

# 情報処理（11週目）

## Python入門

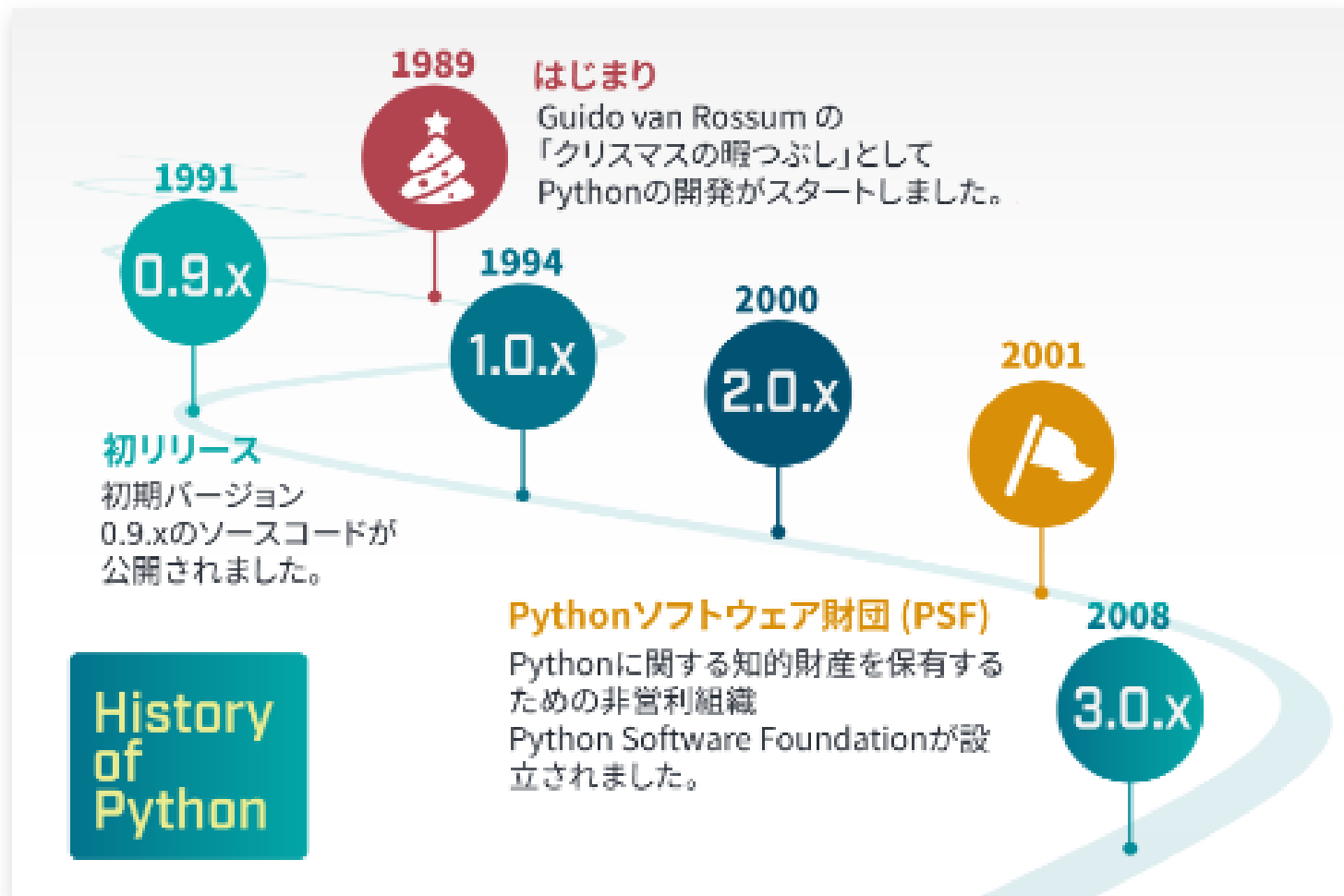
王 忠奎

立命館大学 ロボティクス学科

# 講義の流れ

- Pythonの歴史と特徴
- Pythonを用いたプログラミングの環境設定
- Print関数
- 四則演算
- データ型
- 変数
- コメントとコメントアウト
- キーボードから入力関数input
- レポート課題

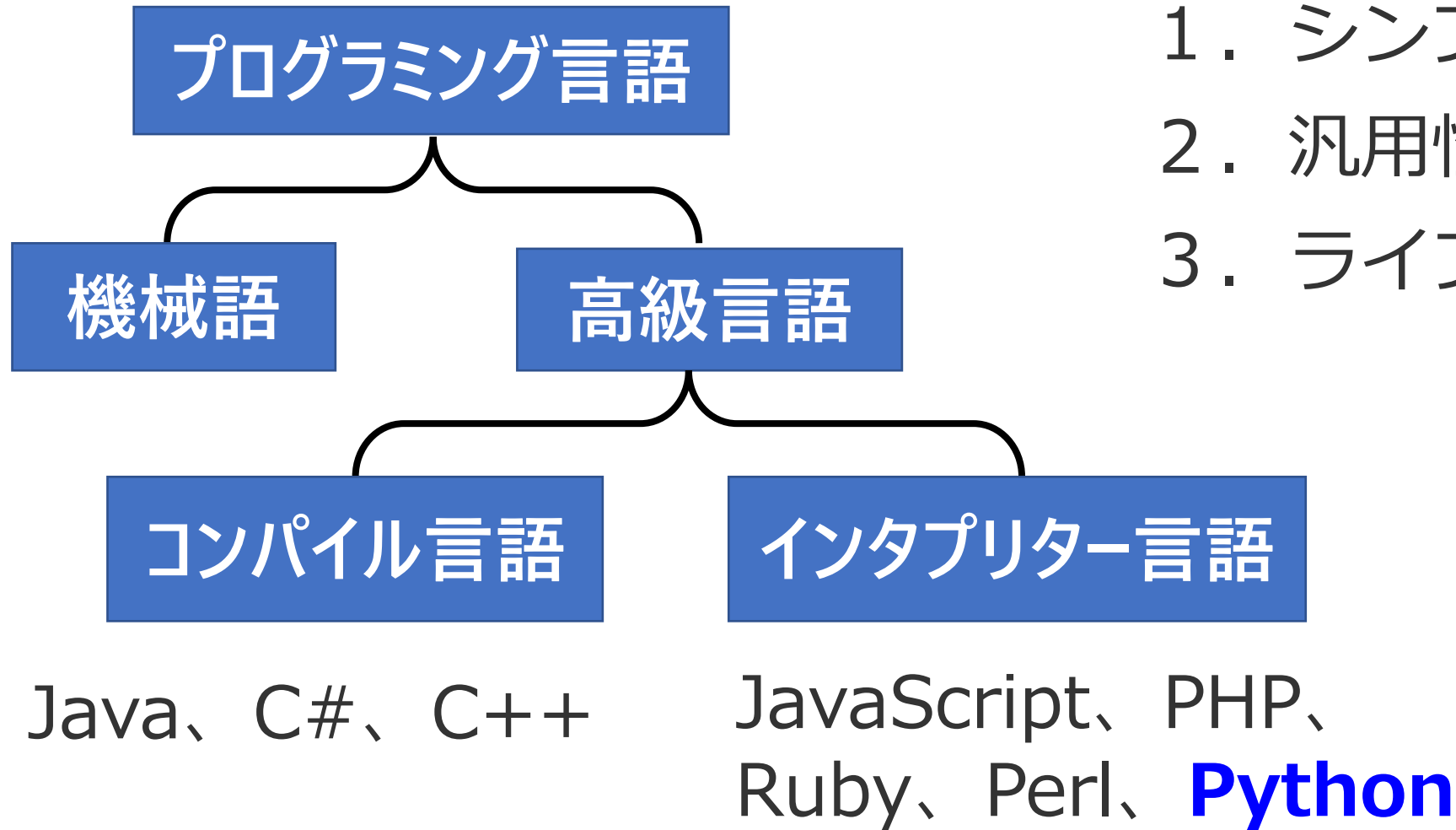
# Pythonの歴史



Guido van Rossum

Pythonという名称は、イギリスで大人気だったテレビ番組「空飛ぶモンティ・パイソン」(Monty Python's Flying Circus) が由来です。

# Pythonの特徴















1. シンプル
2. 汎用性が高い
3. ライブラリが多い

<https://www.tiobe.com/tiobe-index/>

# プログラミング言語ランキング

TIOBEインデックスにより

Jun 2024	Jun 2023	Change	Programming Language		Ratings	Change
1	1		 Python		15.39%	+2.93%
2	3	▲	 C++		10.03%	-1.33%
3	2	▼	 C		9.23%	-3.14%
4	4		 Java		8.40%	-2.88%
5	5		 C#		6.65%	-0.06%
6	7	▲	 JavaScript		3.32%	+0.51%
7	14	▲▲	 Go		1.93%	+0.93%
8	9	▲	 SQL		1.75%	+0.28%
9	6	▼	 Visual Basic		1.66%	-1.67%
10	15	▲▲	 Fortran		1.53%	+0.53%
11	11		 Delphi/Object Pascal		1.52%	+0.27%
12	19	▲	 Swift		1.27%	+0.33%

# 環境設定

- Python 3のインストール
  - <https://www.youtube.com/watch?v=pwEt50n2K1c>
  - <https://www.python.org/downloads/>
  - 注意点：「**Add Python 3.9 to PATH**」をチェック
  - pipツールも一緒にインストール
  - Python version：Python 3.9.4
- 総合開発環境（IDE）PyCharmのインストール
  - <https://www.jetbrains.com/pycharm/download/#section=windows>
  - 無料バージョンCommunity version
  - 注意点：「Add launchers dir to the PATH」をチェック
  - Python 3.9.4を「Existing interpreter」に追加

# インストール確認

- コマンドプロンプトを開き、python --version コマンドを入力してみましょう。

C:\ コマンドプロンプト

```
Microsoft Windows [Version 10.0.22000.675]  
(c) Microsoft Corporation. All rights reserved.
```

```
C:\Users\steve>python --version  
Python 3.9.4
```

```
C:\Users\steve>
```

# Pythonインタプリタ

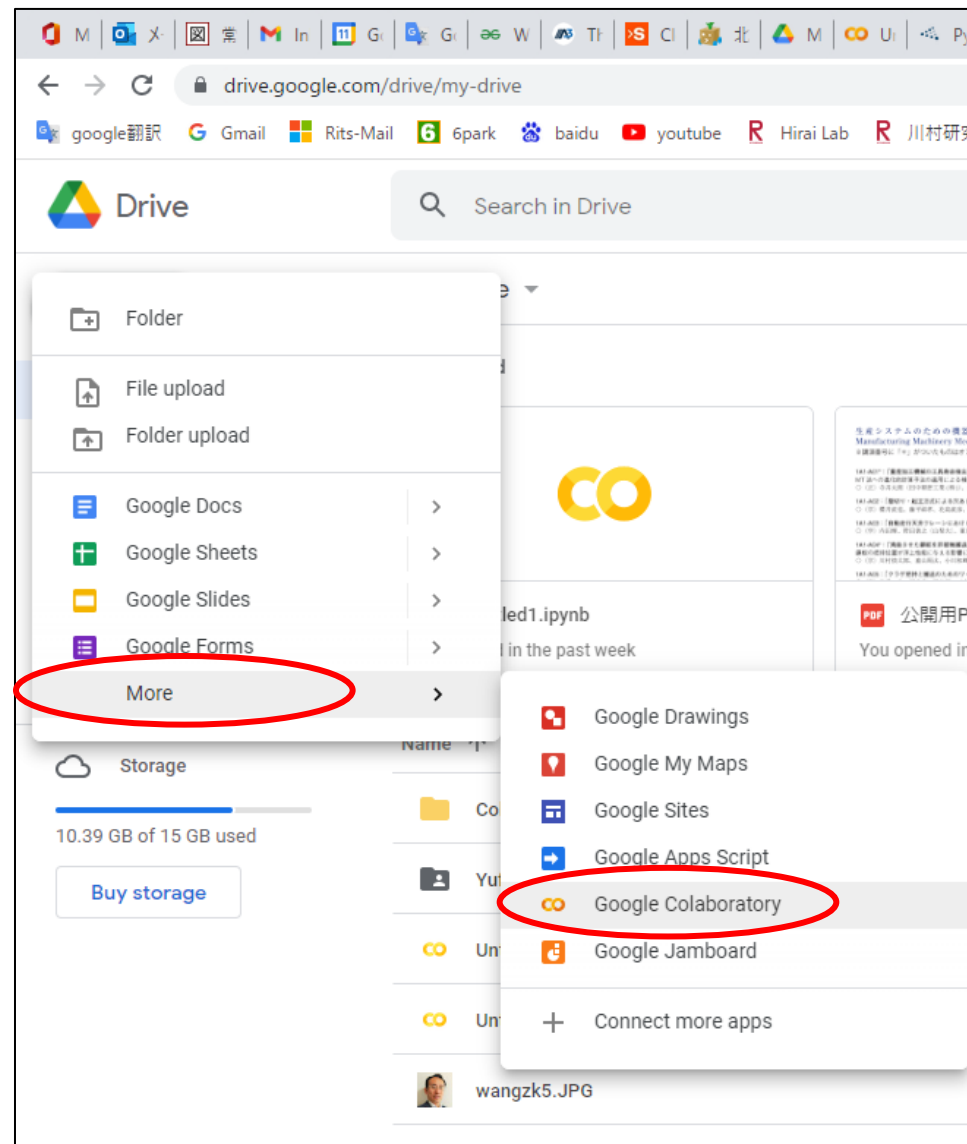
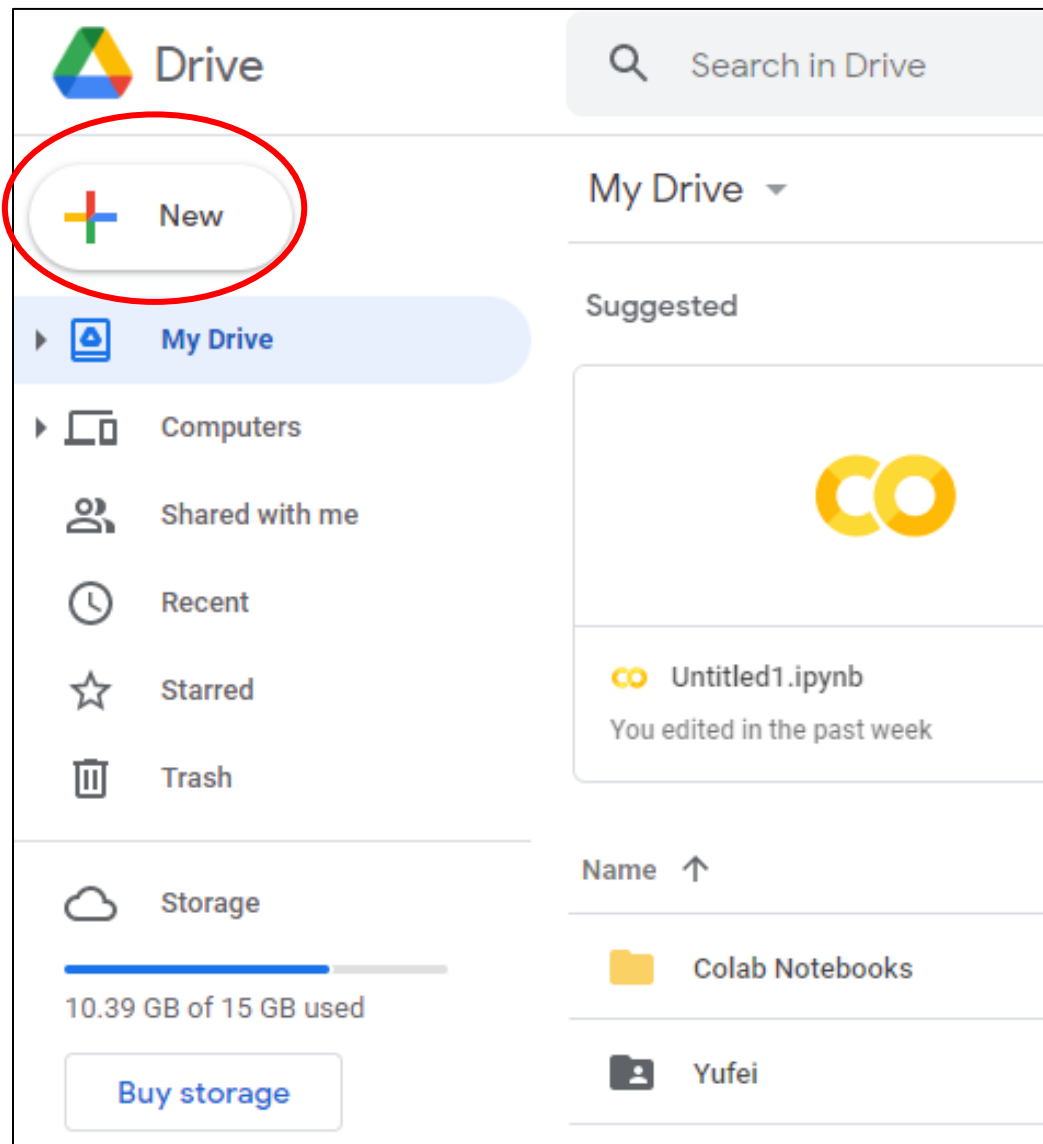
- Pythonインタプリタは「対話モード」とも呼ばれ、ユーザーとPythonが対話的にプログラミングを行うことができる

```
コマンド プロンプト - python
Microsoft Windows [Version 10.0.22000.675]
(c) Microsoft Corporation. All rights reserved.

C:\Users\steve>python
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr 6 2021, 13:40:21) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1+2
3
>>> print('hello world!')
hello world!
>>>
```



# オンライン開発環境：Google Colaboratory



# オンライン開発環境：Google Colaboratory



Untitled2.ipynb ☆

コメント

共有



ファイル 編集 表示 挿入 ランタイム ツール ヘルプ すべての変更を保存しました

+ コード + テキスト

✓ RAM  / ディスク

編集



✓ 0 秒 `print('Hello World!')`

Hello World!

✓ 0 秒 完了時間: 23:27



# 最初のPythonプログラム

The screenshot shows the PyCharm IDE interface. The main editor window displays a Python file named `hello.py` with the following code:

```
1 print("Hello World!")
```

The left sidebar shows the project structure for `wangzk`, listing various Python files and images. The bottom panel shows the Run console output:

```
Run: hello x  
"C:\Program Files\Python39\python.exe" C:/Users/steve/PycharmProjects/wangzk/hello.py  
Hello World!  
Process finished with exit code 0
```

The status bar at the bottom indicates the current file encoding is UTF-8 with 4 spaces, and the Python version is 3.9.

# 画面へ出力 — — — print関数

文字列、数値、リスト、辞書などを出力

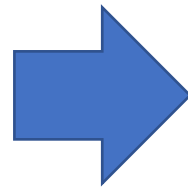
```
print('Hello World!')  
  
print(100)  
  
print([0, 1, 2])  
  
print({'a': 0, 'b': 1, 'c': 2})
```



```
Hello World!  
  
100  
  
[0, 1, 2]  
  
{'a': 0, 'b': 1, 'c': 2}
```

# print関数 – 改行なしで出力（引数end）

```
print('abc')  
print('xyz')  
  
print('abc', end='----')  
print('xyz')  
  
print('abc', end='')  
print('xyz')
```



```
abc  
xyz  
abc----xyz  
abcxyz
```

# print関数 – 複数の変数を出力 (引数sep)

```
i = 100  
print('apple', i, 0.123)  
  
print('apple', i, 0.123, sep='----')  
  
print('apple', i, 0.123, sep='\n')
```



```
apple 100 0.123  
apple---100---0.123  
apple  
100  
0.123
```

# print関数 – 変数を文字列に埋め込んで出力

文字列の途中に変数の値を挿入して出力したい場合、  
下記の三つの方法がある

- パーセント%を使う**printf**形式
- 文字列メソッド**format()**
- **f**文字列（フォーマット文字列）

# print関数 – パーセント%を使う形式

```
s = 'Tanaka'
```

```
i = 25
```

```
print('Tanaka is %d years old.' % i)
```

```
print('%s is %d years old.' % (s, i))
```



```
Tanaka is 25 years old.
```

```
Tanaka is 25 years old.
```



# print関数 – 文字列メソッドformat()を使う形式

```
print('Tanaka is {} years old.'.format(i))
```

```
print('{} is {} years old.'.format(s, i))
```

```
print('{0} is {1} years old. {0},{0},{0}'.format(s, i))
```

```
print('{name} is {age} years old.'.format(name=s, age=i))
```



```
Tanaka is 25 years old.
```

```
Tanaka is 25 years old.
```

```
Tanaka is 25 years old. Tanaka, Tanaka, Tanaka
```

```
Tanaka is 25 years old.
```

# print関数 – f文字列 (フォーマット文字列)

```
print(f'{s} is {i} years old.')  
s = 'abc'  
print(f'right : {s:~>8}')  
print(f'center : {s:~^9}')  
print(f'left : {s:~<8}')  
s = 1234  
print(f'zero padding : {s:08}')
```



```
Tanaka is 25 years old.  
right : _____abc  
center : ___abc___  
left : abc_____  
zero padding : 00001234
```

# 四則演算

- 足し算（加算）：`+` 演算子
- 引き算（減算）：`-` 演算子
- 掛け算（乗算）：`*` 演算子
- 割り算（除算）：`/` 演算子
- 割り算の整数部（整数除算）：`//` 演算子
- 割り算の剰余（余り, mod）：`%` 演算子
- べき乗：`**` 演算子
- `ZeroDivisionError`

# 四則演算 (+, \*, /, //, %)

```
print(f'5 + 7.485 = {5 + 7.485}')
```

```
print(f'8.21 * 3.5 = {8.21 * 3.5}')
```

```
print(f'18 / 2.4 = {18 / 2.4}')
```

```
print(f'15 // 2 = {15 // 2}')
```

```
print(f'15.78 // 2.4 = {15.78 // 2.4}')
```

```
print(f'15 % 2 = {15 % 2}')
```

```
print(f'15.78 % 2.4 = {15.78 % 2.4}')
```



```
5 + 7.485 = 12.485
```

```
8.21 * 3.5 = 28.73500000000000003
```

```
18 / 2.4 = 7.5
```

```
15 // 2 = 7
```

```
15.78 // 2.4 = 6.0
```

```
15 % 2 = 1
```

```
15.78 % 2.4 = 1.38
```

演算子の `//` は商の整数部分だけを取得します。丸め方は負の無限大の方向に丸められます。例えば商の結果が `7.56` であれば `7`、`-4.785` であれば `-5` となります。

# 四則演算 (\*\*, 優先順位, 0での割り算)

```
print(f'10 ** 3 = {10 ** 3}')
```

```
print(f'2 ** 0.5 = {2 ** 0.5}')
```

```
print(f'10 ** -2 = {10 ** -2}')
```

```
s1 = 100 / 10 ** 2 + 2 * 3 - 5
```

```
s2 = 100 / (10 ** 2) + (2 * 3) - 5
```

```
print(f'100 / 10 ** 2 + 2 * 3 - 5 = {s1}')
```

```
print(f'100 / (10 ** 2) + (2 * 3) - 5 = {s2}')
```

```
print(f'10 / 0 = {10 / 0}')
```



```
print(f'10 / 0 = {10 / 0}')
```

```
ZeroDivisionError: division by zero
```

```
10 ** 3 = 1000
```

```
2 ** 0.5 = 1.4142135623730951
```

```
10 ** -2 = 0.01
```

```
100 / 10 ** 2 + 2 * 3 - 5 = 2.0
```

```
100 / (10 ** 2) + (2 * 3) - 5 = 2.0
```

# 四則演算

リスト、タプル、文字列の演算  
(連結)

```
a_list = [0, 1, 2]
b_list = [10, 20, 30]
a_tuple = (0, 1, 2)
b_tuple = (10, 20, 30)
a_str = 'abc'
b_str = 'xyz'

print(f'a_list + b_list = {a_list + b_list}')
print(f'a_tuple + b_tuple = {a_tuple + b_tuple}')
print(f'a_str + b_str = {a_str + b_str}')
```



```
a_list + b_list = [0, 1, 2, 10, 20, 30]
a_tuple + b_tuple = (0, 1, 2, 10, 20, 30)
a_str + b_str = abcxyz
```

# 四則演算

リスト、タプル、文字列の演算（繰り返し）

```
print(f'b_list * 3 = {b_list * 3}')  
print(f'b_tuple * 3 = {b_tuple * 3}')  
print(f'b_str * 3 = {b_str * 3}')  
print(f'a_list + b_list * 3 = {a_list + b_list * 3}')
```



```
b_list * 3 = [10, 20, 30, 10, 20, 30, 10, 20, 30]  
b_tuple * 3 = (10, 20, 30, 10, 20, 30, 10, 20, 30)  
b_str * 3 = xyzxyzxyz  
a_list + b_list * 3 = [0, 1, 2, 10, 20, 30, 10, 20, 30, 10, 20, 30]
```

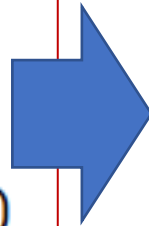
# データ型

データ型		記述例
int	整数	num = 6
float	浮動小数点	num = 3.14
str	文字列	str = 'abc'
bool	ブール	flg = True, flg = False
datetime	日付	date = datetime.datetime(2020, 1, 31, 12, 36, 45)
list	配列	list = ['abc', 6, True]
tuple	タプル	tuple = ('abc', 6, True)
dictionary	辞書	dict = {'Key1': 'Val1', 'Key2': 'Val2'}



# データ型の表示

```
a = 1234
b = 'abc'
c = 3.14159
d = [1, 2, 3, 4]
e = (1, 2, 3, 4)
g = {'key1': 1, 'key2': 2}
print(f'{a} is type {type(a)}')
print(f'{b} is type {type(b)}')
print(f'{c} is type {type(c)}')
print(f'{d} is type {type(d)}')
print(f'{e} is type {type(e)}')
print(f'{g} is type {type(g)}')
```



```
1234 is type <class 'int'>
abc is type <class 'str'>
3.14159 is type <class 'float'>
[1, 2, 3, 4] is type <class 'list'>
(1, 2, 3, 4) is type <class 'tuple'>
{'key1': 1, 'key2': 2} is type <class 'dict'>
```

# データ型の変換

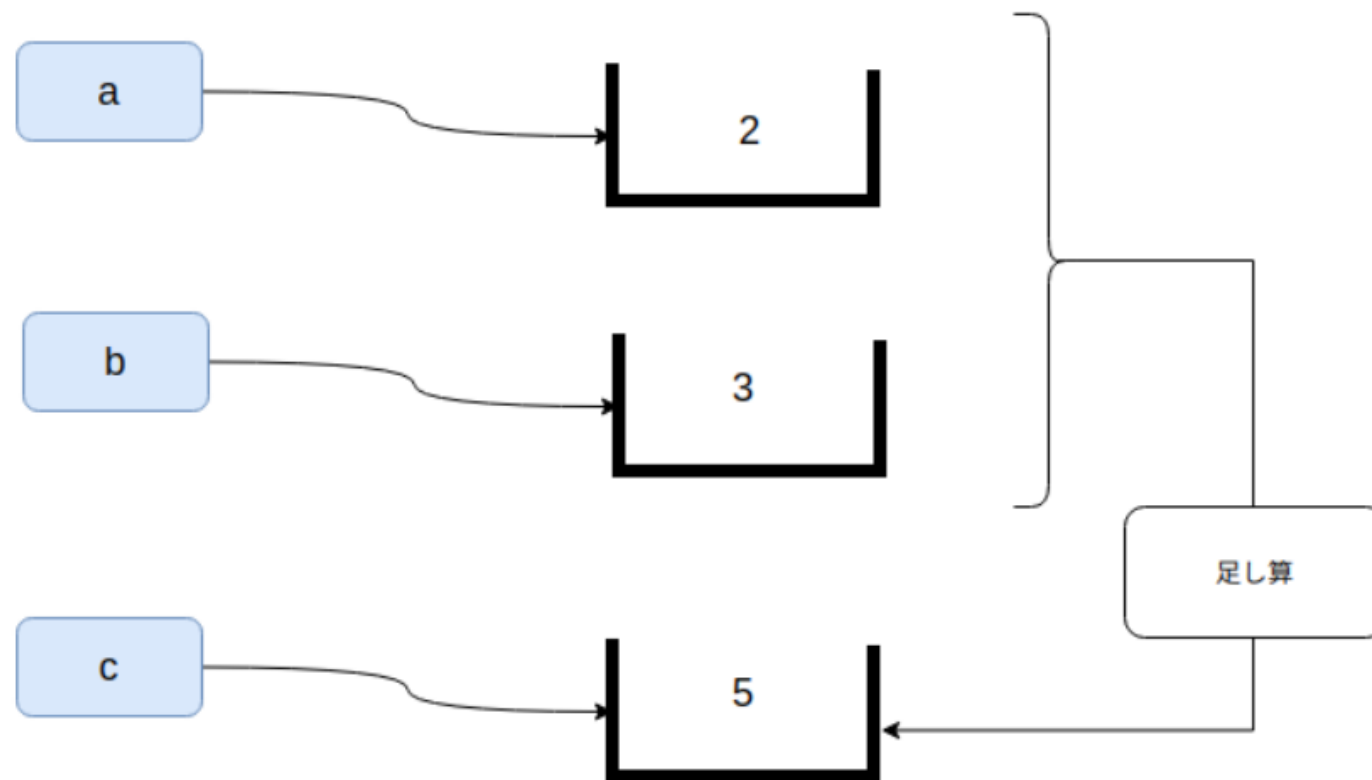
```
h = str(a)
print(f'{h} is type {type(h)}')
i = str(c)
print(f'{i} is type {type(i)}')
j = int('456')
print(f'{j} is type {type(j)}')
k = float('4.56')
print(f'{k} is type {type(k)}')
```



```
1234 is type <class 'str'>
3.14159 is type <class 'str'>
456 is type <class 'int'>
4.56 is type <class 'float'>
```

# 変数

```
1 | a = 2  
2 | b = 3  
3 | c = a + b
```



変数とは、プログラミング上でデータを保持するためのメモリ上の領域、つまり「データを格納するためのハコ」につけた名前、ラベルのようなものです)

# 変数

## Pythonの変数は変数宣言や型の宣言は不要

### 変数の命名規則

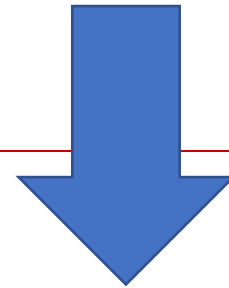
- アルファベットや数字などを使えます
- 記号は、アンダースコア ( \_ ) だけが使えます
- 数字から始まってはいけません。 Abc123という変数名はOKですが、123abcは数字で始まるのでエラーとなります
- If, while, forなどの、 Python言語の予約語は使えません
- ひらがなや漢字なども使えますが、エラーの原因になってしまうので、半角のアルファベットと数字、 アンダースコアだけを使うようにしましょう
- 大文字と小文字は区別される

# 変数

## 定義と代入

```
a = 1
b = 'ABC'
c = [1, 2, 3]
d = {'apple': 200, 'orange': 100, 'banana': 150}

print(a)
print(b)
print(c)
print(d)
```



```
1
ABC
[1, 2, 3]
{'apple': 200, 'orange': 100, 'banana': 150}
```

# 変数 - 再代入

```
a = 1  
print(a)  
a = 'aaa'  
print(a)  
b = a  
print(b)
```



```
1  
aaa  
aaa
```

# 変数 – 複数同時の初期化、代入

```
x, y, z = 1, 2, 3
a, b, c = x, y, z

print(f'x = {x}')
print(f'a = {a}')
print(f'y = {y}')
print(f'b = {b}')
print(f'z = {z}')
print(f'c = {c}')
```



```
x = 1
a = 1
y = 2
b = 2
z = 3
c = 3
```

# コメント、コメントアウトの書き方

- #によるコメント、コメントアウト
- トリプルクォート（三重引用符）による複数行コメント、コメントアウト



# コメント、コメントアウトの書き方 1

## インラインコメント

```
weight = 82 # your weight  
height = 1.76 # your height  
  
BMI = weight/height/height # your BMI  
print(f'Your BMI is {BMI}')
```

# コメント、コメントアウトの書き方 2

## ブロックコメント

```
# weight = 82 # your weight  
# height = 1.76 # your height  
  
# BMI = weight/height/height # your BMI  
# print(f'Your BMI is {BMI}') # print out
```

PyCharmでの行コメントのショートカット：ctrl + /

# コメント、コメントアウトの書き方 3

## トリプルクォートコメント

```
'''  
weight = 82 # your weight  
height = 1.76 # your height  
  
BMI = weight/height/height # your BMI  
print(f'Your BMI is {BMI}')
```

# キーボードから入力

```
val1 = input('Enter a string: ')
print(f'The string is {val1}')
print(type(val1))
```

```
val2 = input('Enter a number: ')
print(f'The number is {val2}')
print(type(val2))
```

```
print(val2 + 3)
```

Enter a string: *abc*

The string is abc

<class 'str'>

Enter a number: *123*

The number is 123

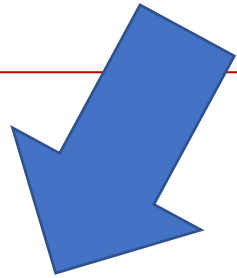
<class 'str'>

Traceback (most recent call last):

File "[C:\Users\steve\PycharmProjects\wangzk\input\\_function.py](C:\Users\steve\PycharmProjects\wangzk\input_function.py)", line 9, in <module>

```
print(val2 + 3)
```

TypeError: can only concatenate str (not "int") to str



# レポート課題

- 割り勘ツールをつくろう。金額と、人数、割引券をキーボードから入力し、1人あたり何円払うかと、その上で何円不足するかを表示しよう。例えば、13547円を4人で払い、全体で1000円分の割引券を持っていたとすると、1人3136円払って3円不足する。
- Pythonのプログラムを書いて、「report1\_name.py」で保存して、manaba+Rで提出してください。
- 他人にわかってもらうために、コメントを入れてください。