プログラミング

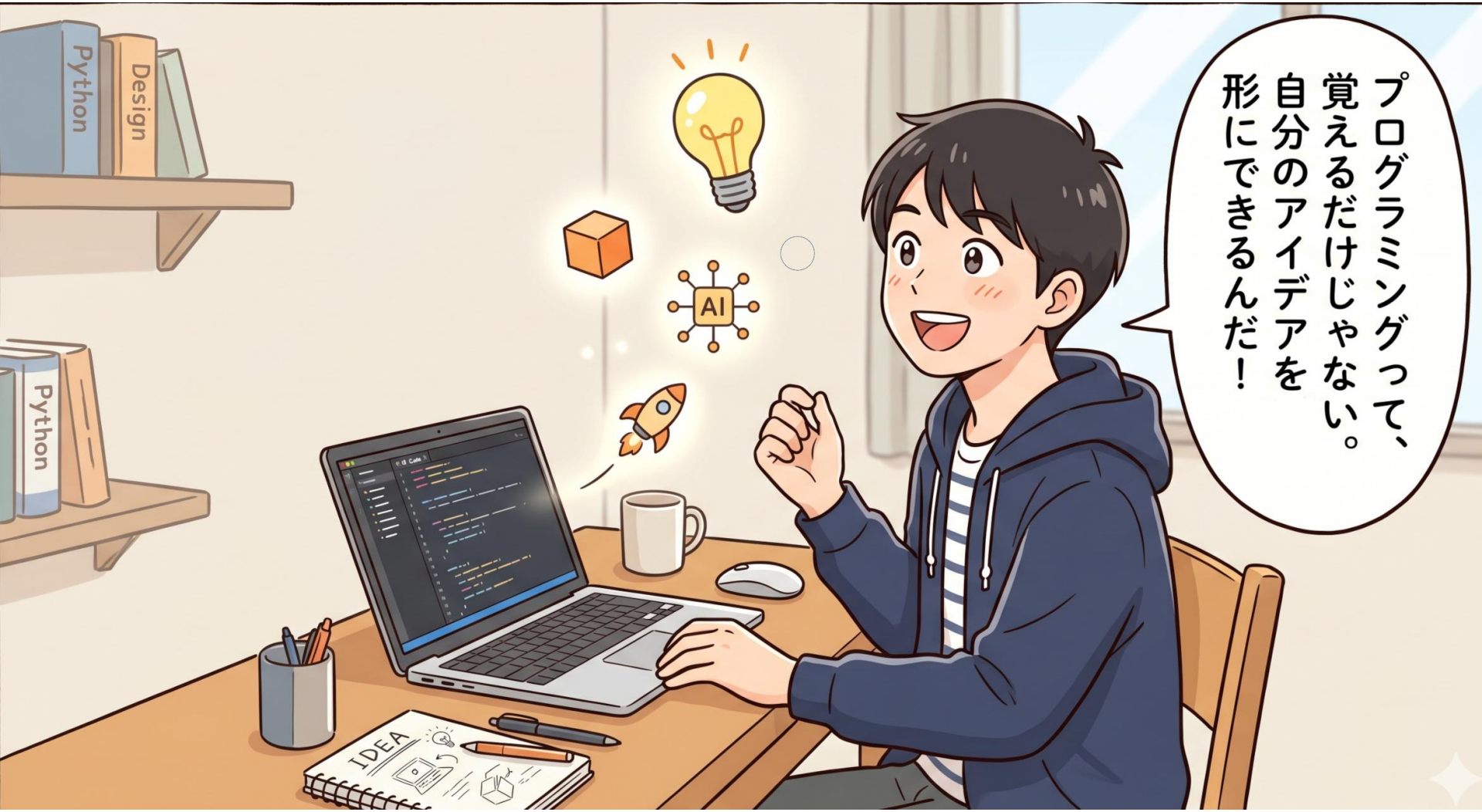
（プログラミング基礎・用語集・体験）

URL: https://www.kkaneko.jp/pro/man/index.html

金子邦彦

資料中の図などは Google Nano Banana 2 を用いて作成

1. プログラミングの創造性と学ぶ意義

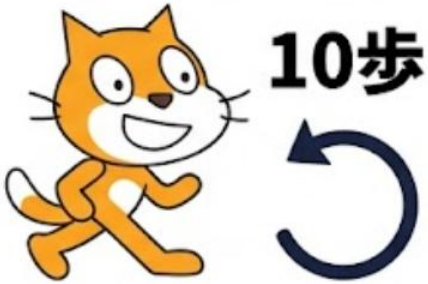


本授業の図は生成AIである Google Nano Banana 2 を用いて作成

プログラムとは ― 命令を書いた手順の集まり

プログラムは、命令を書いた手順の集まり

指示を与える → **自動で処理** → **結果を得る**



くりかえし

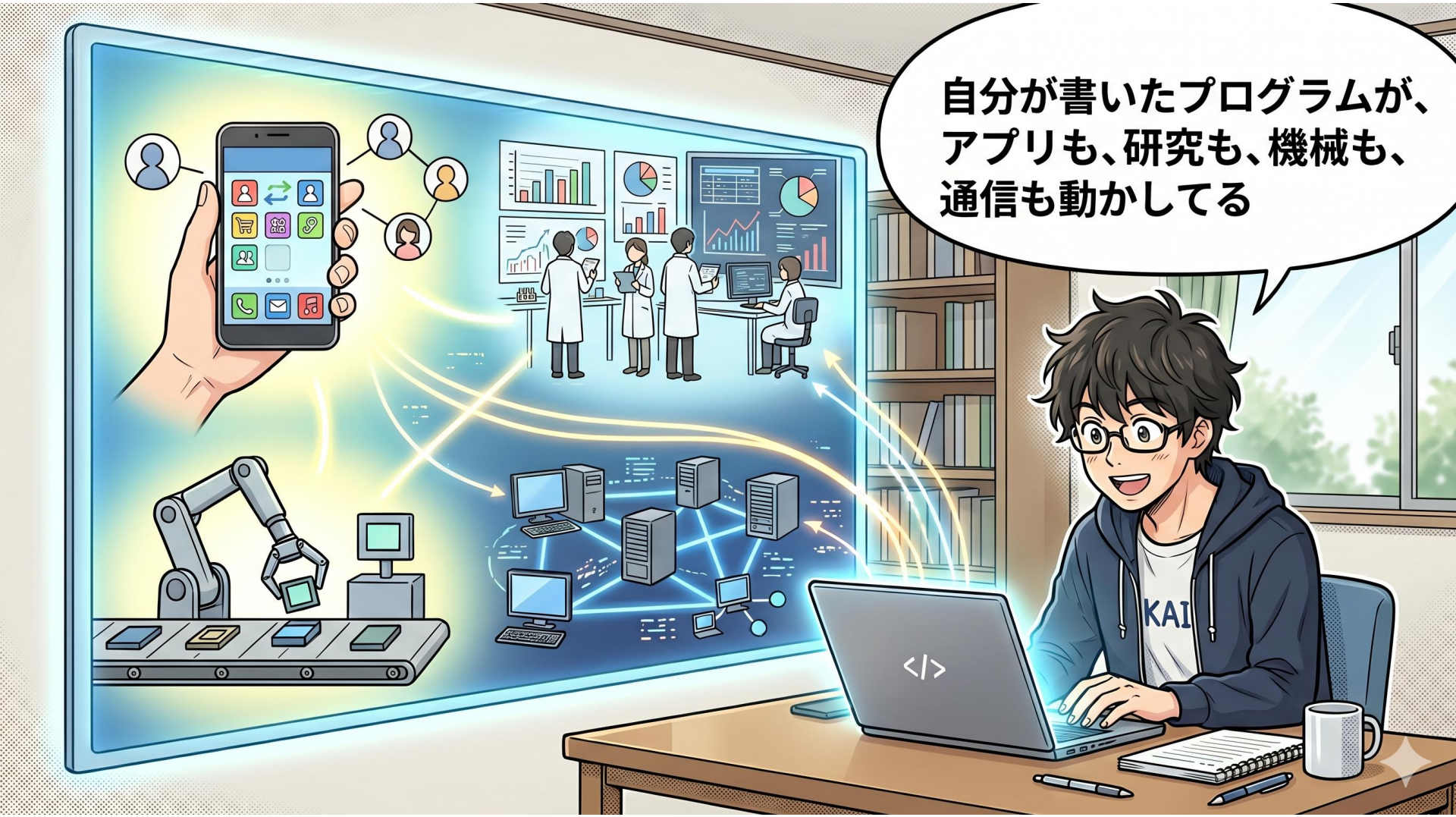
合計40歩進んだ!

だから複雑な作業も自動化・効率化できる

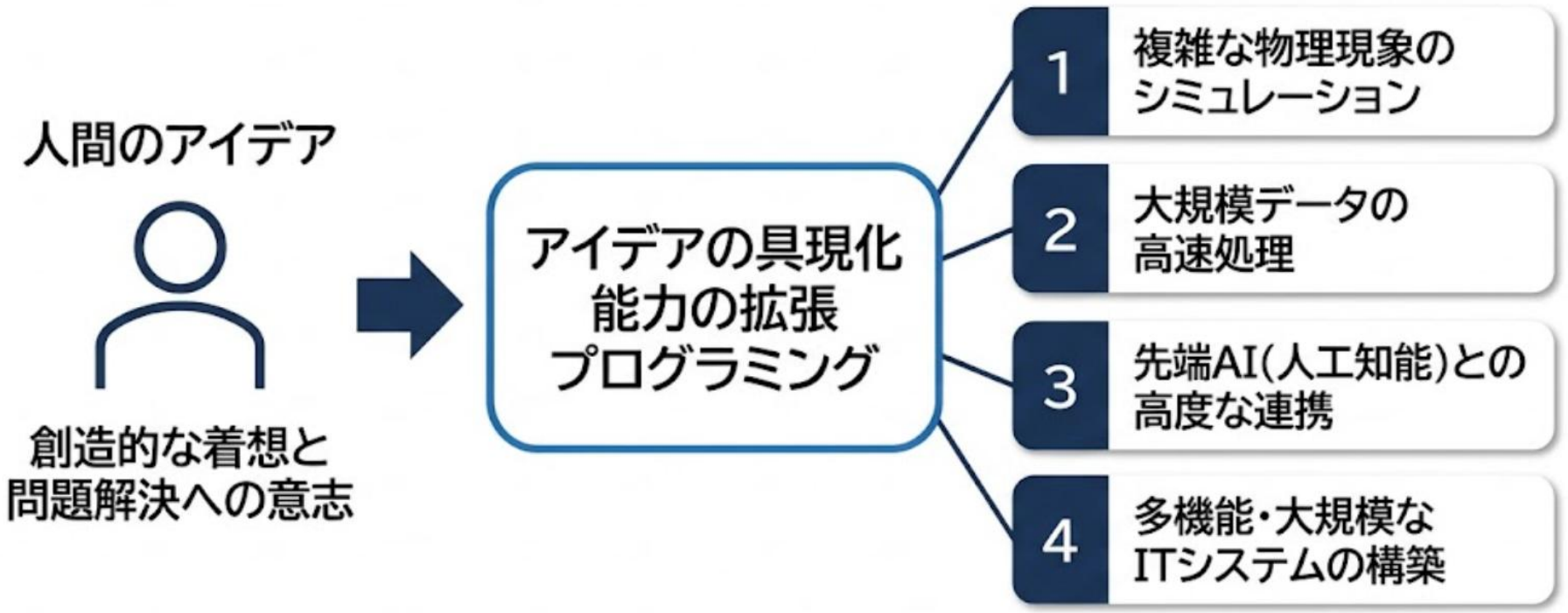
プログラミングは創造的

プログラミングを学ぶことで何が身につくか

2. プログラミングの有用性


自分が書いたプログラムが、
アプリも、研究も、機械も、
通信も動かしてる
KAI
</>

プログラミングで何ができるのか



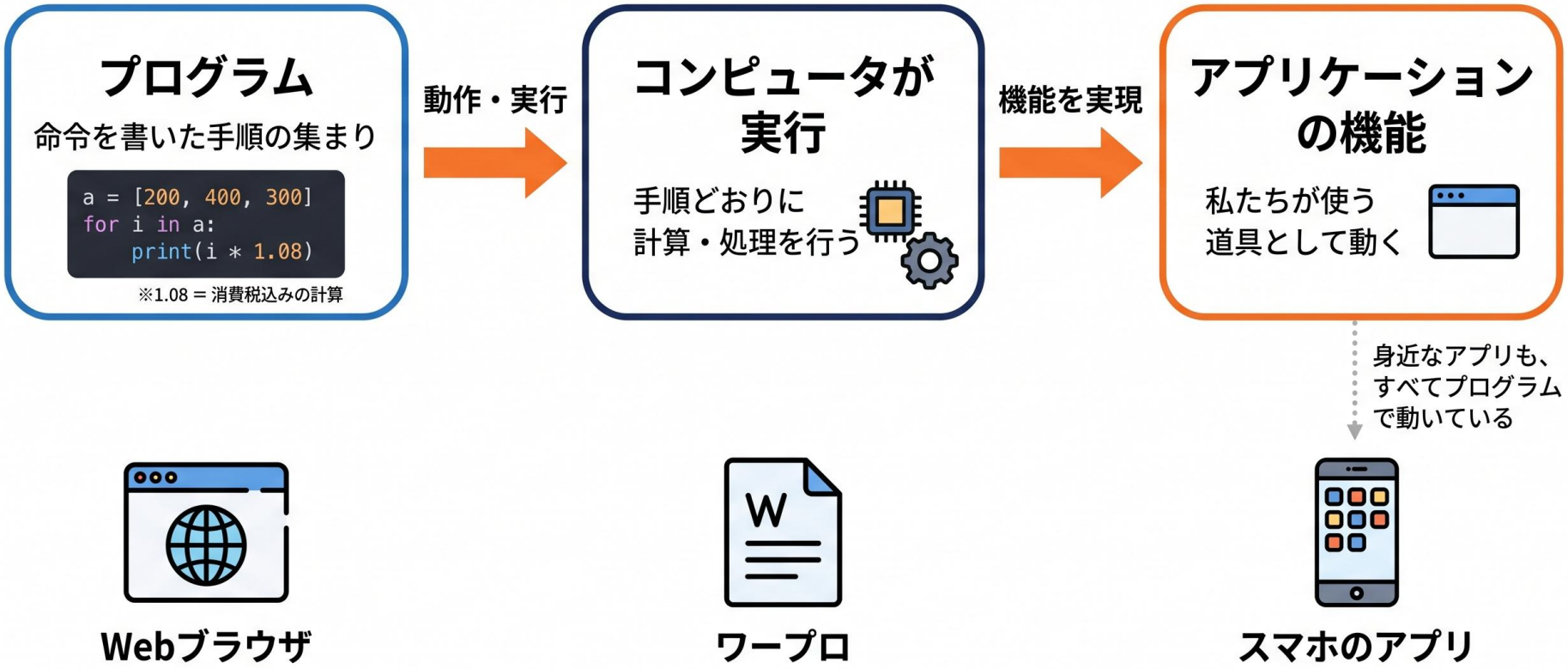
誤解:単なる『覚えて、問題を解くだけの勉強』

正解:『自分のアイデアを形にする』創造的な作業

プログラムとは — 命令を書いた手順の集まり

だから複雑な作業も自動化・効率化できる

プログラミングの用途① アプリケーションの実現



プログラムが動作することで、ブラウザやワープロなどアプリケーションの機能が実現される。

プログラミングの用途② 最先端の研究・実験

Database Lab.

Python言語で記述する

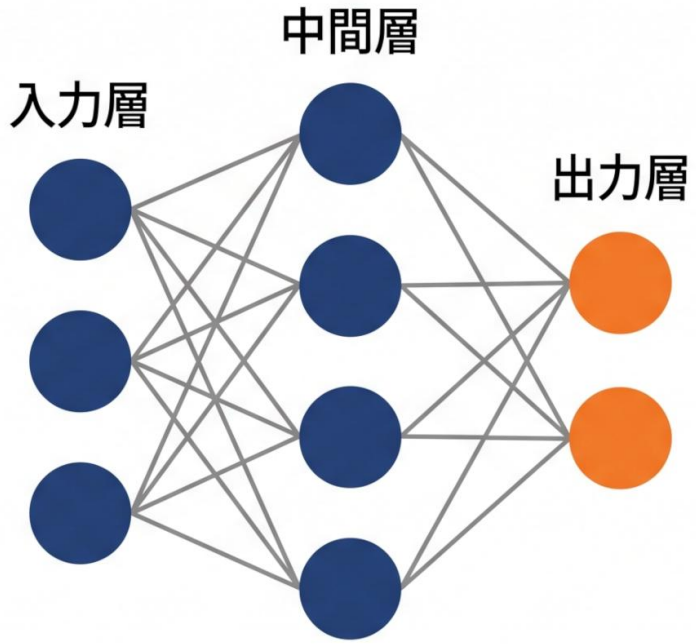
```
import torch.nn as nn

model = nn.Sequential(
    nn.Linear(3, 4),
    nn.ReLU(),
    nn.Linear(4, 2),
)

pred = model(x)
loss = loss_fn(pred, y)
loss.backward()
```

人間に読みやすい文法で
命令を記述

ニューラルネットワークで AIを構築する



データから規則を学習し、
判断・予測を行うAIシステム

■ AI構築以外にも様々なことが可能

■ Python以外の言語もある

プログラミングの用途③ コンピュータの制御

Database Lab.

プログラムが
コンピュータを制御する

プログラム（命令を書いた手順の集まり）

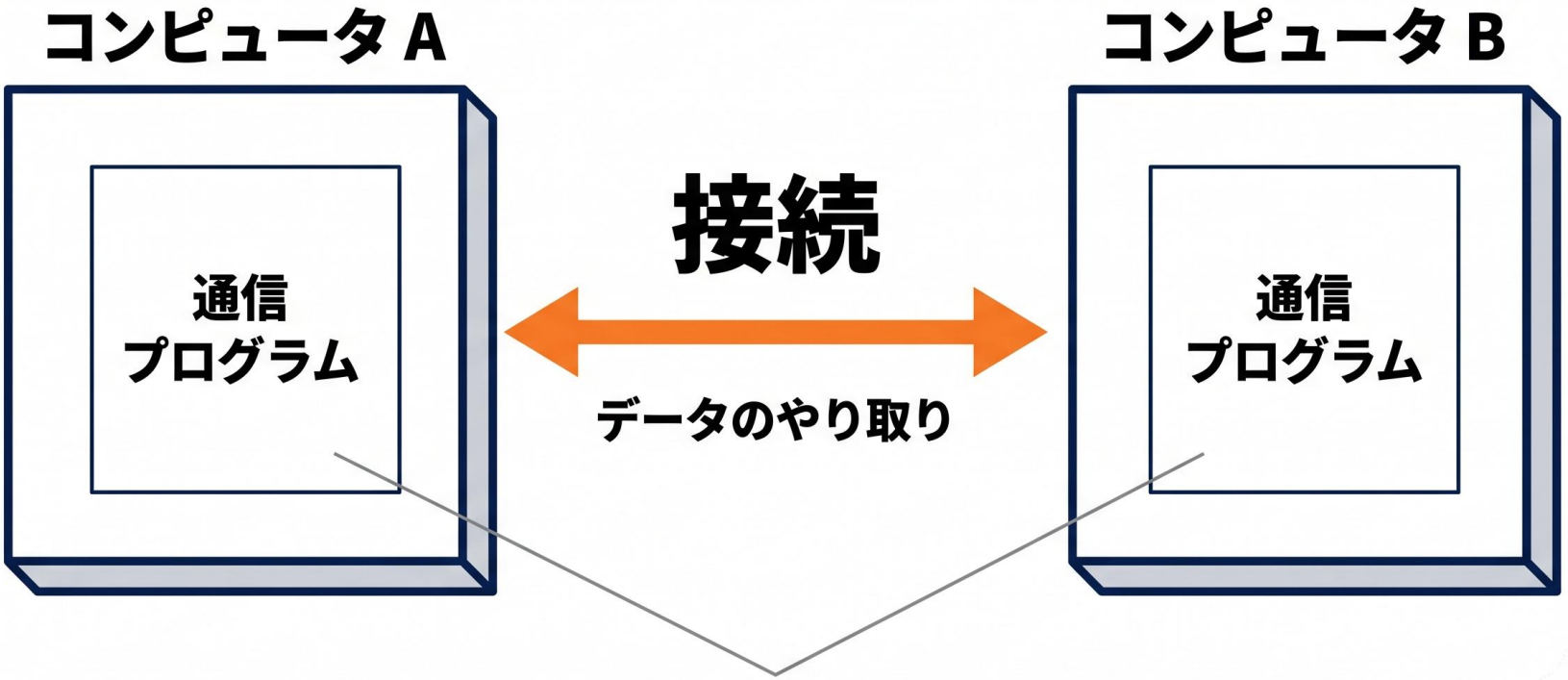
命令を実行

コンピュータ

CPU	メモリ	入出力
制御・演算	記憶	入力・出力

プログラミングの用途④ コンピュータ同士の接続

コンピュータ同士の接続にはプログラムが必要



接続を行う処理そのものをプログラムが担当する
相手とやり取りする手順（ルール＝プロトコル）に従ってプログラムが動作する

3. ソースコード、開発環境


ソースコードは
人とコンピュータの
“共通のことば”。
読めて、書けて、
あとから直せる。
だから扱いやすい！
書く
実行
結果

ソースコード

ソースコードは、人間が読み書きできる言葉で書かれたプログラムである

Python 例

```
price = 100
tax = price * 0.1
print(price + tax)
```

人間が読める言葉
（プログラミング言語）で
書かれている

編集・修正

3つの特徴

1. 読める … 何をする処理か理解できる
2. 書ける … 人が新しく作成できる
3. 直せる … 必要に応じて改変できる

改変
変更前：`tax = price * 0.1`
変更後：`tax = price * 0.08`

実行での流れ

Python プログラムのパソコン上での実行

Python プログラムはオンライン実行（例：Trinket）のほか，パソコンでも実行可能．（パソコンでの実行の場合には，Python 処理系のインストールが必要）

① Python プログラムのファイル保存

```
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

```
kaneko@www:/tmp$ python foo.py
big
15
```

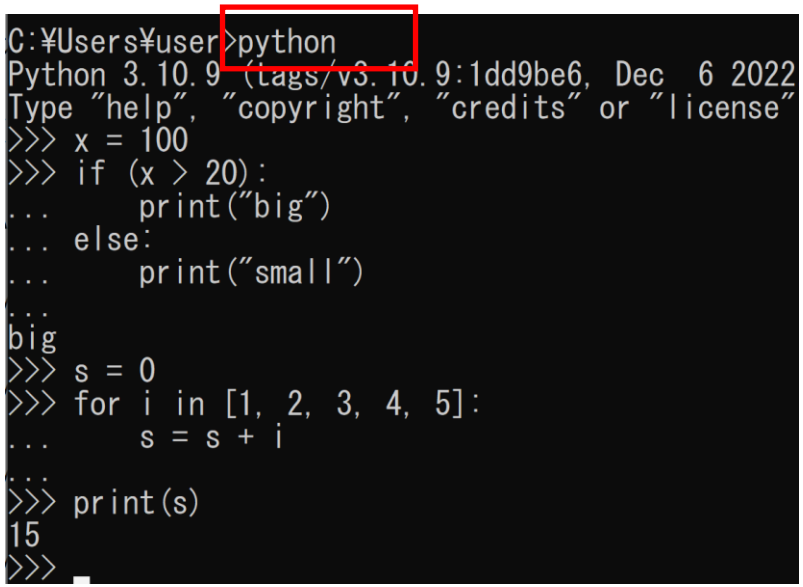
プログラムを実行するには，シェル（例えば，Windows の場合はコマンドプロンプト）を開き，「python foo.py」のようなコマンドで実行

コマンドプロンプトによるPython実行

Windowsでコマンドプロンプトから**Python**を実行する場合

- Pythonのインストール（https://www.python.org から）
- **pythonコマンド**でPythonを起動すると，**プログラムを入力するたびに結果が得られる対話的実行**が可能
- 終了は**exit()コマンド**

Python プログラム実行結果



コマンドプロンプトで Pythonで開始

- Windows では，**python** コマンドで実行
- 終了は **exit()**

開発環境とは

開発環境とは、**プログラミング**におけるさまざまなことを支援する機能をもった**プログラム**

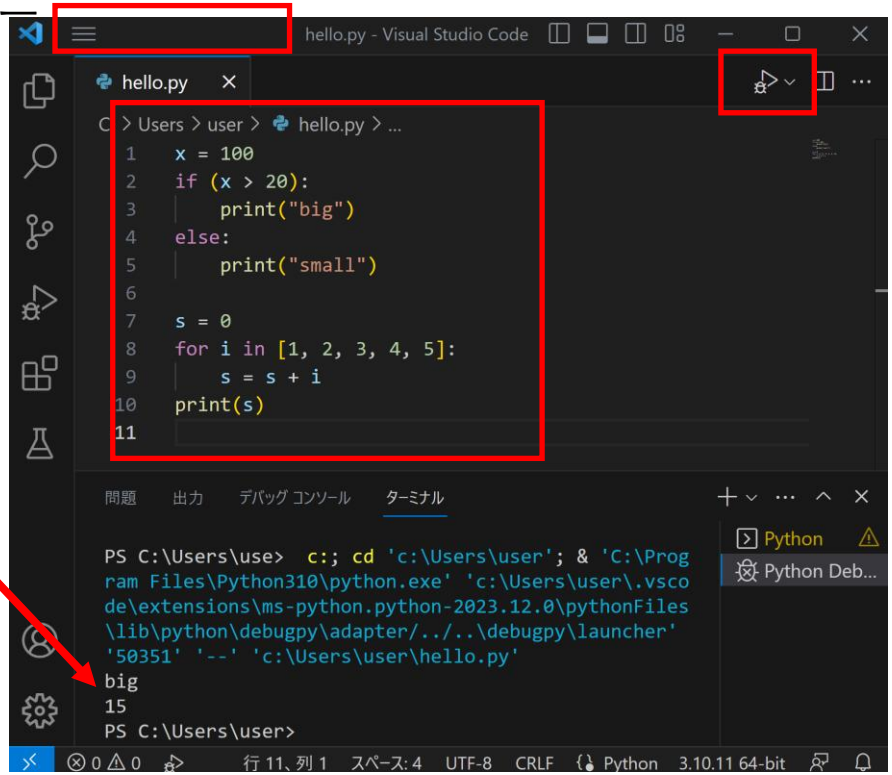
- プログラムの作成、編集（**エディタ**）
- プログラム中の誤り（**バグ**）の発見やテストの支援（**デバッガ**）
- プログラムの実行
- マニュアルの表示
- プログラムが扱うファイルのブラウズ
- プログラムの配布（**パッケージ機能**など），共有，共同編集
- 公開，共有，共同編集
- バックアップ，バージョン管理

※　これらが簡単に行えるようになる

Visual Studio Codeによる開発

- Visual Studio Codeは，Microsoftが開発した多言語対応の統合開発環境
- Pythonのインストール（https://www.python.org から）
- Visual Studio Codeのインストール（https://www.microsoft.com/ja-jp/dev/products/code-vs.aspx）が必要
- 編集画面でプログラムを編集し，ターミナル（端末）で実行結果を確認できる．デバッグ機能により変数探索も可能である．

メニュー



編集

実行ボタン

実行結果

※「デバッグ」の機能により変数探索も可能

Jupyterノートブックによる開発とドキュメント化

Jupyterノートブックは，Anacondaに含まれる対話型の開発環境.

- Pythonのインストール（https://www.python.org から）
- Anacondaに含まれている（https://www.anaconda.com）
- ノートブックの画面上で，コードセルやテキストセルを追加可能
- 実行ボタンで実行でき，グラフ表示や画像の表示が容易である.
- プログラム，実行結果，テキスト，画像を含む全体をノートブックとして保存できる.

実行ボタン

編集

実行結果

編集

実行結果

編集

実行結果

※ 変数探索は「%whos」

```
%whos
```

```
Variable   Type       Data/Info
------------------------------
i          int        5
plt        module     <module 'matplotlib.pyplo<...>¥¥matplotlib¥¥pyplot.py'>
s          int        15
x          list       n=4
y          list       n=4
```

ステップ実行と通常実行の概要

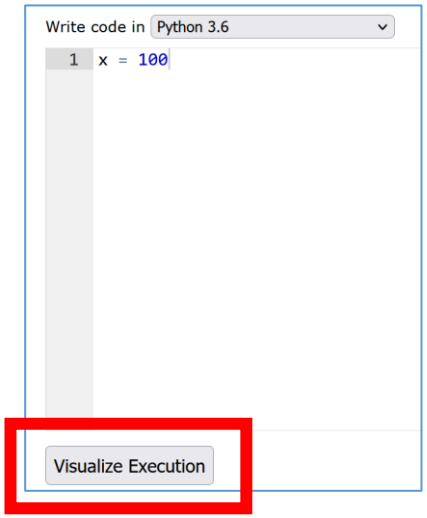
- **ステップ実行**では，**１行ずつの実行**が行われ，そのときの変数の値の変化などを**確認**できる．**プログラムの動作を細かく追跡**でき、不具合が発生している箇所の特定，プログラムの学習に役立つ

- **通常実行**は，**プログラムを最初から最後まで一度に実行**するもので，プログラム実行中の変数の値の変化を確認するなどは困難。

Python Tutor によるステップ実行の活用

- **Python Tutor** は **Python などのプログラムを書き実行できる**サイト．**ステップ実行，変数の値表示**などの機能がある．
- **Python Tutor**のウェブサイト： https://www.pythontutor.com/ で「Python」を選択
- メイン画面でプログラムを書き，Visualize Executionボタンをクリックすると実行される．通常実行はLastボタン，ステップ実行は他のボタンで行う．

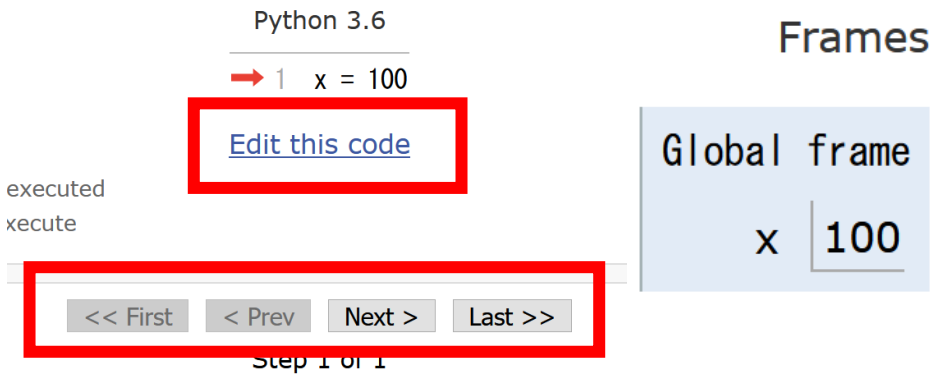
メイン画面で、プログラムを書く



Visualize Execution ボタン

メイン画面に戻るには Edit this code

変数の値を視覚的に確認できる



通常実行: Last
ステップ実行: 他のボタン

オンライン実行環境の使いわけ

目的：手軽に即実行

目的：本格的なAI・データ処理

OneCompiler

ブラウザだけで即座にプログラムを実行し、結果を確認できる

ビジュアルプログラミングにも対応

GDBonline

ブラウザだけで即実行・結果確認ができる

オブジェクト観察やステップ実行など、学習用機能が充実

Google Colab

無料で利用でき、ファイル共有も可能。AIアシスタント付き

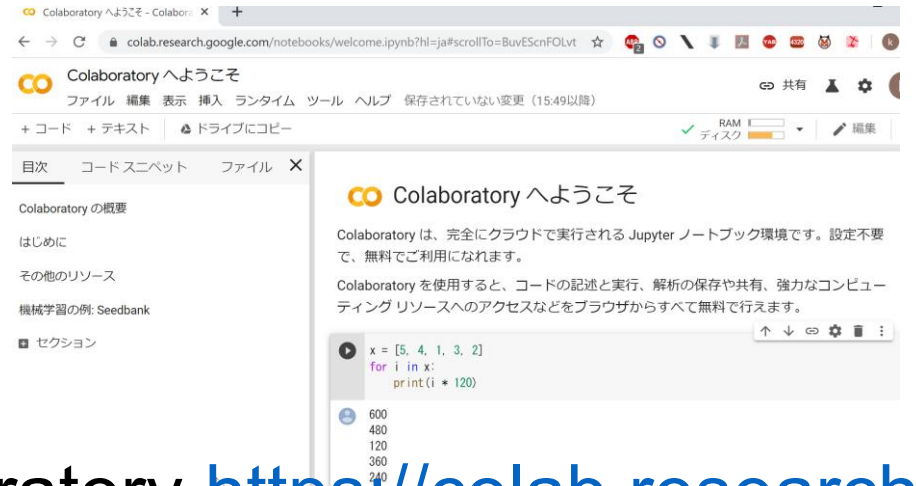
Googleアカウントがあれば、AIや3次元処理など本格的なプログラムも実行可能

OneCompiler / GDBonline で手軽に → **Google Colab で本格的に**

この２つは、この授業で使用

別の授業で紹介。自力で調べることも十分に可能

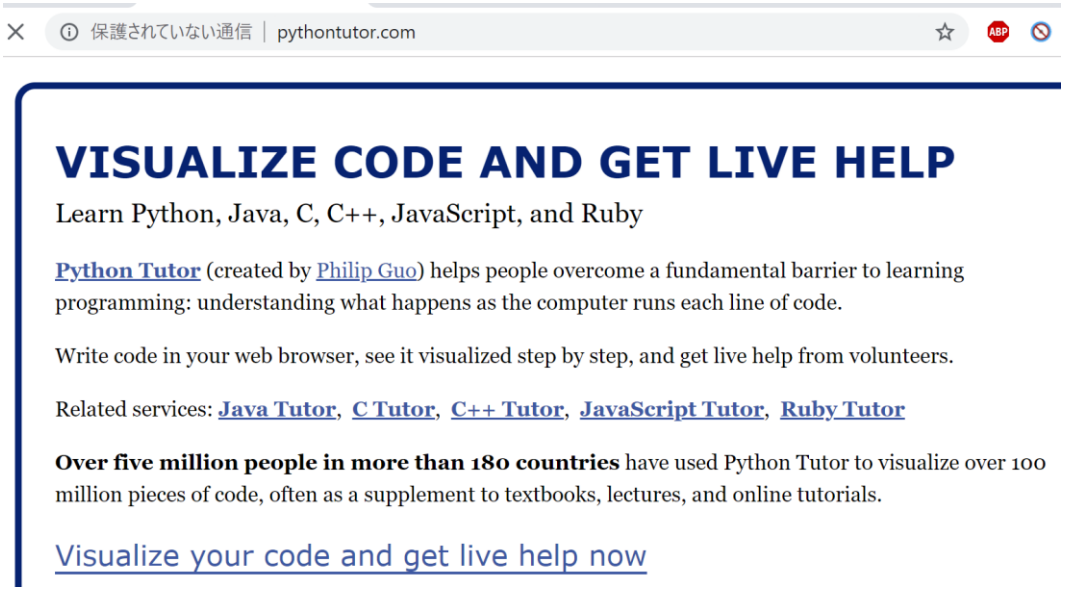
Google Colaboratory



Google Colaboratory https://colab.research.google.com/

- **Python の開発環境**
- 人工知能，データサイエンス，その他多数のパッケージがインストール済み
- **コードセル**，**テキストセル**を複数**ノートブック**にまとめ，**保存**や**公開**できる
- **ノートブックにより，記録が簡単に残せる．ビジュアルな表示も簡単に可能**
- プログラムの共有も簡単

Python Tutor



Python Tutor　http://www.pythontutor.com/

- Python, C, Java, JavaScript
- **ステップ実行，オブジェクトの表示がビジュアル**に