AIプログラミング

PYTHONによるEXCEL・WORD連携、AIプログラミング

情報システムアナリスト
小谷　和海
ksys@kkotani.net

コンピュータによって世界は大きく変わって来ている
一つがインターネット

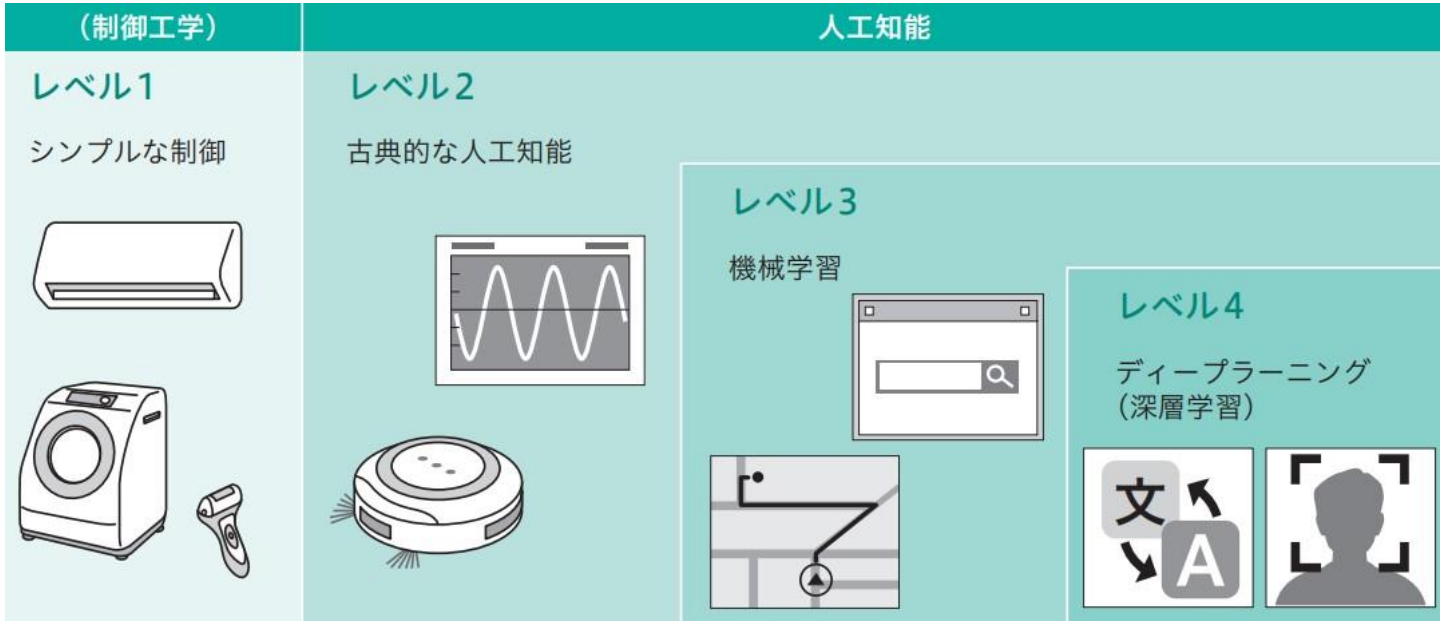
- インターネットの始まり
  - 1963年～　米国の国防高等研究計画局
- 全世界で約73%(3/4)
  - 世界人口　83億、　インターネットユーザ　約60億（2025年11月）
- インターネットの特徴　　独立して機能
  - 個々の端末が「自律・分散・協調」の原理で、各端末が独立して機能しながら、ネットワークを通じて相互に通信する仕組み。

二つ目が
人工知能(AI)

- 人工知能とは何か？
  - 「考える」「見分ける」「えらぶ」といった、人間がやっていることを、コンピュータができるようにした仕組み。
  - ただし、「AIとはこれだ！」という決まった答えは、専門家のあいだでもまだない。

- 強いAIと弱いAI
  - 強いAI
    - 人間みたいに考えたり、学んだり、自分で新しいことを理解したりできるAI
      ⇒ まだ実現されているとは言えない
  - 弱いAI
    - あらかじめ決められたルールや、たくさんのデータを使って動くAI
      たとえば「顔認証」「天気予報」など
    - AIのほとんどは、この弱いAI

人工知能(AI)の分類



レベル1:　きめられたとおりに動くロボット　(人が教えたルールをそのままやるだけ)
　　例:エアコン、洗濯機　など
レベル2:　ちょっとかしこいロボット(少しむずかしいことができる)
　　例:そうじロボット　など
レベル3:　まなんで強くなるロボット(たくさんの例を見て、“こういうときはこうする”を覚える)
　　例:走行ルートの最適探索　など
レベル4:自分で大事なところを見つけて学ぶロボット(大変むずかしいパターンも理解できる)
　　例:画像認識、音声認識、ChatGPT　など

⇒　かつては高度なAIの実現には100年以上かかると考えられていたが、レベル4(ディープラーニング)により予測は大きく変わった。

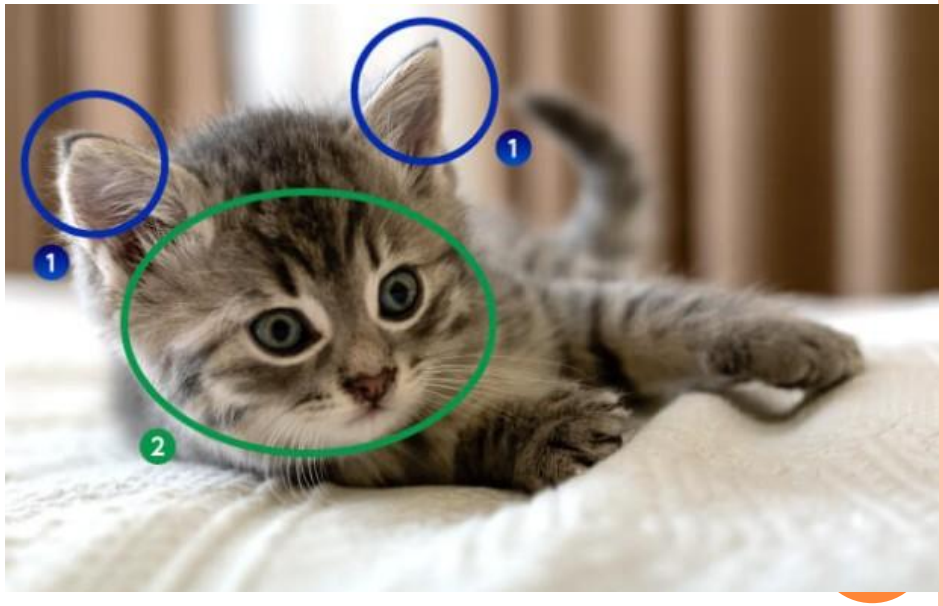
ディープラーニング（深層学習）とは

- Googleが“猫を見つけた”大ニュース（2012年）
  - 2012年、Googleの研究チームがコンピューターに猫を見つけさせることに成功！　このニュースは世界中を驚かせた。
  - コンピューターに1000万枚の画像 を見せて学ばせた。
- 猫を見つけるために、コンピューターは画像の中から 大事な形 を探す方法。

① 耳が2つある

② 顔の形がまるくカーブしている

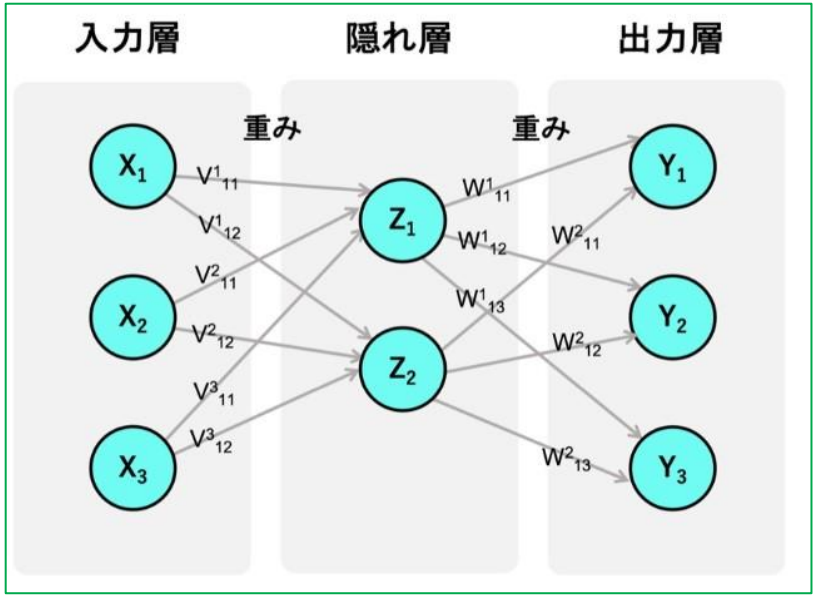
③ 目の位置がこのへんなど

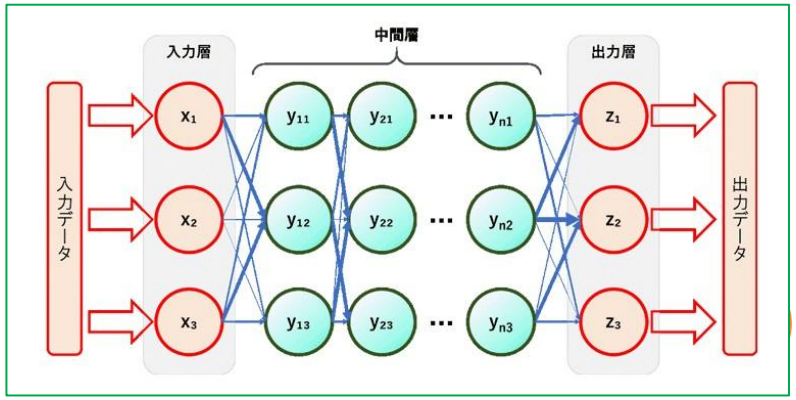


ニューラルネットワーク

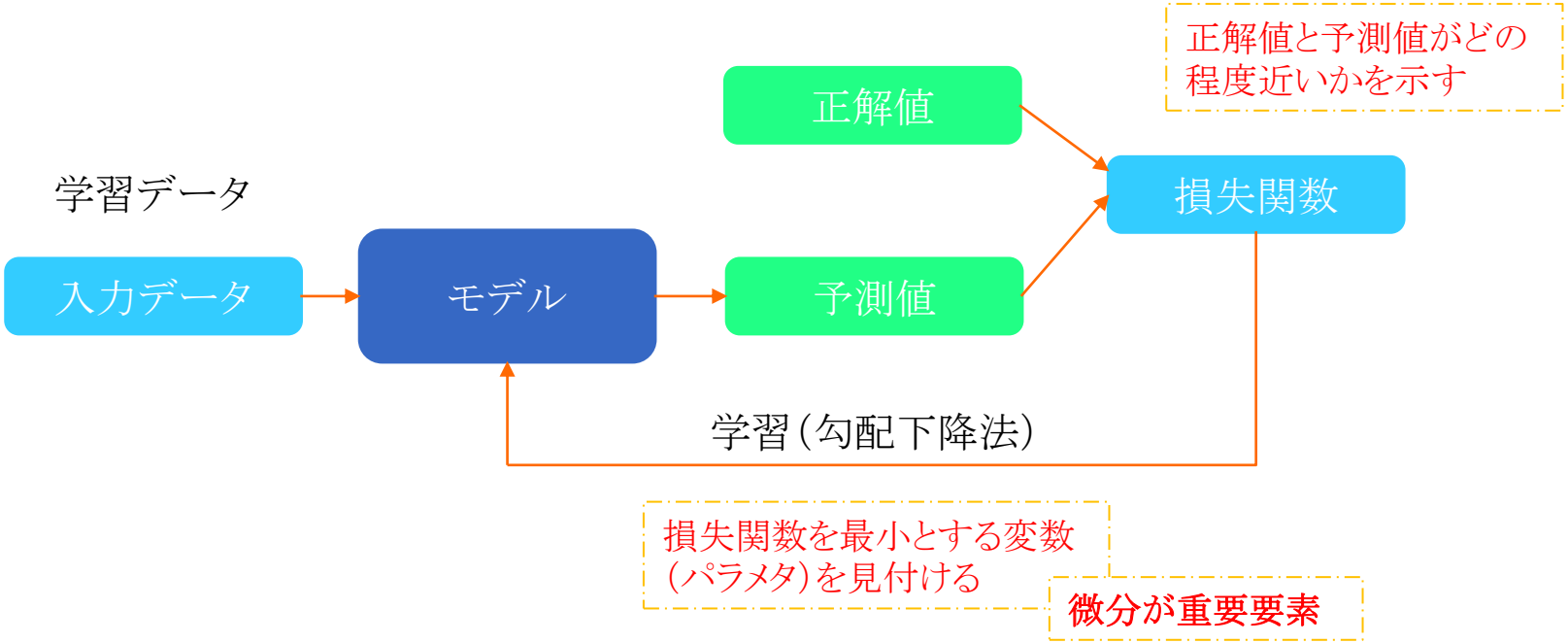
人間の脳は、神経細胞とそのネットワークで構成されている。
この動きを真似ることで高度な情報処理ができると考えた。
ニューラルネットワークは「層（Layer）」で構成されています。

- **入力層**
  データが最初に入る場所です。たとえば、画像認識の場合、ピクセルの値が入力されます。
- **隠れ層**
  入力データを処理する場所です。ここで計算が行われ、データの特徴が抽出されます。
  複数の隠れ層があるものを「ディープニューラルネットワーク」と呼びます。
- **出力層**
  最終的な結果を出す部分です。
  たとえば、猫か犬かを判別するタスクでは「猫」「犬」の確率が出力されます。



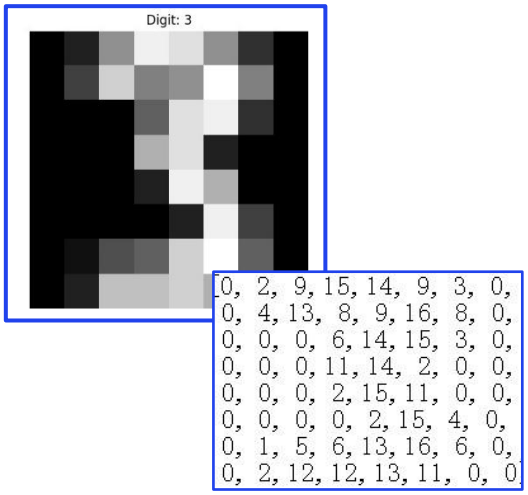


ニューラルネットワーク
学習の仕組

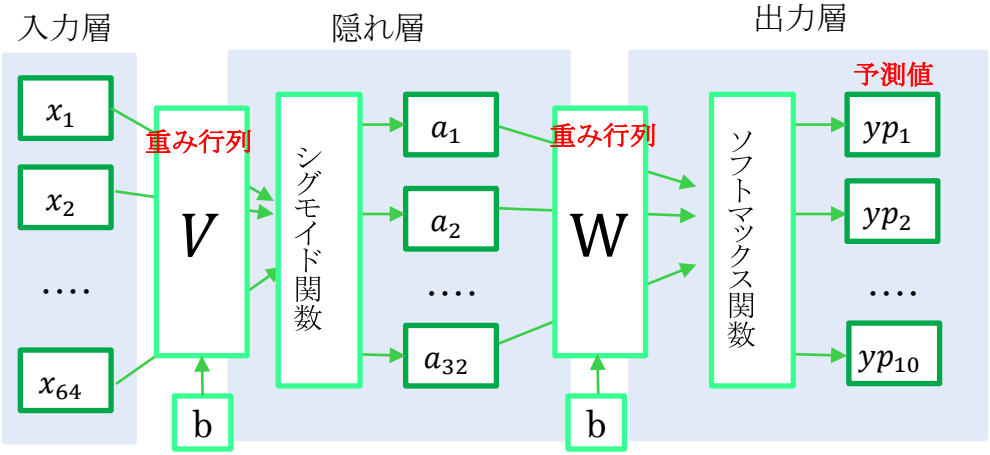


手書き文字認識3層ニューラルモデル

- 手書き数字文字認識（0～9）を行う3層ニューラルネットワーク
  - 8×8のグリッドの手書き数字文字認識



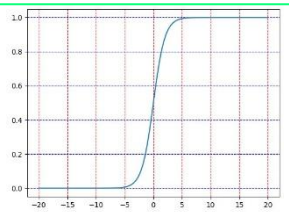
x_i
出力層
$(i = 1, \cdots, 64)$



X @ V + b

a @ W + b

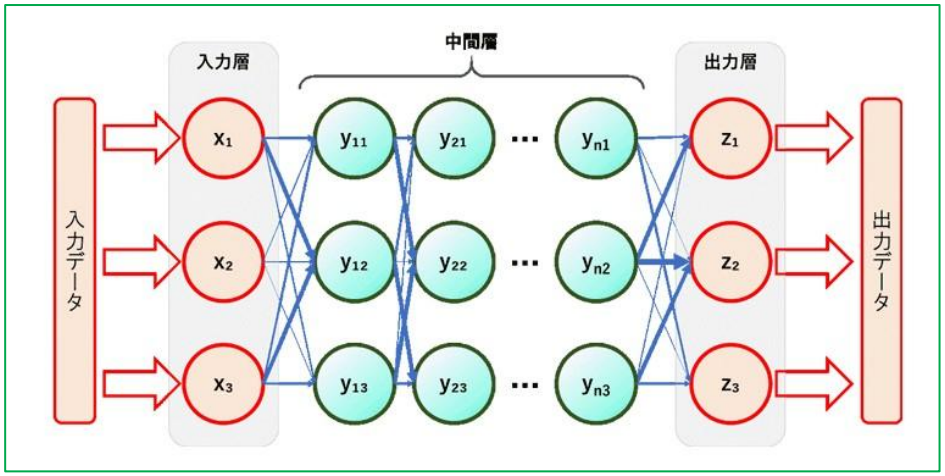
シグモイド関数（活性化関数）
入力の値を0から1の範囲に変換



ソフトマックス関数（活性化関数）
入力された数値を0から1の範囲の値に変換し、全体の合計が1になるようする。

ディープラーニング（層が複数）が強力な理由

- 層が深いほど情報を段階的に処理できるようになる。
  （複雑な問題を解けるようになる）
  - 画像なら
    - 第1層：エッジ
    - 第2層：角、模様
    - 第3層：パーツ（目・鼻）
    - 第4層：顔全体第
    - 5層：人物の種類
  - 言語なら
    - 第1層：文字・単語
    - 第2層：文法的関係
    - 第3層：意味的関係
    - 第4層：文脈
    - 第5層：意図・推論



深さが増えるほど、より抽象度の高い特徴を自動で学習できる。

PythonによるEXCEL・WORD連携、AIプログラミング

AIプログラミングとは何か？

世界で最も使われている言語 PYTHON　人気の理由

- **学び易い**
  - 直ぐに実行できる
  - 文法がシンプル
  - 読みやすい
  - 言語機能の一貫性
- **最新の有名アプリでの実績**
  - Instagram、Google　Maps、Gmail、You Tube　など
- **将来性が高い**
  - **Webアプリケーション開発**
  - **機械学習と人工知能**

AIプログラミングとは何か？

- プログラミングは、コンピュータに特定の作業をさせるための命令を、プログラミング言語を使って書くことです。

〈やりたい事〉
データ集計

→

〈処理手順〉
1. データ入力
2. データ集計
3. 結果を出力

→

〈プログラム〉
```
data = [1,2,3]
total = sum(data)
print(total)
```

→

〈実行〉
6

- AIプログラミングとは生成AIを使ってプログラミングを行う。
  - 生成AIの魅力
    - ✓ 話しかけるだけでコードが書ける
    - ✓ 難しい所を助けてくれる
    - ✓ 時間を節約できる

AIエージェント

- 目的を伝えるだけで、自分で考えて仕事を進めるAI

- 3つのAIエージェント
  - タスク分解
    - あなたはタスク分解が得意なAIエージェントです。
      出力は必ず箇条書きにしてください。
  - 文書の要約
    - あなたは要約が得意なAIエージェントです。100字以内でまとめて。
  - 文書自動生成
    - あなたは文書作成が得意なAIエージェントです。
      テーマについて、構造化されたレポートを作成してください。

OpenAIの言語モデルによる AIプログラミング

OpenAIの言語モデルは、自然言語を理解し、ディープラーニングによって文章やコードを生成するAIです。日本語で指示を送ると、その内容を解析し、必要な文章やプログラムを自動で作成します。

1. 長所

- ✓ **自然言語処理:** 人間らしい自然な文章を生成でき、会話や説明文を滑らかに作成できます。
- ✓ **応用範囲の広さ:** 文書作成、プログラミング支援、アイデア出し、学習サポートなど、多様な場面で利用できます。

2. 短所・課題

- ✓ **誤った情報の生成:** もっともらしく誤った内容を出すことがあるため、内容の確認が重要です。
- ✓ **最新情報の制限:** モデルの学習時点より新しい出来事や情報には対応できない場合があります。

OpenAIの言語モデルを使ったAIによるプログラミング

日本語でプログラム生成の要求を行うと言語モデルがプログラムを自動生成します。
（ChatGPTサービスを使っているのではなく、GPTモデルをAPIを通じて利用しています。）

GPTとは、OpenAIが開発した大規模言語モデル（LLM）で、正式名称は Generative Pre-trained Transformer。
大量のテキストを事前学習し、文脈を理解して人間のような文章を生成できるAIモデルのこと。

AIプログラミング
素数を算出するプログラムの生成

- 指示
  - 素数を生成するプログラムを提案して下さい。

- 素数　1と自分自身以外には約数を持たない自然数
  - 素数は数学の基礎として非常に重要。例えば、素数は暗号技術や数論研究において重要な役割を果たしている。

黄色の数が素数

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

講習内容

- 一日目
  - Pythonの基本プログラム文法・仕様
  - Pythonによる　Excel連携
- 二回目
  - Pythonによる　Excel連携
  - Pythonによる　Word連携
  - OpenAIの言語モデルによるAIプログラミング

教材ダウンロードサイト

- 教材ダウンロードサイト
  - https://kkotani.site/
  - 第一回、第二回のテキストはPDFデータで提供
  - プログラムはzip圧縮されている。
    - Ex、Xlの2つのフォルダーにプログラムは格納
      - Ex_AI　　Python基本プログラム
      - Xl_word_AIprog　　Excel&word&AI連携プログラム

プログラミングのポイント

python™

演算子

演算子	例	説明
+	a + b	足し算
-	a - b	引き算
*	a*b	掛け算
/	a/b	割り算
//	a//b	aをbで割った商の整数値
%	a%b	aをbで割った時の割り切れなかった余り
**	a**n	aをn回掛けた数（べき乗）

Ex1 計算してみよう

```
Python 3.7.3 (/usr/bin/python3)
>>> 4+2
6

>>> 4-2
2

>>> 4*2
8

>>> 4 /2
2.0

>>> 4 ** 2
16

>>> (4 + 2) * 3
18

>>>
```

```
Python 3.7.3 (/usr/bin/python3)
>>> x = 5
>>> x
5

>>> x * 2
10

>>> x / 2
2.5

>>> x -2
3

>>> x + 2
7

>>> x = 9
>>> x
9

>>> x = x -7
>>> x
2

>>>
```

比較演算子

演算子	例	説明
==	a == b	2つの値がイコールの時に「正」
!=	a != b	2つの値がイコールでない時に「正」
>	a > b	左の値が右の値より大きい時に「正」
<	a < b	左の値が右の値より小さい時に「正」
>=	a >= b	左の値が右の値以上の時に「正」
<=	a <= b	左の値が右の値以下の時に「正」

Ex2 試してみよう

```
Python 3.7.3 (/usr/bin/python3)
>>> x = 5
>>> x
5

>>> x > 2
True

>>> x <= 8
True

>>> x < 2
False

>>> x == 5
True

>>> x != 5
False

>>> x != 3
True

>>> |
```

条件に合う場合の処理
と合わない場合の処理

- If～else文

```
if　条件式:
    処理1・・・・・
    処理2・・・・・
else:
    処理3・・・・・
    処理4・・・・・
```

条件式がTrueのときに実行します

条件式がFalseのときに実行します

同じスペース数で字下げ（インデント）されている部分が同一ブロックと見なされる。

python™

Ex3 偶数・奇数の判定

- 2で割って余りで偶数・奇数を判定

```
from random import randint
num = randint(0,100)

#以下に入力して下さい。
if num % 2 == 0:
    result = '偶数'
else:
    result = '奇数'

print(num,result)
```

条件を満たす間、繰り返し実行する

- While　文

```
while　条件式:
        処理1・・・・・
        処理2・・・・・
        処理3・・・・・
```

条件式がTrueの間、繰り返し実行される

python™

Ex4 WHILE文

- countの値が10以上になるまで処理を繰り返す

```
count = 1

while count <= 10:
    print(count)
    count += 1
```

COUNT = COUNT + 1

- 1から10までの合計を計算してみよう。

繰り返し実行する処理

- Forループ　文

```
for 変数 in オブジェクト:
    処理1・・・・・
    処理2・・・・・
    処理3・・・・・
```

繰り返し実行される

python™

Ex5　FORループ文

- 1文字(もじ)づつ取出(とりだ)す

```
text = '☆□◆△▼※◎●'
for char in text:
    print(char)
```

- 変数(へんすう)を順番(じゅんばん)に取出(とりだ)す

```
dan = 3
for i in range(10):
    print(f"{dan} × {i} = {dan * i}")
```

python™

Ex6-1　リスト

- リストとは、
  - いろいろな種類のデータを順番に入れておける入れ物です。
  - 文字や数字、文章など、どんなデータでも一緒に入れられます。
  - 入れた順番はそのまま記録され、あとから好きな場所のデータを取り出したり、入れ替えたりできます。
  - 角括弧 [ ] 内に、リスト要素をカンマ(,)で区切って並べます。

```
text = [12345,3.14,'✡','□','◆','△','▼','※','◎','●','こんにちは']
for i in range(len(text)):
    print(text[i])
```

python™

Ex6-2　リスト

- リストの入れ物に色の名前が入っています。
- 順番に名前を取り出して、色を変える。

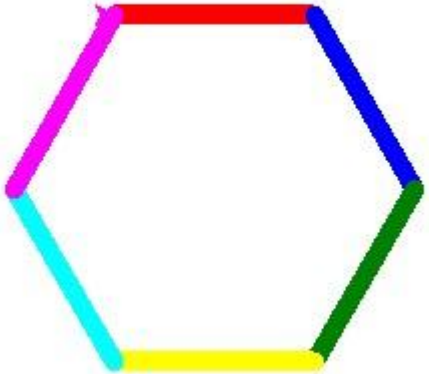
```
import turtle

colors = ["red", "blue", "green", "yellow", "cyan", "magenta"]

t = turtle.Turtle()
t.width(10)
#t.speed(1)

for c in colors:
    t.color(c)
    t.forward(100)
    t.right(60)   # 六角形になるよ

turtle.done()
```



python™

Ex7 辞書型　DICT型オブジェクト

- 辞書は、キー（名前）と値のペア
  - 変数 ＝ {キー1：値1、キー2：値2・・・・・・}
- 参照、追加、変更、削除

```
>>> tel = {'kotani':4098, 'watanabe': 4139 }
>>> tel['sasaki']= 4127
>>> tel
{'kotani': 4098, 'watanabe': 4139, 'sasaki': 4127}
>>> tel['watanabe']
4139
>>> del tel['watanabe']
>>> tel
{'kotani': 4098, 'sasaki': 4127}
>>> tel['kanou']=3000
>>> tel
{'kotani': 4098, 'sasaki': 4127, 'kanou': 3000}
>>> sorted(tel.keys())
['kanou', 'kotani', 'sasaki']
>>> 'kotani' in tel
True
```

辞書型　DICT型オブジェクト
Ex7　英単語の出現回数を数えてみよう

- 単語を調べるのに辞書型は都合が良い。

```
text = """
Keep on asking, and it will be given you:
Keep on asking, and you will find:
Keep on knocking, and it will be opened to you:
for everyone asking receives, and everyone seeking finds.
and to everyone knocking, it will be opened.
"""

text = text.replace(":","")
text = text.replace(",","")
text = text.replace(".","")
words = text.split() #空白で区切ってリスト型作成

counter = {}

#以下にロジックを入力して下さい。
for w in words:
    ws = w.lower()
    if ws in counter:
        counter[ws] += 1
    else:
        counter[ws] = 1  #辞書に無ければ値を1としてキー追加


for k,v in sorted(counter.items()):
    if v >= 3:
        print(k,v)
```

3回以上の単語を抽出する。

Ex8　関数　最大値を返す

- 関数はdefブロックで定義。括弧()内に引数を指定する。戻り値があればreturn文で返す。

```
def max_n(numx):
    #数字のリストを受け取り、その中の最大値を返す関数。
    max_v = numx[0]  #リストの最初の要素で初期最大値を設定
    #リスト内の数字をループ
    #以下にロジックを入力して下さい。
    for number in numx:
        if number > max_v:  #より大きい数があれば更新
            max_v = number
    return max_v            #最大値を返す

num = [3, 1, 4, 1, 5, 9, 2, 6, 5]
print(max_n(num))
```

Ex9 クラス　　クラスとは何か？

- クラスとは
  - ✓ もの（オブジェクト）を作るための説明をまとめたもの。
- 攻撃用のキャラクターで説明すると
  - クラス：　キャラクターに必要なものを**ひとまとめにしたセット**
    - 名前　　キャラの名札（なふだ）
    - HP　　体力メーター
    - 攻撃指示　　attack()
  - このクラスを使って攻撃キャラクターを作ってみる

Ex9 クラス　　攻撃キャラクター

クラスの定義

```
class Player:
    def __init__(self, name, hp):
        self.name = name   # データ（属性）
        self.hp = hp

    def attack(self):
        print(self.name, "は攻撃した！")

# クラスからオブジェクトを作る
p1 = Player("Kazumi", 100)
p2 = Player("Kaoru", 80)

print(p1.name, p1.hp)
p1.attack()

print(p2.name, p2.hp)
p2.attack()
```

キャラクターの名札
体力メーター

攻撃指示

python™

Ex10 OpenAI言語モデルによるAIプログラミング

- プログラム生成の入力例を以下に示す。
  - 素数を生成するプログラムを提案して下さい。
  - さらに高速な素数生成プログラムを提案して下さい。
  - 数当てゲームを提案して下さい。
  - カウントダウンプログラムを生成して下さい。
  - 可愛い猫の画像を取得するプログラムを生成して下さい。
  - excel上にopenpyxlを使用して、九九の計算表を生成するプログラム
  - openpyxlを使ってExcelデータを読んでword文書に変換するプログラム。

    など

- 言語モデルが自動生成したpythonプログラムを以下の名前で格納。

  Ex10_prg数字.py

PYTHONによる
EXCEL･WORD　連携

PiPy　　Python Package Index
OPENPYXLモジュール

- OpenPyXLとは、Excelファイルを Pythonで操作するためのライブラリ
  - Excelファイルのシートやセルからデータを取り出したり、セルにデータ書き出したり様々な操作が可能です。
  - 拡張子が「.xlsx」であること。(.xlsは不可、Excel97-2003)
    - 本演習では　LibreOffice Calc　を使用。

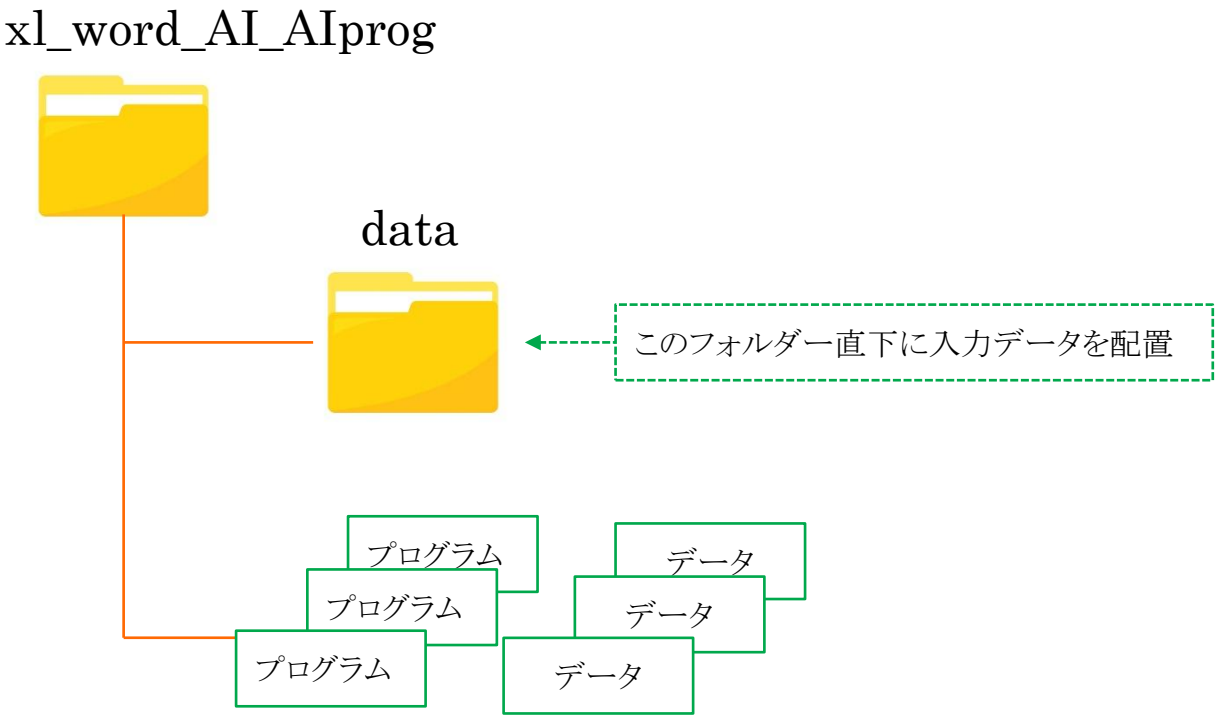
Excel		[openpyxl]
ブック	═══	Workbook オブジェクト
ワークシート	═══	Worksheet オブジェクト
セル	═══	Cell オブジェクト

Python & Excel
なぜ良いのか？

- Excelでコピー＆ペーストを使った多くの面倒な手作業

- Pythonのopenpyxl機能
  - 実感　大変強力で作業軽減が図れる。
  - 複数ファイルを簡単に操作できる。
    - EXCELを立ち上げる必要がない
  - プログラムが別ファイルになるので再利用が容易にできる。
  - 豊富なライブラリィと連携することで自動化がし易い。

本実習のフォルダー構成



XN1　EXCELブック読み込み

- openpyxlによるブックの読み込み

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
print(wb.sheetnames)
```

プログラムを格納するディレクトリの絶対パスを得る

dataフォルダーから売上データ読み込み

読み込んだ売上データのシート名一覧印刷

XN2 EXCELブックのシートの指定

- 読み込むシートの指定方法（3種類）

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb.active
print(ws.title)
ws1 = wb['4月売上']
print(ws1.title)
ws2 = wb.worksheets[0]
print(ws2.title)
```

dataフォルダーから売上データ読み込み

①現在Excelが指定しているシートを指定

②シート名でシートを指定

③シート番号でシートを指定。0番から始まる。

XN3 セル値の読み込み

セル値の読み込み

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['4月売上']
c1 = ws['A4']
print(c1.value)
c1 = ws['B4']
print(c1.value)
```

dataフォルダーから売上データ読み込み

シート名でシートを指定

A4、B4のセル値の読み込み

リストを使ったセル値の読み込み

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'),data_only=True)
ws = wb['4月売上']
col = ['A4','B4','C4','D4','E4','F4']

for i in col:
    c = ws[i]
    print(c.value)
```

dataフォルダーから売上データ読み込み。
計算結果の数値を取得する。

セルの場所をリストで指定

リストからセルの場所を順次取り出し、セル値の読み込み

XN4 セルに値を書き込む

- 売上データの4月売上の最終行に値を書き込む

```
import openpyxl
import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['4月売上']
c1 = ws['A10']
c1.value = datetime.datetime(2026,4,27)
c2 = ws['B10']
c2.value = "kotaniシステムラボ"
c3 = ws['C10']
c3.value = "商品B"
c4 = ws['D10']
c4.value = 3800
c5 = ws['E10']
c5.value = 12
c6 = ws['F10']
c6.value = '=D10*E10'
wb.save(parent.joinpath('XN4_売上データ.xlsx'))
```

dataフォルダーから売上データ読み込み

シート名でシートを指定

A10～F10のセルに値を書き込む。
F10は計算式を書き込んでいる。

売上データ保存

XN5 セルに値を書き込む

- 4月売上の最終行に値を表示形式付きで書き込む

```
import openpyxl
import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['4月売上']
c1 = ws['A10']
c1.value = datetime.datetime(2026,4,27)
c1.number_format = 'yyyy/m/d'
c2 = ws['B10']
c2.value = "kotaniシステムラボ"
c3 = ws['C10']
c3.value = "商品B"
c4 = ws['D10']
c4.value = 3800
c4.number_format = '#,##0'
c5 = ws['E10']
c5.value = 12
c6 = ws['F10']
c6.value = '=D10*E10'
c6.number_format = '#,##0'

for i,row in enumerate(ws.iter_rows()):
    col = 'D' + str(i+4)
    c = ws[col]
    c.number_format = '#,##0'
    col = 'F' + str(i+4)
    c = ws[col]
    c.number_format = '#,##0'
wb.save(parent.joinpath('XN5_売上データ.xlsx'))
```

dataフォルダーから売上データ読み込み

シート名でシートを指定

西暦/月/日
yyyy/mm/dd
yyyy/mm/dd hh:mm:ss 等の指定が可能

データが0の場合、#では何も表示されない。
3桁づつカンマで表示される。

enumerate　インデックス番号, 要素を順番に順に数え上げる

D列、F列の表示形式を全て設定

売上データ保存

XN6 EXCELファイルを一行ずつ読み込む

- 「4月売上」シートを一行ずつ読み込む

```
import openpyxl
import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['4月売上']

for row in ws.iter_rows(min_row=4):
    print(row)
```

dataフォルダーから売上データ読み込み

シート名でシートを指定

for文で1行ずつ読み込む書式

for　row　in　シート変数名.iter_rows(min_row=行番号,min_col=列番号)：
　　処理

XN7 EXCELファイルを一行ずつ読み込む

- 「4月売上」のセル値を取出す

```
import openpyxl
import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['4月売上']

for row in ws.iter_rows(min_row=4):
    if row[0].value  is None:
        break
    value_list = []
    for c in row:
        value_list.append(c.value)
    print(value_list)
```

dataフォルダーから売上データ読み込み

シート名でシートを指定

1列目が空欄の場合にループを抜ける

空のリストを作成

1行の中のセル値を一つずつリストに追加

python™

XN8 EXCELファイルに一行ずつ書き込む

```
import openpyxl
from datetime import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws = wb['6月売上']

data_list = [
    [datetime(2026,6,1), '株式会社 AA商店', '商品A', 7200, 30],
    [datetime(2026,6,3), 'BB企画 有限会社', '商品A', 7200, 15],
    [datetime(2026,6,9), 'kotaniシステムラボ', '商品C', 1200, 20],
]

row_num = 4
for row in data_list:
    c1 = ws['A' + str(row_num)]
    c1.value = row[0]
    c1.number_format = 'yyyy/m/d'
    c2 = ws['B' + str(row_num)]
    c2.value = row[1]
    c3 = ws['C' + str(row_num)]
    c3.value = row[2]
    c4 = ws['D' + str(row_num)]
    #c4.number_format = '#,##0'
    c4.value = row[3]
    c5 = ws['E' + str(row_num)]
    c5.value = row[4]
    c6 = ws['F' + str(row_num)]
    c6.value = '=D' + str(row_num) + '*E' +str(row_num)
    #c6.number_format = '#,##0'
    row_num += 1

wb.save(parent.joinpath('XN8_売上データ.xlsx'))
```

dataフォルダーから売上データ読み込み

シート名で「6月売上」シートを指定

書き込むデータのリスト

リストから1行ずつデータを取出して書き込み

売上データ保存

シートの操作

XN9 シートの挿入、シートの削除

- シートの挿入

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ.xlsx'))
ws_new1 = wb.create_sheet(index=0)
ws_new2 = wb.create_sheet()
print(ws_new1.title)
print(ws_new2.title)

wb.save(parent.joinpath('XN9_売上データ.xlsx'))
```

dataフォルダーから売上データ読み込み

・シートの番号を指定する。0から始る。0の場合、先頭。
・indexを指定しない場合、末尾にシート挿入。

- シートの削除

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('XN9_売上データ.xlsx'))
ws = wb.worksheets[0]
wb.remove(ws)

wb.save(parent.joinpath('XN9_売上データ.xlsx'))
```

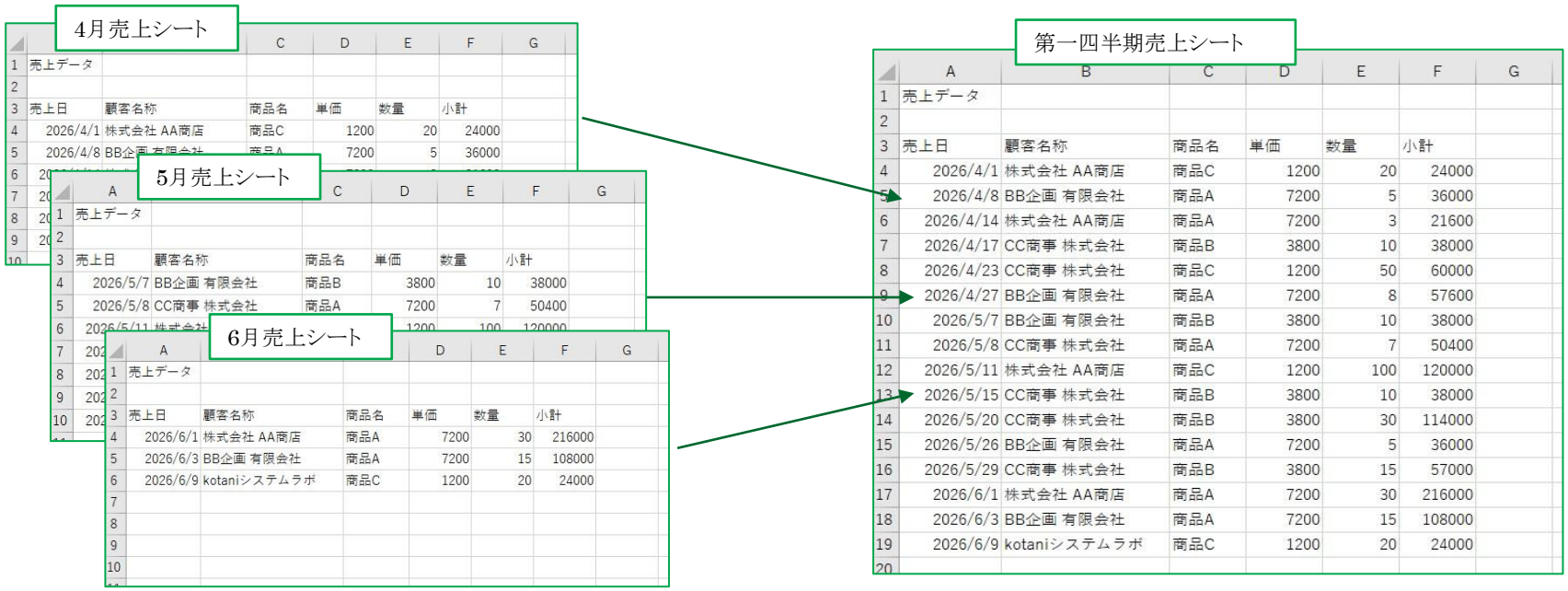
売上データ読み込み

・シート番号を指定して削除。
・シート名指定で削除は以下。
ws=wb[シート名]
wb.remove(ws)

複数のシートをまとめる

XN10　複数のシートを一つにする

- dataフォルダーの売上データ_6月入力ブックの中の4月売上シート、5月売上シート、6月売上シートをまとめて第一四半期売上シートを新規に作成する。
- 第一四半期売上シートは先頭のシート位置に作成する。



4月売上シート

	A	B	C	D	E	F	G
1	売上データ						
2							
3	売上日	顧客名称	商品名	単価	数量	小計	
4	2026/4/1	株式会社 AA商店	商品C	1200	20	24000	
5	2026/4/8	BB企画 有限会社	商品A	7200	5	36000	
6	20						
7	20						
8	20						
9	20						
10							

5月売上シート

	A	B	C	D	E	F	G
1	売上データ						
2							
3	売上日	顧客名称	商品名	単価	数量	小計	
4	2026/5/7	BB企画 有限会社	商品B	3800	10	38000	
5	2026/5/8	CC商事 株式会社	商品A	7200	7	50400	
6	2026/5/11	株式会社		1200	100	120000	
7	202						
8	202						
9	202						
10	202						

6月売上シート

	A	B	C	D	E	F	G
1	売上データ						
2							
3	売上日	顧客名称	商品名	単価	数量	小計	
4	2026/6/1	株式会社 AA商店	商品A	7200	30	216000	
5	2026/6/3	BB企画 有限会社	商品A	7200	15	108000	
6	2026/6/9	kotaniシステムラボ	商品C	1200	20	24000	
7							
8							
9							
10							

第一四半期売上シート

	A	B	C	D	E	F	G
1	売上データ						
2							
3	売上日	顧客名称	商品名	単価	数量	小計	
4	2026/4/1	株式会社 AA商店	商品C	1200	20	24000	
5	2026/4/8	BB企画 有限会社	商品A	7200	5	36000	
6	2026/4/14	株式会社 AA商店	商品A	7200	3	21600	
7	2026/4/17	CC商事 株式会社	商品B	3800	10	38000	
8	2026/4/23	CC商事 株式会社	商品C	1200	50	60000	
9	2026/4/27	BB企画 有限会社	商品A	7200	8	57600	
10	2026/5/7	BB企画 有限会社	商品B	3800	10	38000	
11	2026/5/8	CC商事 株式会社	商品A	7200	7	50400	
12	2026/5/11	株式会社 AA商店	商品C	1200	100	120000	
13	2026/5/15	CC商事 株式会社	商品B	3800	10	38000	
14	2026/5/20	CC商事 株式会社	商品B	3800	30	114000	
15	2026/5/26	BB企画 有限会社	商品A	7200	5	36000	
16	2026/5/29	CC商事 株式会社	商品B	3800	15	57000	
17	2026/6/1	株式会社 AA商店	商品A	7200	30	216000	
18	2026/6/3	BB企画 有限会社	商品A	7200	15	108000	
19	2026/6/9	kotaniシステムラボ	商品C	1200	20	24000	
20							

複数のシートをまとめる
XN10 複数のシートを一つにする

```
import openpyxl
import datetime
from pathlib import Path

parent = Path(__file__).resolve().parent
wb = openpyxl.load_workbook(parent.joinpath('data/売上データ_6月入力.xlsx'))
first_sheet = True
row_list = []

for ws in wb.worksheets:
    if first_sheet:
        start_row = 1
        first_sheet = False
    else:
        start_row = 4

    for row in ws.iter_rows(min_row=start_row):
        if row[0].row > 3 and row[0].value is None:
            break
        value_list = []
        for c in row:
            value_list.append(c.value)
        row_list.append(value_list)

ws_new = wb.create_sheet(title = '第一四半期売上')
num = 1
for row in row_list:
    ws_new.append(row)
    if num > 3:
        ws_new.cell(num,1).number_format = 'yyyy/m/d'
        ws_new.cell(num,6).value = '=D' + str(num) + '*E' + str(num)
    num += 1

top = 1 - len(wb.worksheets)
wb.move_sheet(ws_new, top)

wb.save(parent.joinpath('XN10_第一四半期売上.xlsx'))
```

dataフォルダーから売上データ_6月入力読み込み

空のリストを作成

・最初のシートはヘッダーを含め1行目から読む
・2番目のシートはヘッダーを除外して4行目から読む
・row_listに順次値を格納

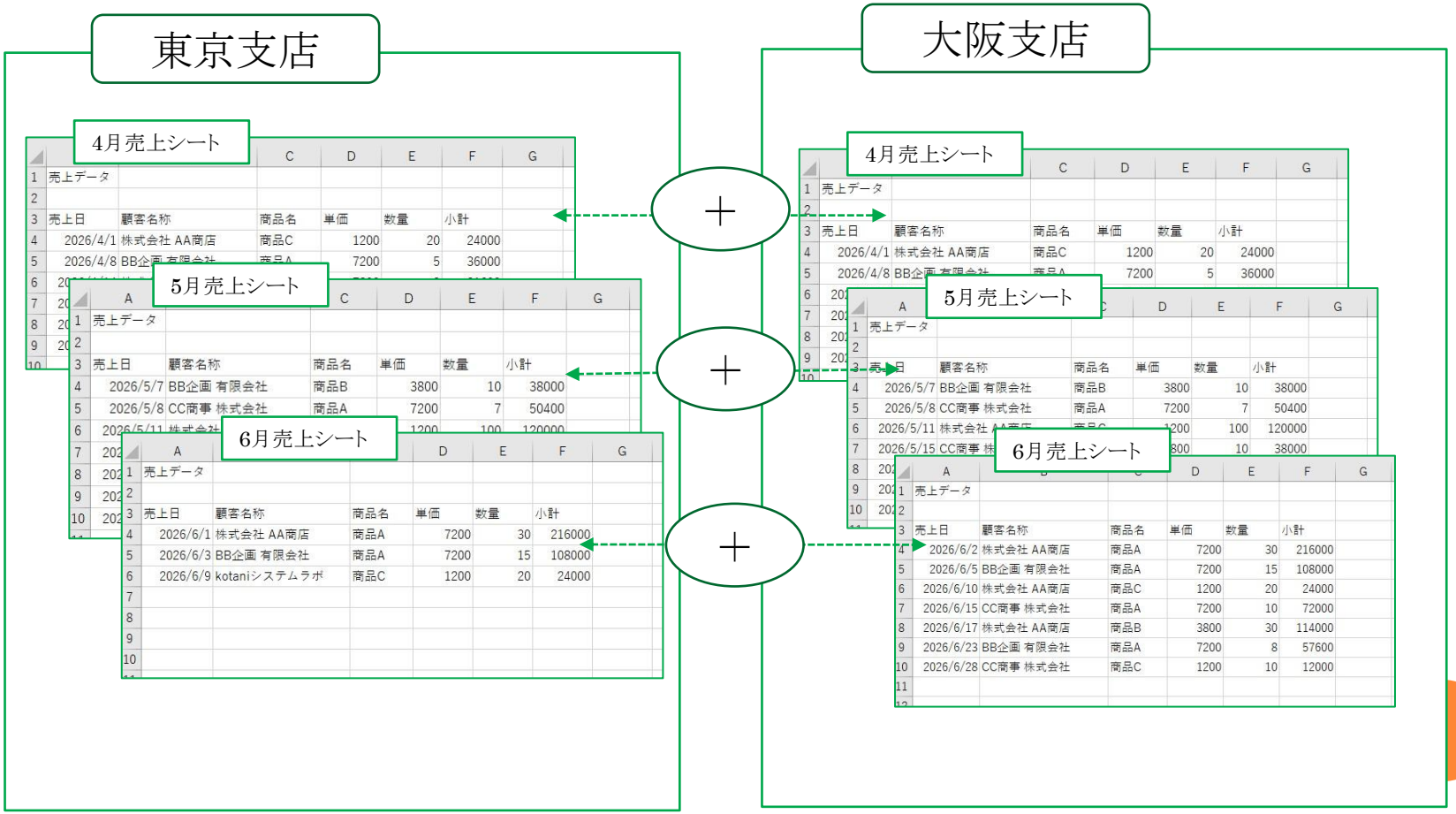
第一四半期売上シートを末尾に作成

・第一四半期売上シートにrow_listの内容を順次格納
・日付の表示形式設定、計算式設定

・先頭シート位置の移動量を計算し、先頭シート位置に移動
・新規ブック名を指定して格納

複数のシートをまとめる
XN11 複数のブックをシート名ごとにまとめる

- data/売上全支店フォルダーにある複数のブックを同名シートごとにまとめて、新しいブックに保存する。



複数のシートをまとめる
XN11 複数のブックをシート名ごとにまとめる

```
import openpyxl
from pathlib import Path

parent = Path(__file__).resolve().parent
wb_list = []

for file in Path(parent.joinpath("data/売上全支店")).glob("*.xlsx"):
    wb = openpyxl.load_workbook(file)
    wb_list.append(wb)
print(wb_list)
wb_new = openpyxl.Workbook()

sheet_names = wb_list[0].sheetnames
for sheet_name in sheet_names:
    ws_new = wb_new.create_sheet(sheet_name)
    is_first_book = True
    row_list = []

    for wb in wb_list:
        ws = wb[sheet_name]
        if is_first_book:
            start_row = 1
            is_first_book = False
        else:
            start_row = 4

        for row in ws.iter_rows(min_row=start_row):
            if row[0].row > 3 and row[0].value is None:
                break
            value_list = []
            for c in row:
                value_list.append(c.value)
            row_list.append(value_list)

    row_num = 1
    for row in row_list:
        ws_new.append(row)
        if row_num > 3:
            ws_new.cell(row_num, 1).number_format = "yyyy/m/d"
            ws_new.cell(row_num, 6).value = "=D" + str(row_num) + "*E" + str(row_num)
        row_num += 1
ws_first = wb_new.worksheets[0]
wb_new.remove(ws_first)
wb_new.save(parent.joinpath("XN11_売上データ全国.xlsx"))
```

data/売上全支店フォルダー内のexcelブック取得

最初のブックからシート名取得

格納用の新規シート作成

各ブック名の同一シート名のデータをrow_listに読み込む

row_list を順次読んで書式設定を行い書き込む

不要シートを削除して保存

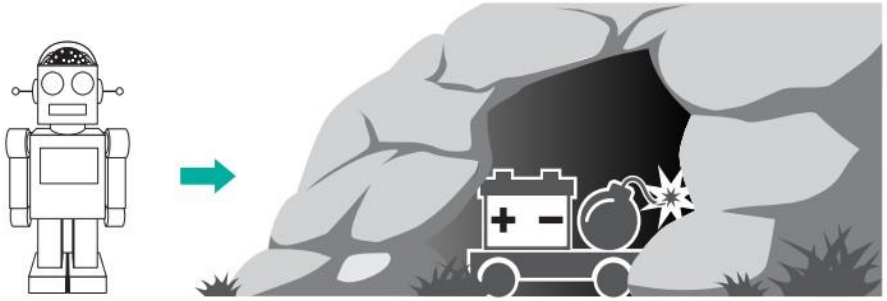
終わり

経歴

- 富士通で32年間ソフトウェア開発（大型OS、コンピュータグラフィックス、CAD等）を中心技術者として従事。
- 8年間トヨタ傘下の共和レザーで基幹情報システム開発、システム監査に従事。
- 2年間ソフトウェア会社（CSE)にて認証ソフト開発コンサル、その後言語変換業務開発コンサルに従事。
- 東海大学海洋学部非常勤講師(2001年～2006年）
- email
  - ksys@kkotani.net

フレーム(枠)問題

- フレーム問題
  - 何を気にして、何を気にしなくていいのかの『枠』を決めるのがむずかしい。
- 爆弾とロボット
  - 洞窟の中にバッテリーがあり、その上には時限爆弾が仕掛けられています。ロボットは「バッテリーを持ち帰る」という指示を受けている。



- ロボット1号(考えなさすぎ)

  バッテリーだけ見て、爆弾のことを考えなかった → 外に出てから爆発してしまった
- ロボット2号(考えすぎ)

  「爆弾は？」「洞くつは？」「地面は？」となんでも心配しすぎて動けない → 何もできずに爆発
- ロボット3号(無視することを決められない)

  「何を無視すればいいの？」をずっと考え続けてしまう → やっぱり動けず爆発

AIは「大事なこと」と「気にしなくていいこと」をうまく選べない。
そのせいで、正しい判断ができなくなる。