

演習授業中の質問対応について

Zoom ミーティング

表示

ミーティング チャット

演習授業中の質問をチューターの先生方が対応させていただきます。

演習にエラーが出たなど問題があったらリアクションの**挙手**を押してください。

質問内容を入力して、「**全員**」宛てに送信してください。

Miho Ishimaru

ここにメッセージを入力します...

宛先: 全員

ミュート解除

ビデオの開始

セキュリティ

参加者 2

画面共有

リアクション

アプリ

ホワイトボード

ノート

詳細

終了

演習3

Python基礎 プログラミング基礎

本スライドは、自由にお使いください。
使用した場合は、このQRコードからアンケート
に回答をお願いします。



統合教育機構 曹 日丹

医療とAI・ビッグデータ入門

演習2-7の構成

Python基礎を学びましょう

演習2 11/16 11:35-12:20

Pythonの変数とデータの型

Pythonを使ってみましょう

演習5

患者の歯に関する病院のリアルワールドデータの説明

演習3 11/30 11:35-12:20

プログラミング基礎

演習6

データクレンジングに必要なライブラリ（Pandas）の応用

演習4

モジュール、パッケージ、ライブラリ

演習7

データクレンジングとデータの可視化

プログラミング（コーディングとも呼ばれます）は、コンピュータに特定のタスクを実行させるための命令を書くプロセスです。プログラミング言語を使用して、コンピュータに対して**明確で順序だった命令のセット**を提供します。

コンピュータは人間とは異なり、曖昧な命令や指示を理解することができません。

コンピュータは命令を上から下へと一つずつ順番に実行します。

プログラミング（コーディングとも呼ばれます）は、コンピュータに特定のタスクを実行させるための命令を書くプロセスです。プログラミング言語を使用して、コンピュータに対して**明確で順序だった命令のセット**を提供します。

if と for Pythonのプログラムで非常に頻繁に使用される構造です。

if文（条件分岐）：

特定の条件が真(True)か偽(False)かに基づいて**プログラムの実行経路**（コードの実行フローを制御する）を変更します。

for文（繰り返し）：

データの集まりの中、要素ごとに一連の操作を**繰り返し実行**します。繰り返す回数は、要素の数で決まります。

if と for

Pythonのプログラムで非常に頻繁に使用される構造です。

if文（条件分岐）：

if文は条件分岐を実現するためのもので、特定の条件が真(True)か偽(False)かに基づいてプログラムの実行経路を変更します。

日本語での**if文**に相当する部分は「**もし～なら**」や「もし条件が成り立つ場合は」などのように言えます。

for文（繰り返し）：

データの集まりの中、要素ごとに一連の操作を繰り返し実行します。繰り返す回数は、要素の数で決まります。

日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

if

日本語での**if文**に相当する部分は「**もし～なら**」や「もし条件が成り立つ場合は」などのように言えます。

if **条件**:

条件が**真**の場合に実行されるコード

else:

条件が**偽**の場合に実行されるコード

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

if **条件**:

条件が**真**の場合に実行されるコード

else:

条件が**偽**の場合に実行されるコード

真(True)か**偽**(False)

条件が真(True)か偽(False)かを判定するためには、**比較演算子**や**論理演算子**を使用します。

Colab

検索google colab Colaboratory へようこそ - Colaboratory - Google

 Colaboratory へようこそ
ファイル 編集 表示

目次

はじめに

データサイエンス

機械学習

その他のリソース

使用例

セクション

ノートブックを開く

例

最近

Google ドライブ

GitHub

アップロード

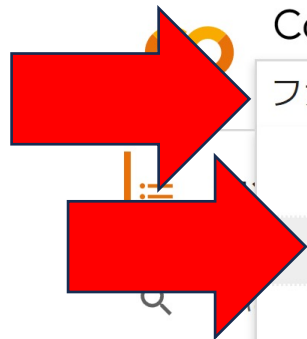
ノートブックを検索

タイトル	最終閲覧	最初に開いた日時	
演習2コード	9:43	10月25日	
Colaboratory へようこそ	9:21	2022年12月23日	
演習3コード	9:21	9:21	
演習4コード	11月2日	11月1日	
演習5コード	11月1日	10月13日	

+ ノートブックを新規作成

キャンセル

検索google colab Colaboratory へようこそ - Colaboratory - Google



Colaboratory へようこそ

ファイル 編集 表示 挿入 ランタイム ツール ヘルプ

ノートブックを新規作成

ノートブックを開く

Ctrl+O

ノートブックをアップロード

名前の変更

ドライブにコピーを保存

コピーを GitHub Gist として保存

GitHub にコピーを保存

保存

Ctrl+S

変更履歴

ダウンロード

印刷

Ctrl+P

ド + テキスト

ドライブにコピー

Colab へようこそ

に Colab をよくご存じの場合は、この動画でインタラクティブなラドの履歴表示、コマンドパレットについてご覧ください。



Colab とは

検索google colab Colaboratory へようこそ - Colaboratory - Google

ノートブックを開く

例 >

最近 >

Google ドライブ >

GitHub >

アップロード >

タイトル	所有者	最終閲覧 ▲	最終更新 ▼		
演習3コード.ipynb					
演習準備資料.ipynb	曹日丹	11月1日	11月1日		
演習1116確認.ipynb のコピー	曹日丹	10月31日	10月27日		
2023入門dataframe.ipynb	曹日丹	10月31日	10月27日		
演習1116確認.ipynb	曹日丹	10月25日	10月25日		

+ ノートブックを新規作成

キャンセル

検索google colab Colaboratory へようこそ - Colaboratory - Google

ノートブックを開く

例 >

最近 >

Google ドラ
イブ >

GitHub >

アップロード >



参照

または、ここにファイルをドラッグしてください

演習3コード.ipynb

演習授業中の質問対応について

Zoom ミーティング

表示

ミーティング チャット

演習授業中の質問をチューターの先生方が対応させていただきます。

演習にエラーが出たなど問題があったらリアクションの**挙手**を押してください。

質問内容を入力して、「**全員**」宛てに送信してください。

Miho Ishimaru

メッセージは誰に表示されますか？
宛先: **全員** ▼

ここにメッセージを入力します...

ミュート解除

ビデオの開始

セキュリティ

参加者 2

画面共有

リアクション

アプリ

ホワイトボード

ノート

詳細

終了

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 〇〇
```

```
if age < 18:
    print("未成年です。")
```

条件を定義しました。
条件が**真**の場合に実行されるコード

```
else:
    print("成人です。")
```

条件が**偽**の場合に実行されるコード

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 18
```

```
if age < 18:
```

```
    print("未成年です。")
```

18歳は小なり18歳ではないため、条件が**偽**と判定しました。

```
else:
```

```
    print("成人です。")
```

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 18
```

```
if age < 18:  
    print("未成年です。")
```

```
else:  
    print("成人です。")
```

18歳は大なりイコール18歳であるため、実行されます

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 18
```

```
if age < 18:  
    print("未成年です。")
```

```
else:  
    print("成人です。")
```

18歳は大なりイコール18歳であるため、

実行されます



成人です。

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 18
```

```
if age < 18:
```

```
    print("未成年です。")
```

```
else:
```

```
    print("成人です。")
```

Pythonのインデント（字下げ）：

- ・プログラムの構造を示すために使用スペースです
- ・インデントの位置によって、コードの開始と終了が判別されます。
- ・インデントは通常4つのスペースを使用することが推奨されています。

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

デモ：ifを使って簡単なプログラムを書きましょう

```
age = 18
if age < 18:
    print("未成年です。")
else:
    print("成人です。")
```

比較演算子: 値や変数の比較を行います。

== : 等しい

!= : 等しくない

< : より小さい

> : より大きい

<= : 以下

>= : 以上

論理演算子: 2つ以上の条件を組み合わせて比較を行います。

and : かつ

or : または

not : でなければ

Colab

if

日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

if **条件1:**

#条件1がTrueの場合の処理

elif **条件2:**

#条件2がTrueの場合の処理

elif **条件3:**

#条件3がTrueの場合の処理

else: **それ以外の場合**

#すべての条件がFalseの場合の処理

条件は2つ以上ある場合は、**elif**を使います。
elifは「**else if**」の短縮形です。日本語では「**それとももし**」や「**そうではなくてもし**」と言い換えることが一般的です。

条件1ではなくて、
もし条件2に当てはまる**なら**

条件1でも**条件2**でも**なくて**、
もし条件3に当てはまる**なら**

if 日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

```
if 条件1:
    #条件2がTrueの場合の処理
elif 条件2:
    #条件2がTrueの場合の処理
elif 条件3:
    #条件3がTrueの場合の処理
elif 条件4:
    #条件4がTrueの場合の処理
else:
    #すべての条件がFalseの場合の処理
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

if 日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

```
if score < 60
    #条件1がTrueの場合の処理
elif 60 <= score < 70
    #条件2がTrueの場合の処理
elif 70 <= score < 80
    #条件3がTrueの場合の処理
elif 80 <= score < 90
    #条件4がTrueの場合の処理
else:
    #すべての条件がFalseの場合の処理
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

score >= 90

if 日本語での**if文**に相当する部分は「もし～なら」や「もし条件が成り立つ**場合は**」などのように言えます。

```
if score < 60
    #条件1がTrueの場合 "D"
elif 60 <= score < 70
    #条件2がTrueの場合 "C"
elif 70 <= score < 80
    #条件3がTrueの場合 "B"
elif 80 <= score < 90
    #条件4がTrueの場合 "A"
else:
    #すべての条件がFalseの場合 "A+"
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

if 日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

```
if score < 60
    print("D")
elif 60 <= score < 70
    print("C")
elif 70 <= score < 80
    print("B")
elif 80 <= score < 90
    print("A")
else:
    print("A+")
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

Colab

if 日本語での**if文**に相当する部分は「**もし～なら**」や「**もし**条件が成り立つ**場合は**」などのように言えます。

```
score = 95
```

```
if score < 60
```

```
    print("D")
```

```
elif 60 <= score < 70
```

```
    print("C")
```

```
elif 70 <= score < 80
```

```
    print("B")
```

```
elif 80 <= score < 90
```

```
    print("A")
```

```
else:
```

```
    print("A+")
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

Colab

if 日本語での**if文**に相当する部分は「もし～なら」や「もし条件が成り立つ**場合は**」などのように言えます。

```
score = 95
```

```
if score < 60 ×
    print("D")
elif 60 <= score < 70 ×
    print("C")
elif 70 <= score < 80 ×
    print("B")
elif 80 <= score < 90 ×
    print("A")
else:
    print("A+") 実行されます
```

成績の値に基づいて評価します

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

Colab

if と for

Pythonのプログラムで非常に頻繁に使用される構造です。

if文（条件分岐）：

if文は条件分岐を実現するためのもので、特定の条件が真(True)か偽(False)かに基づいてプログラムの実行経路（コードの実行フローを制御する）を変更します。

日本語での**if文**に相当する部分は「**もし～なら**」や「もし条件が成り立つ場合は」などのように言えます。

for文（繰り返し）：

データの集まりの中、要素ごとに一連の操作を繰り返し実行します。繰り返す回数は、要素の数で決まります。

日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

```
for 変数 in 要素の集まり:
```

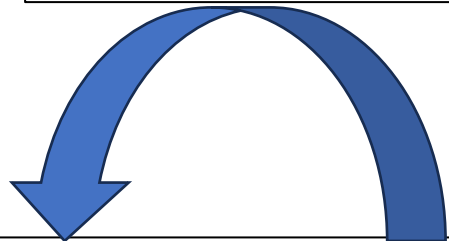
```
    繰り返し実行するコード
```

}

forブロック

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

繰り返すたびにデータの集まりの要素を順番に変数に代入します。



```
for 変数 in 要素の集まり:  
    繰り返し実行するコード
```

要素の集まり：

リスト fruits = ["apple", "banana", "cherry"]

タプル fruits = ("apple", "banana", "cherry")

辞書 fruits = {"apple": "red", "banana": "yellow", "cherry": "red"}

セット fruits = {"apple", "banana", "cherry"}

整数列：

range(n): 0以上n未満までの範囲の整数列を作成されます。

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

デモ：forを使ってプログラムを書きましょう

1. リストの各要素を表示する

```
fruits = ["apple", "blueberry", "melon", "strawberry"]
```

リストの要素を一つずつ取り出します

```
print(fruits[0])
```

```
print(fruits[1])
```

```
print(fruits[2])
```

```
print(fruits[3])
```

apple

blueberry

melon

strawberry

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

デモ：forを使ってプログラムを書きましょう

1. リストの各要素を表示する

```
fruits = ["apple", "blueberry", "melon", "strawberry"]  
for f in fruits:  
    print(f)
```

リストの要素を一つずつ取り出します

f = "apple"

print(f)

apple

f = "blueberry"

print(f)

blueberry

f = "melon"

print(f)

melon

f = "strawberry"

print(f)

strawberry

リスト `fruits` 内の各要素が変数 `f` に順番に代入され、`print()` 関数によって画面に表示されます。結果として、リスト内の要素が順番に表示されることになりました。

Colab

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

デモ：forを使ってプログラムを書きましょう

2. 0から3までの整数を順番に表示します

整数の集まり：

range(n): 0以上n未満までの範囲の整数列を作成されます。

```
for i in range(n):  
    print(i)
```

range(n): a以上b未満までの範囲の整数列を作成されます。

```
for i in range(a, b):  
    print(i)
```


for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

デモ：forを使ってプログラムを書きましょう

2. 0から3までの整数を順番に表示します

整数の集まり：

range(n): 0以上n未満までの範囲の整数列を作成されます。

```
for i in range(4):  
    print(i)
```

range(n): a以上b未満までの範囲の整数列を作成されます。

```
for i in range(0, 4):  
    print(i)
```

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

デモ：forを使ってプログラムを書きましょう

2. 0から3までの整数を順番に表示します

整数の集まり：

range(n): 0以上n未満までの範囲の整数列を作成されます。

```
for i in range(4):  
    print(i)
```

```
for i in range(0, 4):  
    print(i)
```

```
i = 0  
    print(i)  
i = 1  
    print(i)  
i = 2  
    print(i)  
i = 3  
    print(i)
```

0
1
2
3

0
1
2
3

range(n): a以上b未満までの範囲の整数列を作成されます。

Colab

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
for s in scores:
```

```
    print("D")
```

```
    print("C")
```

```
    print("B")
```

```
    print("A")
```

```
else:
```

```
    print("A+")
```

成績	評価
90以上	A+
80以上	A
70以上	B
60以上	C
60未満	D

条件1

条件2

条件3

条件4

条件5

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
for s in scores:
    if s < 60:
        print("D")
    elif 60 <= s < 70 :
        print("C")
    elif 70 <= s < 80:
        print("B")
    elif 80 <= s < 90 :
        print("A")
    else:
        print("A+")
```

成績	評価	
90以上	A+	条件1
80以上	A	条件2
70以上	B	条件3
60以上	C	条件4
60未満	D	条件5

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
s = 99
```

```
if s < 60: X
```

```
    print("D")
```

```
elif 60 <= s < 70: X
```

```
    print("C")
```

```
elif 70 <= s < 80: X
```

```
    print("B")
```

```
elif 80 <= s < 90: X
```

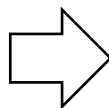
```
    print("A")
```

```
else:
```

```
    print("A+")
```

○

実行されます



A+

for 日本語での**for文**に相当する部分は「**～のために繰り返す**」や「**～ごとに繰り返す**」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
s = 75
```

```
if s < 60: X
```

```
    print("D")
```

```
elif 60 <= s < 70: X
```

```
    print("C")
```

```
elif 70 <= s < 80: O
```

```
    print("B")
```

```
elif 80 <= s < 90:
```

```
    print("A")
```

```
else:
```

```
    print("A+")
```

実行されます

以下、実行されません

A+

B

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

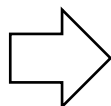
```
scores = [99, 75, 59, 85, 60]
```

```
s = 59
```

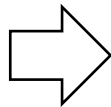
```
if s < 60:
    print("D")
elif 60 <= s < 70:
    print("C")
elif 70 <= s < 80:
    print("B")
elif 80 <= s < 90:
    print("A")
else:
    print("A+")
```

○
実行されます

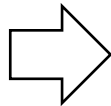
以下、実行されません



A+



B



D

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
s = 85
```

```
if s < 60: X
```

```
    print("D")
```

```
elif 60 <= s < 70: X
```

```
    print("C")
```

```
elif 70 <= s < 80: X
```

```
    print("B")
```

```
elif 80 <= s < 90: O
```

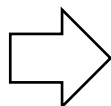
```
    print("A")
```

```
else:
```

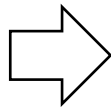
```
    print("A+")
```

実行されます

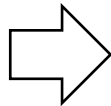
以下、実行されません



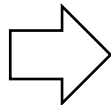
A+



B



D



A

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
s = 60
```

```
if s < 60: X
```

```
    print("D")
```

```
elif 60 <= s < 70: O
```

```
    print("C")
```

```
elif 70 <= s < 80:
```

```
    print("B")
```

```
elif 80 <= s < 90:
```

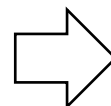
```
    print("A")
```

```
else:
```

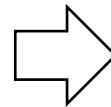
```
    print("A+")
```

実行されます

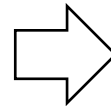
以下、実行されません



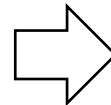
A+



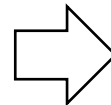
B



D



A



C

for 日本語での**for文**に相当する部分は「**～のために**繰り返す」や「**～ごとに**繰り返す」などのように言えます。

演習：コード02を書いてみましょう

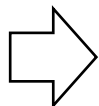
5人の成績の値に基づいて評価します

```
scores = [99, 75, 59, 85, 60]
```

```
for s in scores:
```

```
    if s < 60:
        print("D")
    elif 60 <= s < 70 :
        print("C")
    elif 70 <= s < 80:
        print("B")
    elif 80 <= s < 90 :
        print("A")
    else:
        print("A+")
```

繰り返す回数は、リストの要素の数で決まります。
要素を全て代入した後、繰り返しが終わります。



A+
B
D
A
C

Colab

関数

特定の処理を実行するためのコードです。

関数 特定の処理を実行するためのブロックをまとめたものです。

タスク：四則演算

$x = a + 4*b + 5*c$

関数 特定の処理を実行するためのブロックをまとめたものです。

タスク：四則演算

$$x = a + 4*b + 5*c$$

$a = 1$ $b = 2$ $c = 3$	→	$x = 1 + 4*2 + 5*3$	→	24
-------------------------------	---	---------------------	---	----

$a = 4$ $b = 5$ $c = 6$	→	$x = 4 + 4*5 + 5*6$	→	54
-------------------------------	---	---------------------	---	----

...	→		→	...
-----	---	--	---	-----

関数 特定の処理を実行するためのブロックをまとめたものです。

処理を関数のコードに変換しました。

関数

```
def q(a, b, c):
```

```
    x = a + 4*b + 5*c
```

```
    return x
```

関数 特定の処理を実行するためのブロックをまとめたものです。

処理を関数のコードに変換しました。

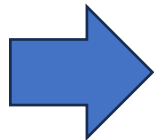
関数

```
def q(a, b, c):
```

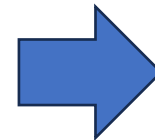
```
    x = a + 4*b + 5*c
```

```
    return x
```

```
a = 156  
b = 243  
c = 399
```



```
x = q(156, 243, 399)
```



関数 特定の処理を実行するためのブロックをまとめたものです。

処理を関数のコードに変換しました。

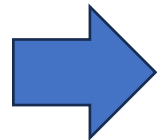
関数

```
def q(a, b, c):
```

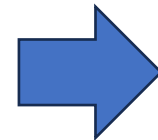
```
    x = a + 4*b + 5*c
```

```
    return x
```

```
a = 156  
b = 243  
c = 399
```



```
x = q(156, 243, 399)
```



```
3123
```


関数 特定の処理を実行するためのブロックをまとめたものです。

関数の中身を見てみます。

```
def q(a, b, c):
```

キーワード

関数名

引数：関数にデータを渡すための変数

```
    x = a + 4*b + 5*c
```

```
    return x
```

関数 特定の処理を実行するためのブロックをまとめたものです。

関数の中身を見てみます。

```
def q(a, b, c):
```

キーワード

関数名

引数：関数にデータを渡すための変数

```
    x = a + 4*b + 5*c
```

関数の本体：関数の処理を記述します

```
    return x
```

関数 特定の処理を実行するためのブロックをまとめたものです。

コードの中身を見てみます。

```
def q(a, b, c):
```

キーワード

関数名

引数：関数にデータを渡すための変数

```
    x = a + 4*b + 5*c
```

関数の本体：関数の処理を記述します

```
    return x
```

戻り値：関数が値を返す場合、return
文を使用してその値を指定します


関数 特定の処理を実行するためのブロックをまとめたものです。

```
def 関数名(引数1, 引数2, ...):
```

```
    処理1
```

```
    処理2
```

```
    return 戻り値
```



インデント：
プログラムの構造を示すために使用される空白文字
処理1、処理2およびreturnは、関数の一部内容として
認識されています。

関数 特定の処理を実行するためのブロックをまとめたものです。

```
def 関数名(引数1, 引数2, ...):
```

処理1

return 戻り値

```
def q(a, b, c):
```

$x = a + 4*b + 5*c$

return x

関数qを作成しました。

関数qを呼び出します。

```
x = q(1, 2, 3)  
print(x)
```

関数 特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

BMI（Body Mass Index、ボディマス指数）は、体重と身長を基にして身体の肥満度を評価するための指数です。体重（kg）を身長（m）の2乗（身長×身長）で割った数値がBMI指数となります。

BMIは以下の数式で計算されます：

$$\text{BMI} = \text{体重 (kg)} / (\text{身長 (m)} * \text{身長 (m)})$$

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
bmi = weight / (height** 2)
```

処理方法：計算方法

引数a

weight = ____m

引数b

height = ____Kg

関数

******：プログラミング言語では累乗（べき乗）の演算子です。
height 2**：身長の高さの2乗の意味です。

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

`bmi = weight / (height** 2)` 処理方法：計算方法

引数a

weight = ____m

引数b

height = ____Kg

```
def calculate_bmi( 引数a 引数b ):
```

処理方法：計算方法

return

関数

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

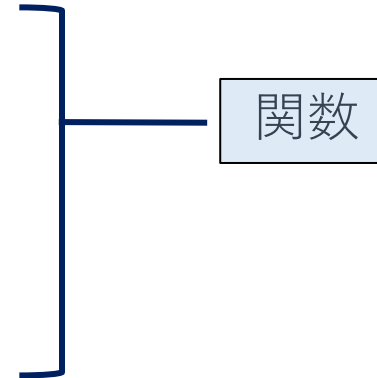
演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
def calculate_bmi( weight, height ):
```

```
    
```

```
    return 
```



関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

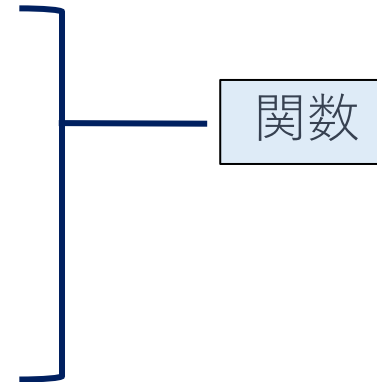
演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
def calculate_bmi( weight, height ):

    bmi = weight / (height** 2)

    return bmi
```



関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
def calculate_bmi( weight, height ):
    bmi = weight / (height** 2)
    return bmi
```

関数calculate_bmiを作成しました。

関数calculate_bmiを呼び出します。

weight = 70
height = 1.75

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
def calculate_bmi( weight, height ):  
    bmi = weight / (height** 2)  
    return bmi
```

```
mybmi = calculate_bmi(70, 1.75)  
print(mybmi)
```

関数calculate_bmiを作成しました。

関数calculate_bmiを呼び出します。

weight = 70
height = 1.75

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを計算する関数を書きましょう

```
def calculate_bmi( weight, height ):  
    bmi = weight / (height** 2)  
    return bmi
```

```
mybmi = calculate_bmi(70, 1.75)  
print(mybmi)
```

22.86

関数**calculate_bmi**を作成しました。

関数**calculate_bmi**を呼び出します。

weight = 70
height = 1.75

Colab

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード04を書いてみましょう

関数

BMIの値に基づいて肥満度を評価します

BMIの値を以下のように解釈することが一般的です：

BMI < 18.5 : 低体重 (Underweight)

$18.5 \leq \text{BMI} < 24.9$: 正常体重 (Normal weight)

$25 \leq \text{BMI} < 29.9$: 軽度の肥満 (Overweight)

BMI ≥ 30 : 肥満 (Obesity)

引数：bmi

処理方法：if文

関数

関数 関数は、特定のタスクや処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを評価します

```
def bmi_category(bmi):
```

引数：bmi

処理方法：if、elif、else文

return BMIの評価

関数

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを評価します

```
def bmi_category(bmi):
```

引数：bmi

```
    if bmi < 18.5:
```

処理方法：if条件式

関数

```
        elif 18.5 <= bmi < 25:
```

```
        elif 25 <= bmi < 30:
```

```
    else:
```


関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード03を書いてみましょう

BMIを評価します

```
def bmi_category(bmi):
```

```
    if bmi < 18.5:  
        return "低体重"
```

if文

return

```
    elif 18.5 <= bmi < 25:  
        return "正常体重"
```

if文

return

```
    elif 25 <= bmi < 30:  
        return "軽度の肥満"
```

if文

return

```
    else:  
        return "肥満"
```

if文

return

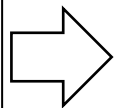
関数

関数 関数は、特定の処理を実行するためのブロックをまとめたものです。

演習：コード04を書いてみましょう

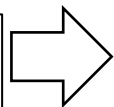
BMIの値に基づいて肥満度を評価します

```
def bmi_category(bmi):  
    if bmi < 18.5:  
        return "低体重"  
    elif 18.5 <= bmi < 25:  
        return "正常体重"  
    elif 25 <= bmi < 30:  
        return "軽度の肥満"  
    else:  
        return "肥満"
```



関数**bmi_category**を作成しました。

```
mybmi = bmi_category(20.5)  
print(mybmi)
```



関数**bmi_category**を呼び出します。

正常体重

課題：WebClass上に回答を入力してください

課題1

下記挨拶文のリストを用意しました。

```
greeting = ["Good morning", "Good afternoon", "Good evening", "Good night"]
```

for文を使用して、順番に言葉を表示するようにコードを書いてください。

繰り返し変数はgとします。

課題2

if、elif、else文を使用して気温によって快適さを判断するコードを書いてください。

気温の変数は、tとします。現在の気温は、15度とします。

0度以下：寒すぎます

0-10度：寒いです

10-25度：快適です

25-35度：熱いです。

35度以上：暑すぎます。

医療とAI・ビッグデータ入門

演習2-7の構成

Python基礎を学びましょう

演習2 11/16 11:35-12:20

Pythonの変数とデータの型

Pythonを使ってみましょう

演習5 12/7 11:35-12:20

患者の歯に関する病院のリアルデータの説明

演習3 11/30 11:35-12:20

プログラミング基礎

演習6

データクレンジングに必要なライブラリ（Pandas）の応用

演習4 12/7 10:40-11:25

モジュール、パッケージ、ライブラリ

演習7

データクレンジングとデータの可視化