200
ch 3/200
[==============================] - 0s
0
```

Python 言語が広く使用されている理由

文法のシンプルさ

- **Python** は，**直感的で読みやすい**文法

拡張性

- **豊富なライブラリとモジュール**を持ち、**多岐にわたる分野で利用**が可能

柔軟性

- シンプルなスクリプトも、高度なプログラムも作成可能

主な機能

- **基本的な機能**： 変数、データ型、条件分岐（if）、繰り返し（for、while）、関数
- **オブジェクト指向**： クラス、スーパークラス、サブクラス、継承
- **モジュールとパッケージ**： 再利用可能なコード
- **例外処理**： エラーを効果的に処理
- **ファイル操作**： ファイルの作成、読み出し、書き込み
- **表示**： テキストでの表示、ビジュアルな表示

Python 言語が広く使用されている理由

文法のシンプルさ

- **Python** は，**直感的で読みやすい**文法
- 例えば、**print** で簡単に**出力**できる、**if** や **else** で**条件分岐**、**for** や **while** で**繰り返し（ループ）**

拡張性

- **多岐にわたる分野で利用**が可能
- 例えば、**関数やクラスを定義**するための **def** や **class**、**継承**や**オブジェクトの属性名と値**を操作するための **super** や **vars** などがある。

柔軟性

- シンプルなスクリプトも、高度なプログラムも作成可能
- **オブジェクト指向**の機能を持ち、**__init__** や **self** のようなキーワードを使用して**クラス**を利用できる。

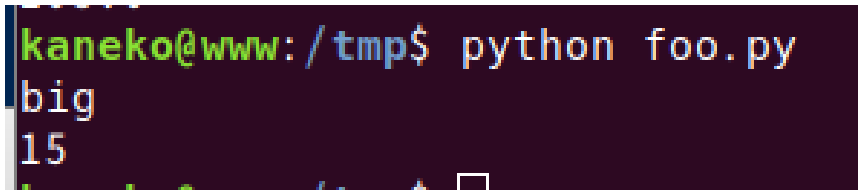
コマンドによる Python プログラム実行

① Python プログラムの保存

```
x = 100
if (x > 20):
    print("big")
else:
    print("small")
s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)
```

作成した **Python プログラム**の**ソースコード**を，例えば「foo.py」という名前の**ファイルに保存**

② Python プログラムの実行



プログラムを実行するには，シェル（例えば，Windows の場合はコマンドプロンプト）を開き，「python foo.py」のようなコマンドで実行

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）

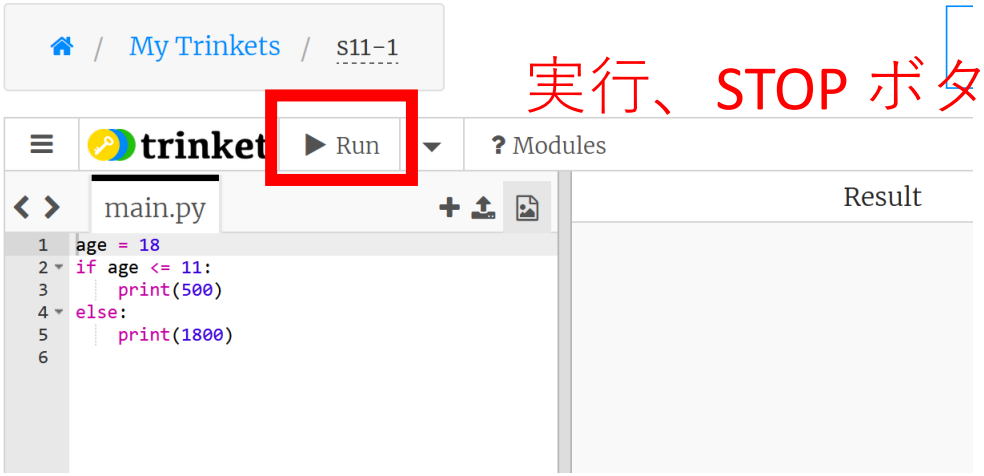
- **Python の標準機能**を登載、その他、次のモジュールやパッケージがインストール済み

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/0fd59392c8

のように、違うプログラムには違う URL が割り当てられる



- 実行が開始しないときは、「**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

演習
Python プログラムの実行

資料：６０，６１

【トピックス】

- **trinket**
- **Python プログラムのソースコード**
- **Python プログラムの実行**

① trinket の次のページを開く

https://trinket.io/python/6c652f1c2f

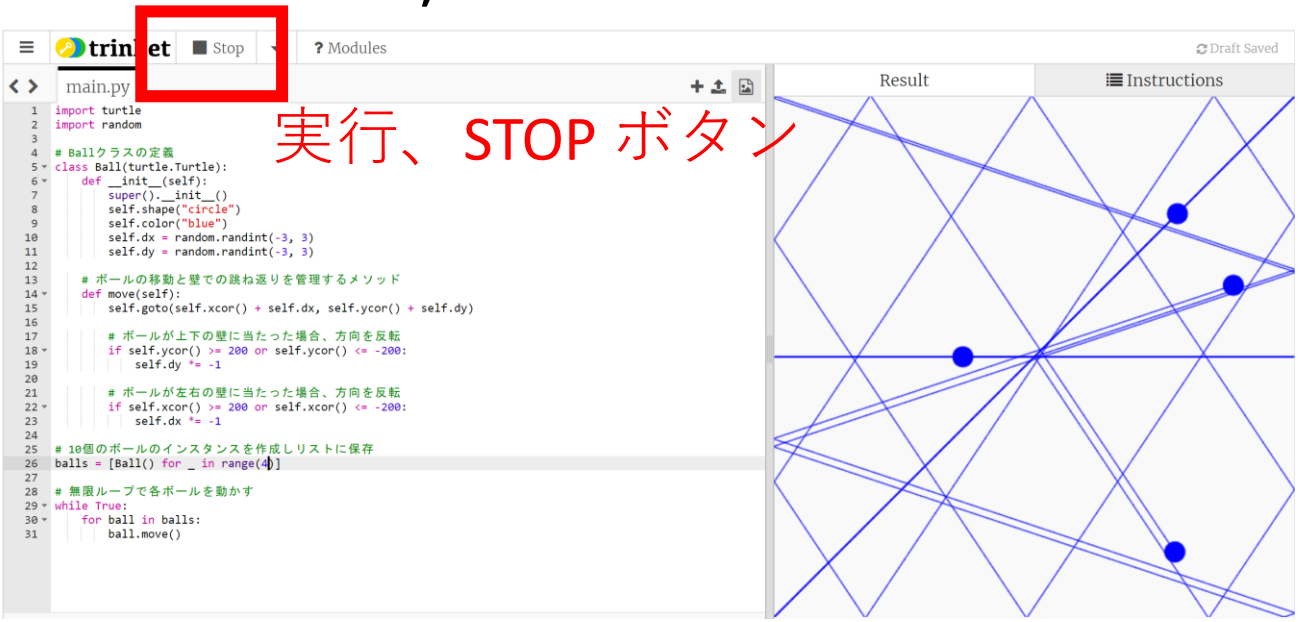
② 実行結果が，次のように表示されることを確認

- 実行が開始しないときは、「**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

③ trinket の次のページを開く

https://trinket.io/python/94d1563844

④ 実行結果が，次のように表示されることを確認



ボールが
壁に当たったら
反射する.

- 実行が開始しないときは、「**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

演習
簡単なプログラムでも
さまざまなことが可能

資料：６３～６６

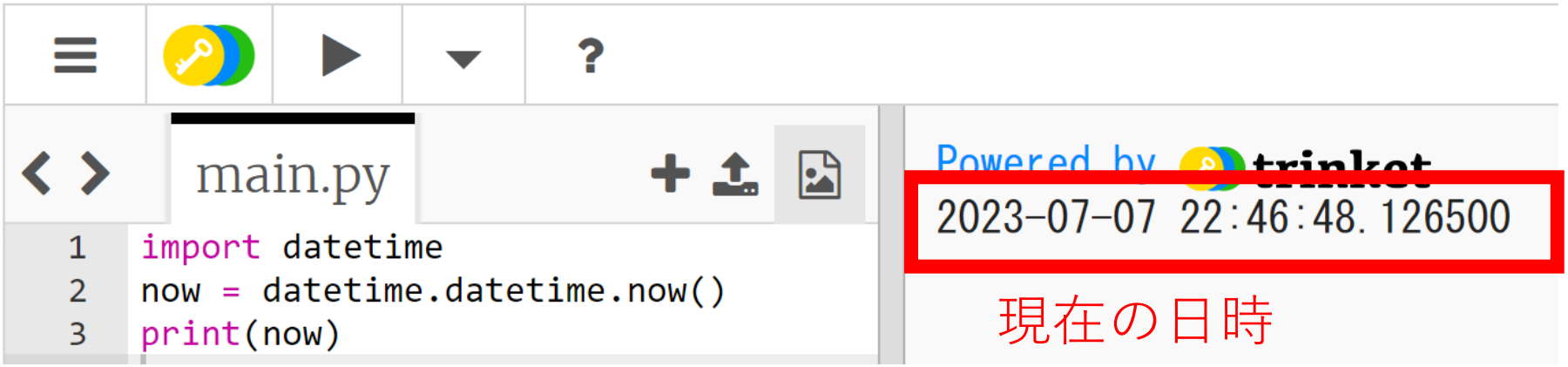
【トピックス】

- **trinket**
- **現在の日時**
- **平方根**
- **円の面積**
- **三角形の面積**

オペレーティングシステム（コンピュータ）のタイマーを利用．**現在の日時が表示される**

① trinket の次のページを開く
https://trinket.io/python/2b804ab19a

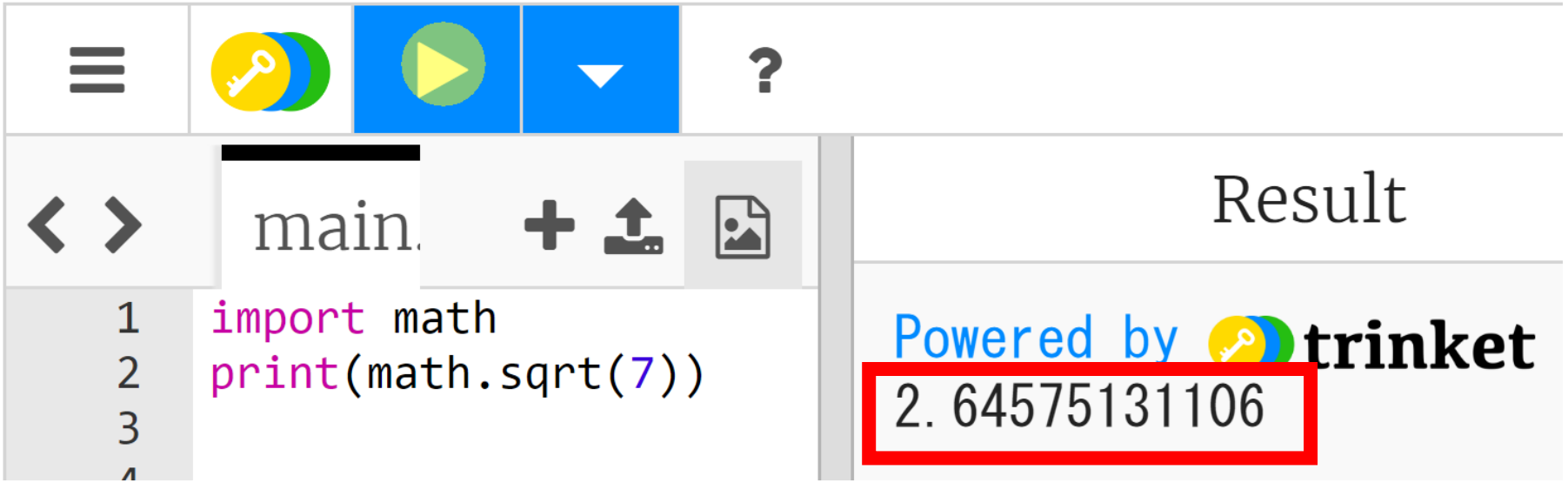
② 実行結果が，次のように表示されることを確認



面積が 7 の正方形の一辺の長さ

③ trinket の次のページを開く
https://trinket.io/python/597e5771ff

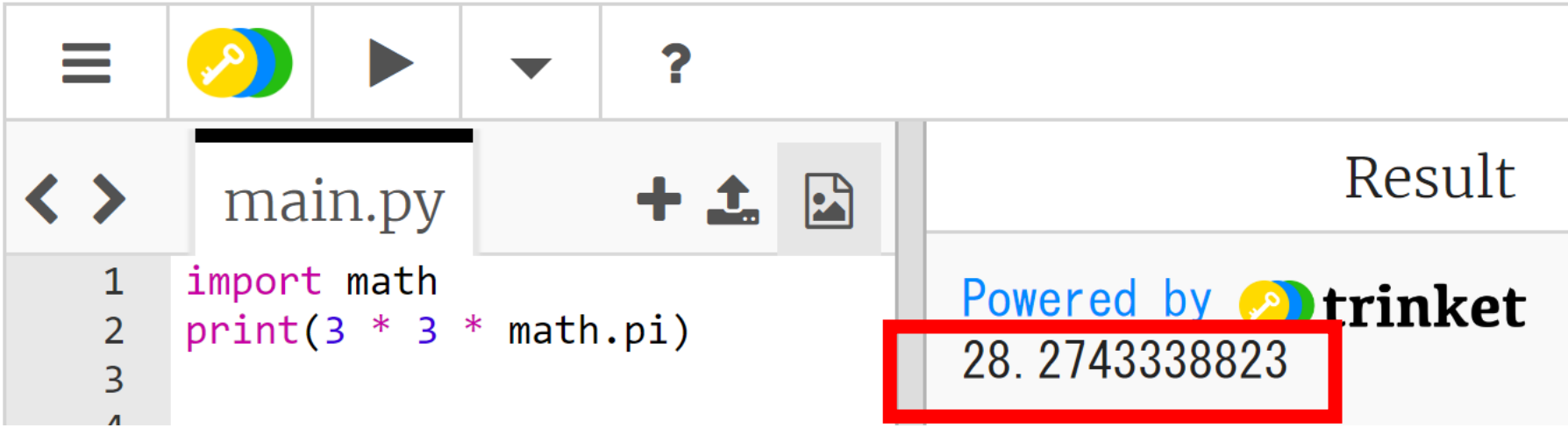
④ 実行結果が，次のように表示されることを確認



半径 **3** の円の面積は？

⑤ trinket の次のページを開く
https://trinket.io/python/4e3559f879

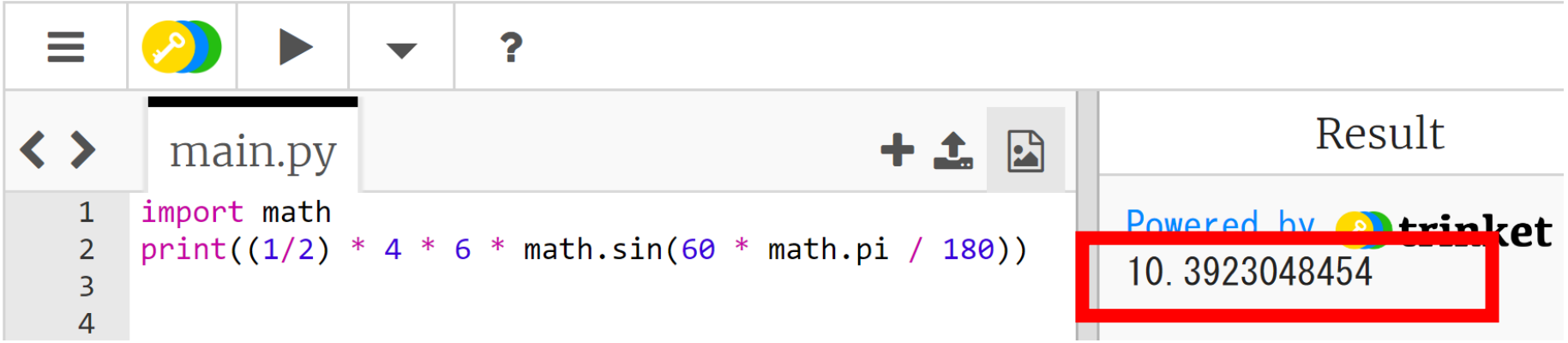
⑥ 実行結果が，次のように表示されることを確認



三角形の２辺の長さが，**4**と**6**で，その間の角度が**60**度のとき，面積は (1/2) × **4** × **6** × sin(**60**)

⑦ trinket の次のページを開く
https://trinket.io/python/bdcce27488

⑧ 実行結果が，次のように表示されることを確認



まとめ

- **プログラミングは創造的な活動**であり，多様な問題を解決し，**人間の能力を増幅**する
- **プログラミング**は，計算，データ処理，データ送受信，人工知能，グラフィックス，シミュレーションなどの**多様な活動**を可能にする
- **Python** は読みやすさと書きやすさに優れ，広い応用範囲を特徴とする**プログラミング言語**
- 学習サイトの Trinket では，Python プログラムの作成と共有が可能
- **プログラミングを学ぶ**ことで，**コンピュータを活用**し，**アイデアを具現化**し，**新たな創造**を**促進**できる.

1-5. 人工知能とプログラミングの関連

人工知能とプログラミングの関連性

- 人工知能とプログラミングの関連性
  - 人工知能は、**プログラムで動作**している。
  - **プログラムを通じて人工知能の動きや原理を理解する**ことができる。
- 人工知能の調整やカスタマイズ
  - **人工知能の微調整や最適化のため**にはプログラミングスキルが必須。
- インターフェースの役割
  - **人工知能と人間の間**の**コミュニケーションインターフェース作成**にもプログラミングは不可欠。

1-6. 15回の全体計画（次のステップへ）

１５回の全体概要

- **ディープラーニングの実践**を通じて**楽しさ**を感じる
- ディープラーニングの仕組みを探求
- **専門知識**、**実践スキル**、**活用力**を向上
- **人工知能の実行**は**パソコン**で行う
- 自然言語処理、画像理解、顔情報処理、姿勢推定などの応用を学ぶ
- 自ら学ぶアクティブラーニングを重視

15回の内容

- 第1回：人工知能の歴史，種類，応用分野
- 第2回：Python プログラムの実行，Python でのデータマネジメント
- 第3回：機械学習の基礎（教師なし学習，教師あり学習）
- 第4回：ディープラーニングの基礎
- 第5回：ディープラーニングでの物体認識，画像分類
- 第6回：ディープラーニングでの物体検出，セグメンテーション
- 第7回：顔情報処理
- 第8回：予測・判断（時系列データ，リカレントニューラルネットワーク，LSTM）
- 第9回：自然言語処理（問答，要約，テキスト生成，単語の特徴ベクトル，単語の類似度
- 第10回：姿勢推定
- 第11回：中間まとめ
- 第12回：実データの分析，意味の抽出，外れ値の判断
- 第13回：人工知能によるデータの生成
- 第14回：人工知能システムの運用，人工知能が社会に受け入れられるには（倫理，プライバシー，個人情報，人工知能の信頼性や説明可能性）
- 第15回：人工知能システムの構築（人工知能システムの開発環境，実行環境）