第一回
23/05/29

ITs
プログラミング教室
Python基礎編

約束 1

講義の内容を全て覚えようとしないで！

効率的なプログラミング学習方法

授業や動画でとりあえず「理解」する。

調べながら実装する。

調べる力をつけることができ、
実際に手を動かすことで記憶に定着する！

約束 2

演習している時、実装している時、
わからないところがあったら…

(1)最初の15分は自分で
解決を試みる。

(2) 15分たっても解決しな
かったら他人に聞く。

「⑴を守らないと他人の時間を無駄にし、
⑵を守らないと自分の時間を無駄にする。」

目次

目的

プログラミングの基礎

プログラミングでどんなことができるのか
を理解する！

プログラミングってなんだろう？

コンピュータは自分で考えられない

しかも人間の言葉が理解できない

プログラミング言語

→ プログラミングとは、
コンピュータに特定の手順や命令を与えること

プログラミングってなんだろう？(どんなものが作れる？)

ソフトウェア開発

- スマホアプリ
- PCアプリ
- ~~ポモドーロタイマー~~

ウェブ開発

- 動画配信
- SNS
- ブログ・掲示板


AI データ分析

- チャット
- 画像認識
- ゲーム


ロボット制御

- 人型ロボット
- 自動運転

ゲーム開発

- スマホゲーム
- PCゲーム
- SwitchとかPS5も


他にも…

- 自動化システム (lineで日直を通知)
- OS開発
- CLIアプリ

プログラミング言語は一つじゃない！

用途・規模によって使い分ける必要がある

今回学ぶのは「Python」という言語

【入門】Pythonとは？特徴やできることを初心者向けに解説 | テックキャンプ ブログ (tech-camp.in)

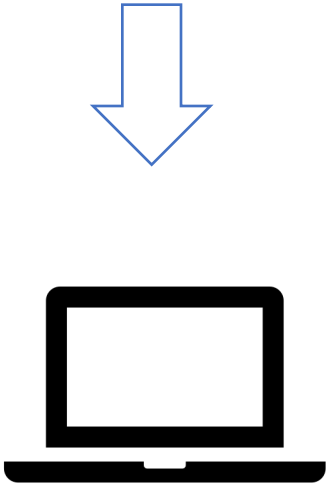
環境構築

Pythonプログラム

↑そのままで
は動かない

翻訳者(？)

コンピュータ
が理解できる
実行形式

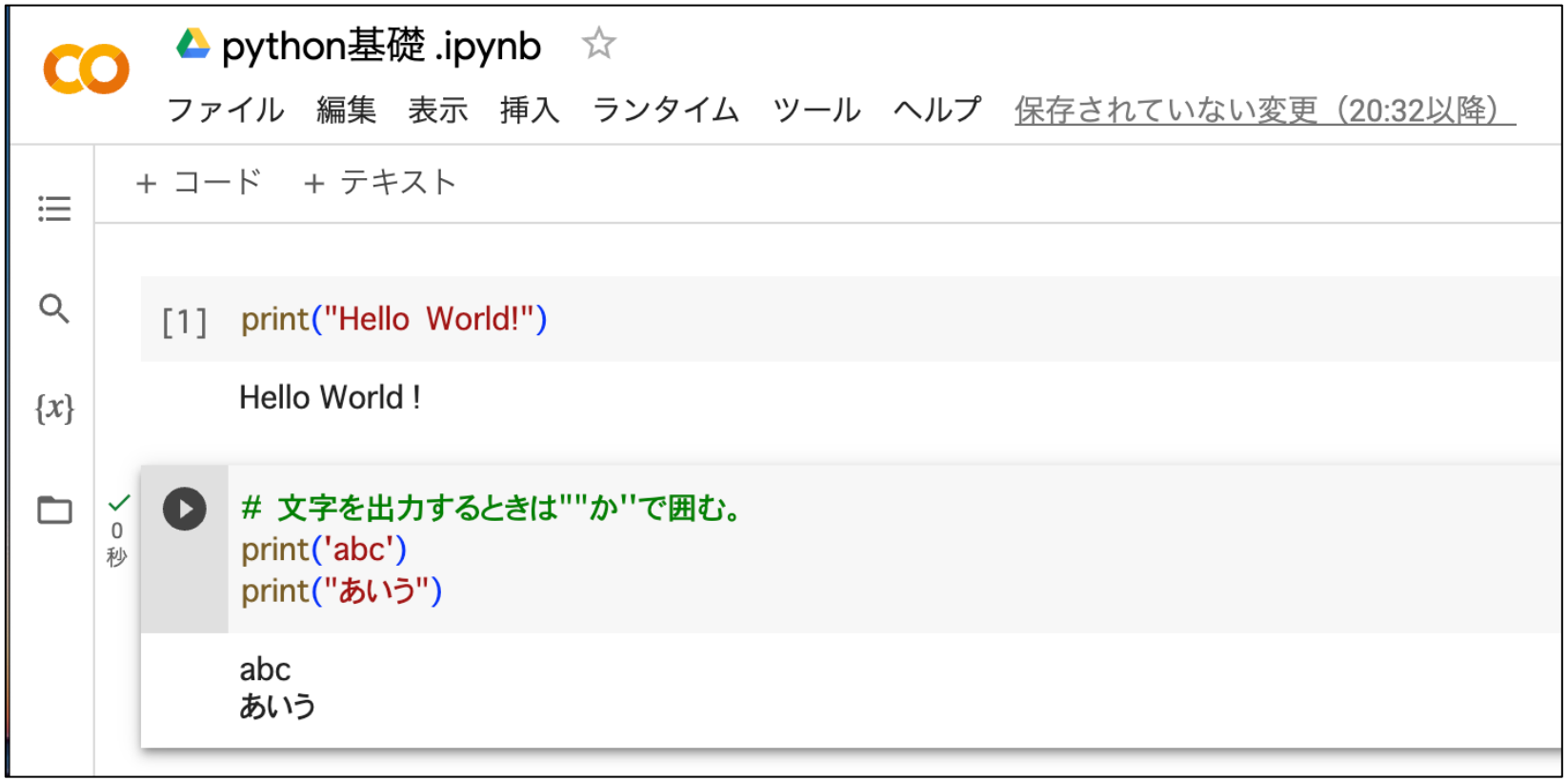


Install !

めんどくさいので
Googleさんの力を借ります

画面共有でやります
参加できなかった人は
録画を見て下さい！

文字を表示させる



演習1.1）
自分の名前を表示させて下さい。

出力例:
阿部 凌央

2分

四則演算


```
# 整数の四則演算
print(12 * 5)
print(144 / 6)
print(144 // 7)
print(134 % 5) # あまり
print(3 ** 4) # べき乗
```

```
60
24.0
20
4
81
```

```
# 小数の四則演算
print(0.1 + 0.1)
print(2.0 - 0.5)
print(0.5 * 2.0)
print(3.0 * 0.1)
print(3.0 / 2.0)
```

```
0.2
1.5
1.0
0.30000000000000004
1.5
```

✓ 0秒

```
# 3つ以上でも計算できる
print(1 + 2 * 4)
print(3 * (4 + 5))
```

```
9
27
```

演習1.2）

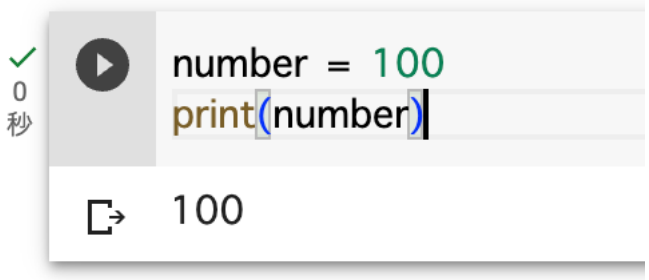
1. 12345679×9の値を求めよ。

2. 100＊6+5 と 100＊(6+5)の違いを確かめよ。

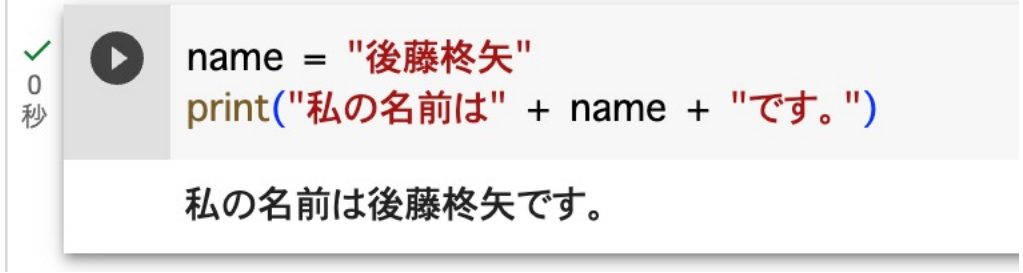
3. 123の3乗を8で割った余りを求めよ。

5分

変数



numberという名前に100という値を紐づける
→ numberに100を**代入する**という

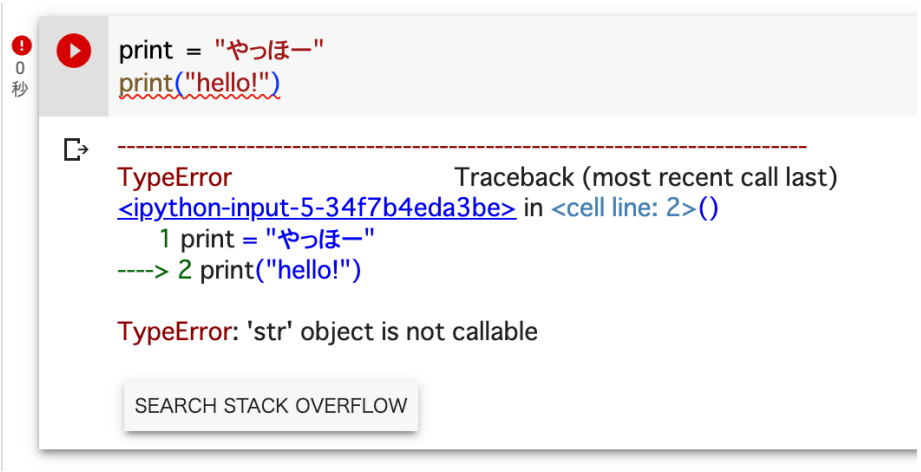


文字列でも使える

計算もできる

変数

- ここでの”=”は、左右が等しいという意味を持たない
- Pythonでは、あらゆるものが変数に代入できる

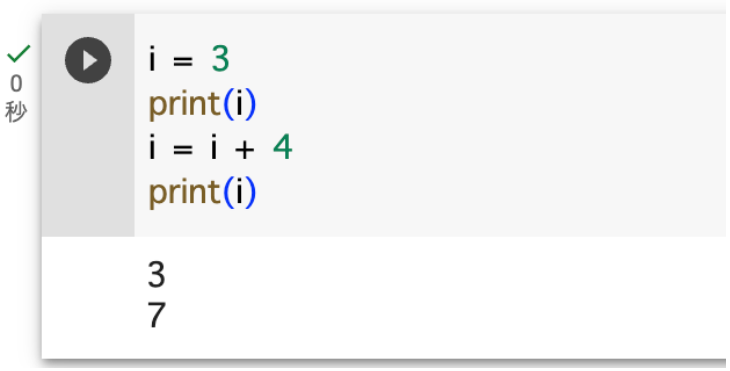


変数名として使用しない
方が良いものがある

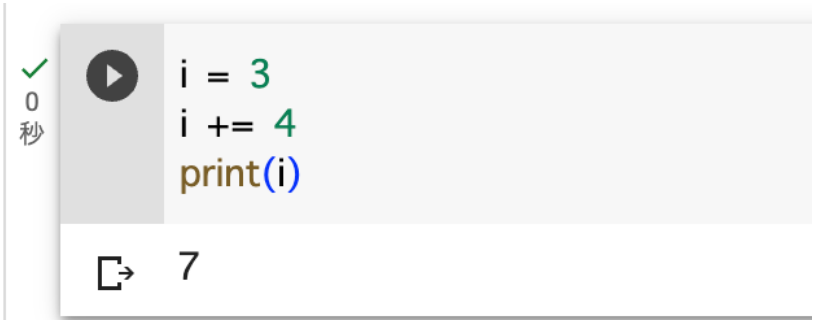
→ 予約語と呼ばれる

予期しないエラーが発生する可能性

例: for, class, def, if　などなど



← 右辺が先に計算され、
左に代入される



← i = i + 4と同じ意味

演習1.3)
1. 変数nameに自分の名前を代入し、
自己紹介をして下さい

出力例:
私の名前は阿部凌央です。

---

2. iに4を代入し、iに5をかけたものを
iに代入して下さい。

出力例:
20


5分

文字を表示させるPart 2

✓ 0秒

```
name = "阿部 凌央"
age = 19
print("私の名前は{0}です。{1}さいです".format(name, age))
```

```
私の名前は阿部 凌央です。19さいです
```

✓ 0秒

```
name = "阿部 凌央"
age = 19
print(f"私の名前は{name}です。{age}さいです")
```

```
私の名前は阿部 凌央です。19さいです
```

文字列に変数を埋め込むことができる。

文字を表示させるPart 2

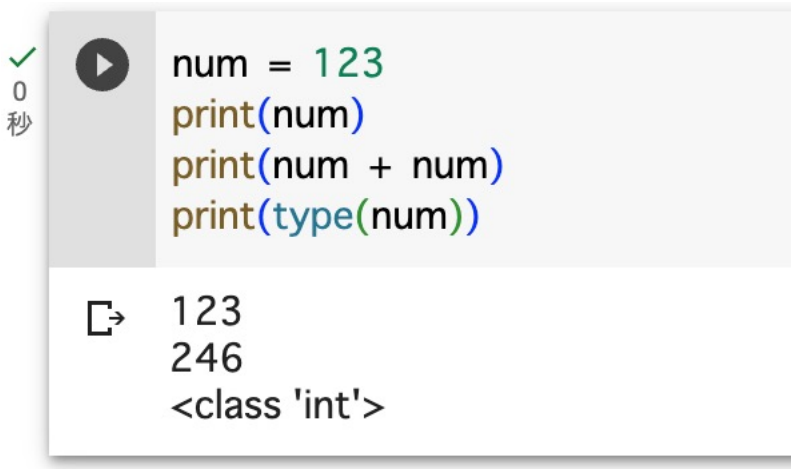
演習1.4)
変数nameに自分の名前を代入し、
自己紹介をして下さい

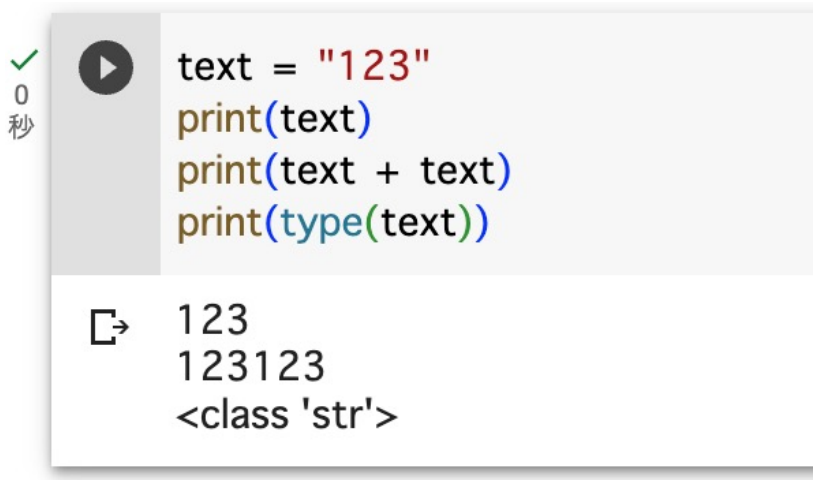
出力例:
私の名前は阿部凌央です。

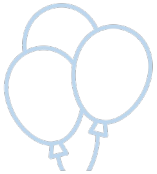
2分

データ型

プログラミングにおいて、123というデータを扱いたいとき、数値としての123なのか、文字列としての”123”なのかによって処理が異なる。







type()でデータ型が確認できる。

データ型

Pythonに組み込まれているデータ型の種類

型	名称	例
int	整数	123, -5, 11111
str	文字列	“hello”, “123”
float	浮動小数	1.2, -3.14, 5.0
bool	真偽地	true, false
list	配列	[1,2,3], [“a”, “b”]
tuple	タプル	(3,4)

Pythonには、存在するすべてのものに型がある。
自分で型を作成することもできる。

演習1.5)
変数piに3.14を代入し、type()を用いて変数pi_typeに変数piの型を代入して下さい。
そのあとf文字列またはformat()を使用して、次のように出力して下さい。

出力:
“変数piはxxx型です。”


3分

リスト


```
# 複数の要素を一つにまとめることができる
languages = ["Chinese", "English", "Indonesian", "Japanese"]
print(languages)
print(type(languages))
```

```
['Chinese', 'English', 'Indonesian', 'Japanese']
<class 'list'>
```

```
# 格納されている要素には番号が付けられている
# 番号を指定して取り出すことができる
# 0から始まることに注意(そのうち便利さに気づくから慣れろ)
print(languages[0])
print(languages[2])
print(languages[-1])
```

```
Chinese
Indonesian
Japanese
```

リストの要素を指定する
番号をインデックスという
例:
languages[0]→languagesリストの
インデックス0

演習1.6)
num_listの最初から5番目の要素と後ろから
４番目の要素を出力してください。

```
num_list = [1,4,9,5,189,0,1,0,9,0,2,4,5,2,4,2,
3,4,2,9,8,9,7,9,0,0,8,0,1,1,4,9,5,9,0,1,0,9,0,
0,2,4,5,2,4,2,2,4,2,1,456]
```

出力:
189
4

3分

num_listはteamsのチャットに送ります

リスト: 要素の追加と書き換え


```
languages = ["Chinese", "English", "Indonesian", "Japanese"]
languages[0] = "German"
languages[-1] = "French"
print(languages)
```

```
['German', 'English', 'Indonesian', 'French']
```

インデックスを指定して
書き換え

```
[2] languages = ["Chinese", "English", "Indonesian", "Japanese"]
    languages.append("German")
    languages.append("French")
    print(languages)
```

```
['Chinese', 'English', 'Indonesian', 'Japanese', 'German', 'French']
```

後ろに追加

```
languages = ["Chinese", "English", "Indonesian", "Japanese"]

languages.insert(2, "French")
print(languages)
```

```
['Chinese', 'English', 'French', 'Indonesian', 'Japanese']
```

インデックスを指定して
追加

リスト: 要素の削除


```
languages = ["Chinese", "English", "Indonesian", "Japanese"]

popped_language = languages.pop()
print(popped_language)
print(languages)
```

```
Japanese
['Chinese', 'English', 'Indonesian']
```

後ろから削除
削除した要素を取り出す

```
languages = ["Chinese", "English", "Indonesian", "Japanese"]

languages.pop(1)
print(languages)
```

```
['Chinese', 'Indonesian', 'Japanese']
```

インデックスを指定して削除
削除した要素を取り出す

```
languages = ["Chinese", "English", "Indonesian", "Japanese"]
languages.remove("English")
print(languages)
```

```
['Chinese', 'Indonesian', 'Japanese']
```

要素を指定して削除

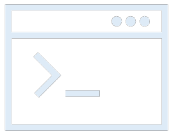
リスト: 要素の追加・削除

演習1.7)

1. “Abe”, “Sasaki”, “Katagishi”を要素とするリストをit_esという変数に代入し、そのリストを出力して下さい。
2. ”Goto”をリストの最後尾に追加して下さい。
3. “Abe”を指定して削除して下さい。
4. 3の状態のリストに対して、2番目の位置に“Fujiwara”を挿入して下さい。
5. 4の状態のリストの3番目を削除し、削除した要素を出力して下さい。

出力:

1. [“Abe”, “Sasaki”, “Katagishi”]
2. [“Abe”, “Sasaki”, “Katagishi”, “Goto”]
3. [“Sasaki”, “Katagishi”, “Goto”]
4. [“Sasaki”, “Fujiwara” “Katagishi”, “Goto”]
5. “Katagishi”



5分