「アルゴリズムとプログラム」

Python入門(Google Colaboratory版)

2023版　V1.1

```
def total(a):
    b = 0
    for i in a:
        b = b + i
    return(b)


x = [0, 1, 2, 3, 4, 5, 6, 7]

s = total(x)
print(s)
```

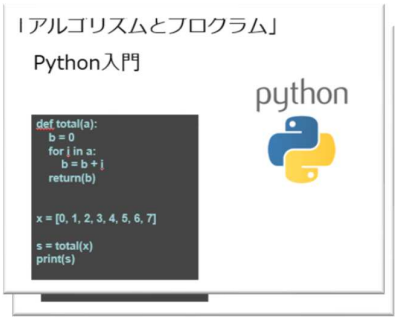
Python入門用　情報I授業用

0	どうしてPython～Pythonの起動まで
1	1行実行:表示と足し算の計算
2	1行実行: 四則演算と変数
3	1行実行とプログラムを区別しよう
4	ドルから円に変換.
5	入力した2個の数で四則演算
6	合格判断
7	合格不合格判断
8	成績A～C
9	0から10までの数を表示
10	0から10までの合計を表示
11	2からx未満までの偶数の合計

12	コインガチャを作る(リスト使用)
13	リストを使った5つの数の合計
14	15～20の説明
15	リストの中から数を探す
16	リストの一番小さい数を見つける
17	一番小さい数をリスト先頭に入替え
18	二重繰り返しで九九に挑戦
19	複雑な二重繰り返しに挑戦
20	最後のチャレンジ: 数の並び替え
21	発展1: Fizz Buzz
22	発展2:バブルソート

ただし、教科書にのっていて授業やる並び替えは大学入試で出ないだろう。これらはアルゴリズムの単なる例

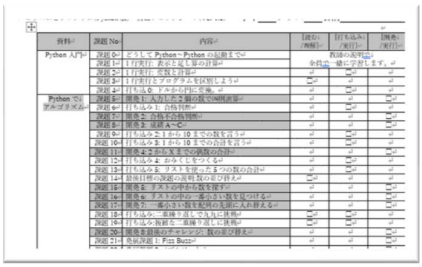
学習の進め方(教材の内容)



指示書

・Python入門
・Pythonでアルゴリズム

この中の課題を自分でやっていきます。



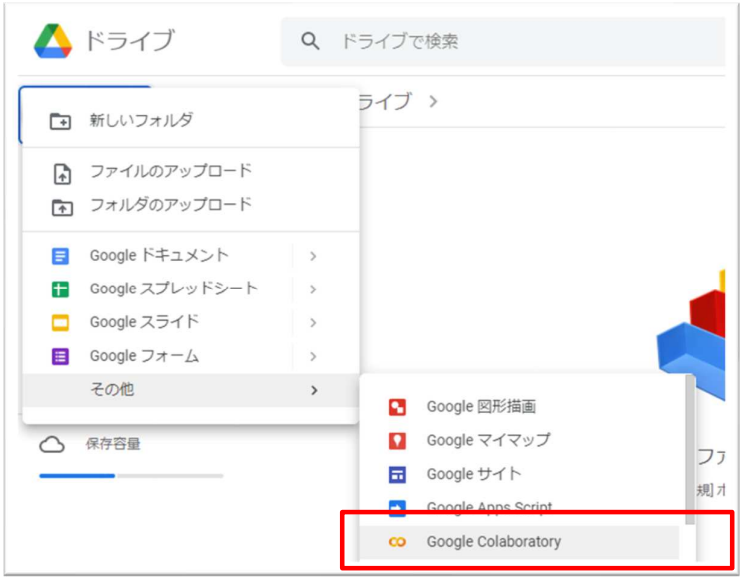
チェックシート

課題をやるか確認します。課題ができたらチェックを書き込みます。

機能部品カード

プログラムを構成するPythonの部品です。各課題にどの部品を使うか下記のように指示しるので参照してください。

実習の開始: Colaboratory使ってみよう:起動



Googleドライブのファイルを保存するフォルダーで、[新規]-[その他]-[Google Colaboratory]を指定

[Google Colaboratory]が使えるようになる。

Pythonを使ってみよう

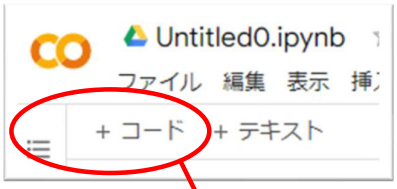
重要:日本語入力はオフにしてください。

学習で使用するColaboratoryというPythonの開発実行環境では、初めにGoogleドライブに作成した一つのファイルの中に複数のコードを入れることができます。
一つ一つのコード(プログラム)を実行することができます。

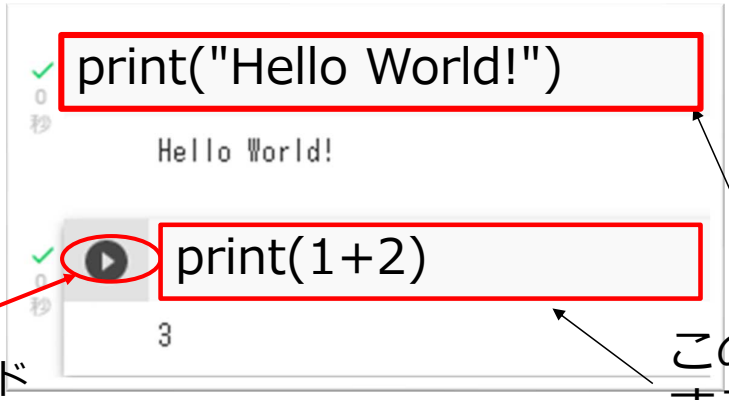
課題1　1行実行: 表示と足し算の計算

一行づつ命令を入力して、実行することができます。

初めてのPythonプログラム



新しいコード(プログラム)を追加



一つのコードを実行

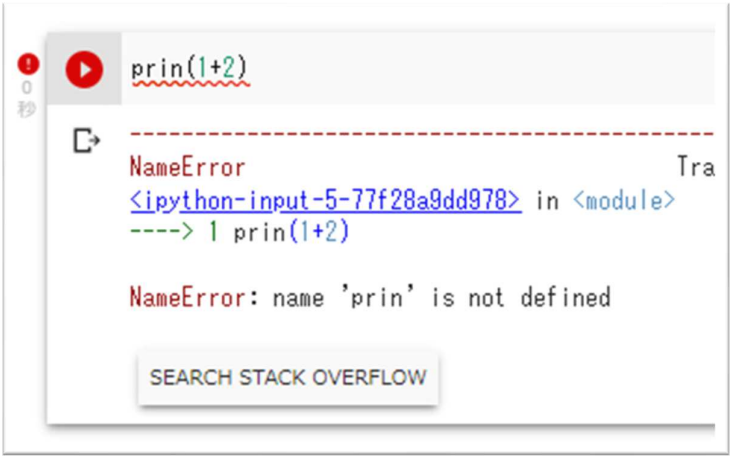
この赤枠の中を入力すると実行結果 print() 表示する

部品01

エラーが出た人はつぎのスライドを見て対応

エラーの対応(1)

プログラムを実行しているとき、エラーがあっても、次に正しく入力すれば問題ありません。



printの綴りが間違ってエラーになりますが、問題ありません。

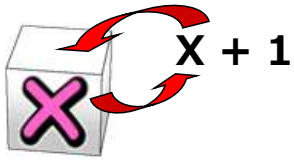
正しく打ち直せばOKです。

課題2　1行実行：変数と四則演算

変数

x = 10

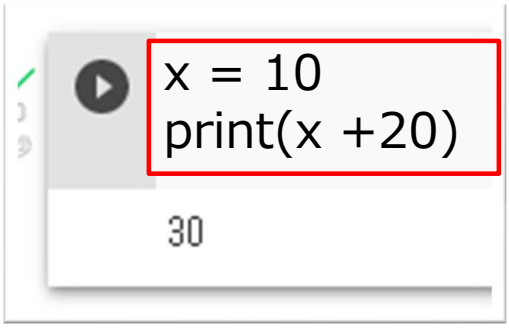
Xと名前をつけた箱(変数)に1を入れる



x = x +1

(x←x + 1のイメージ)

初めにxの箱(変数)の中を取り出し+1する。計算結果をXの箱(変数)に入れなおす。



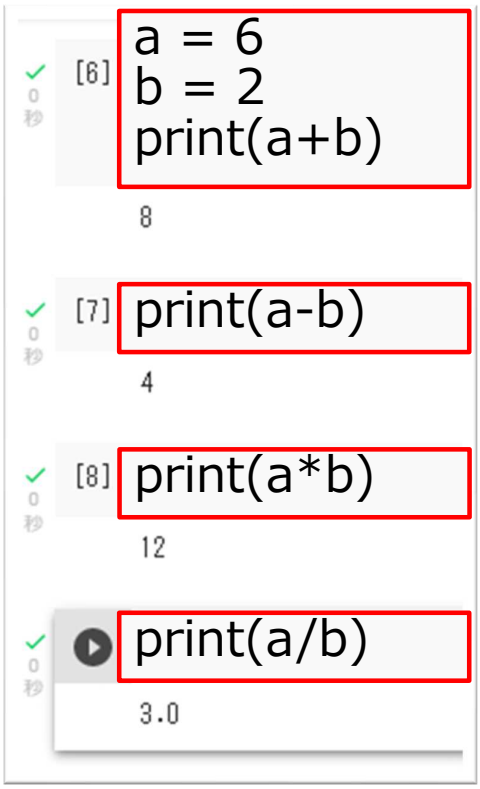
入力して実行

変数に入れた数を、計算で使うことができます。

変数への代入は普通の数学の=とは違う意味なのでイメージを示してみました。

1行実行: 変数と四則演算(その1)

下の赤枠の部分を入力して、変数と四則演算の動作を確認しよう



部品01　部品02

足し算 +	+
引き算 -	-
掛け算 x	*
割り算 ÷	/

エラーが出ても
正しく打ち直せばOKです。

1行実行: 変数と四則演算(その2)

プロンプトの後を入力して試してみよう

```
x = 1
print(x)
```
1

```
x = x + 1
print(x)
```
2

青色は実行結果

重要: 変数名の付け方
使える文字
・小文字英字　・大文字英字
・数字　・_(アンダーバー)
先頭に数字は使えない
〇 abc 〇kakaku 〇 total_a
× 1ban ×take@jp ×日本語

文字列の定義

確認

文字列の扱いを確認しよう

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

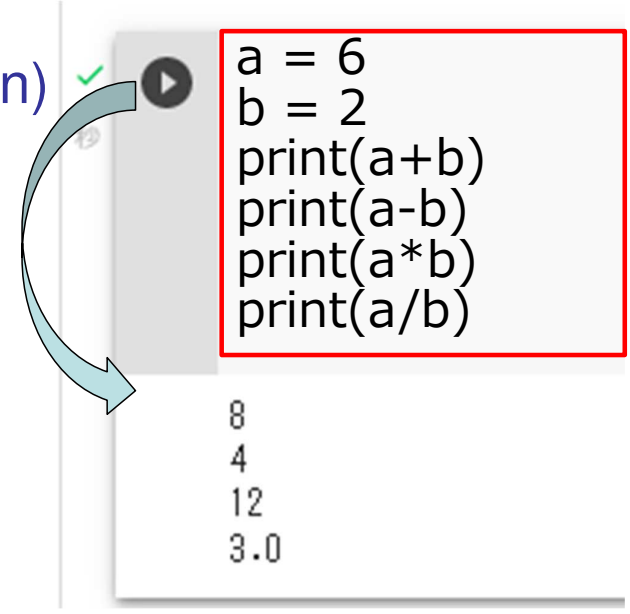
```
a = "Hello"
b = 'World!'
c = a + b
print(c)
```
HelloWorld!

・文字列は”(ダブルクォート)又は’(シングルクォート’で囲んで定義します。
・必ず同じ” 又は’でくくります。
・文字列同志をくっつける時は + を使います。
・文字列としては日本語も使用できます。

課題3 打ち込み1:1行実行とプログラムを区別しよう

複数の命令をまとめて一つのプログラムにして
実行してみよう。

実行(Run)



プログラム
(命令の集まり)

プログラムが実行され、結果が表示される。

打ち込み1:1行実行とプログラムを区別しよう(その3)

実行してエラーが出る場合。
1行づつエラーが出ると止まります。

重要:
Pythonは大文字・小文字を区別します

```
---------------------------------------------------------------
NameError                                 Traceback
<ipython-input-13-e51ce7828a0c> in <module>
      1 a = 6
      2 b = 2
----> 3 prit(a+b)
      4 print(a-b)
      5 print(a*b)

NameError: name 'prit' is not defined
```

・Printの命令が、pritと誤ってエラーになった。

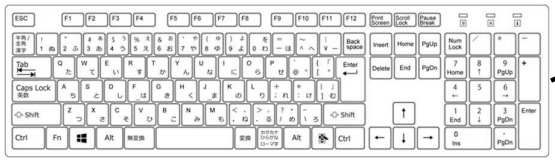
```
8
4
12
---------------------------------------------------------------
NameError                                 Traceba
<ipython-input-15-eebd90d0c90a> in <module>
      4 print(a-b)
      5 print(a*b)
----> 6 print(y/b)

NameError: name 'y' is not defined
```

・5行目まではエラーがなくて正常に実行したが、
6行目に使っていない変数yがあったので、エラーになった。

・１個づつエラーの箇所を修正して、再度実行していきます。

課題4　打ち込み2: ドルから円に変換。

1＄＝130円 とする



```
$?
10
1300
```

プログラムを実行して、$?が出た後、
キーボードから数を入力すると、
それに130を掛けた数を表示する
プログラムを打ち込みます。

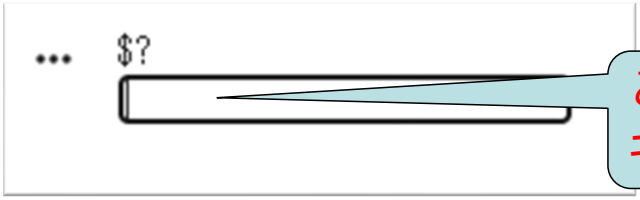
打ち込み2: ドルから円に変換。(その1)

```
[16] print(“$?”)
     a = int(input())
     print(a*130)

     $?
     10
     1300
```

プログラム
(命令の集まり)

部品 01　部品 02　部品 03

新しくプログラムを打ち込んで実行してみよう。



このように表示されたら、
キーボードから入力

補足: 情報を入力する時は上図のような画面になります。

これで「Python入門」の学習は終わりです。
続いて「 Pythonでアルゴリズム」で
学習を続けてください。

「Pythonでアルゴリズム」の準備

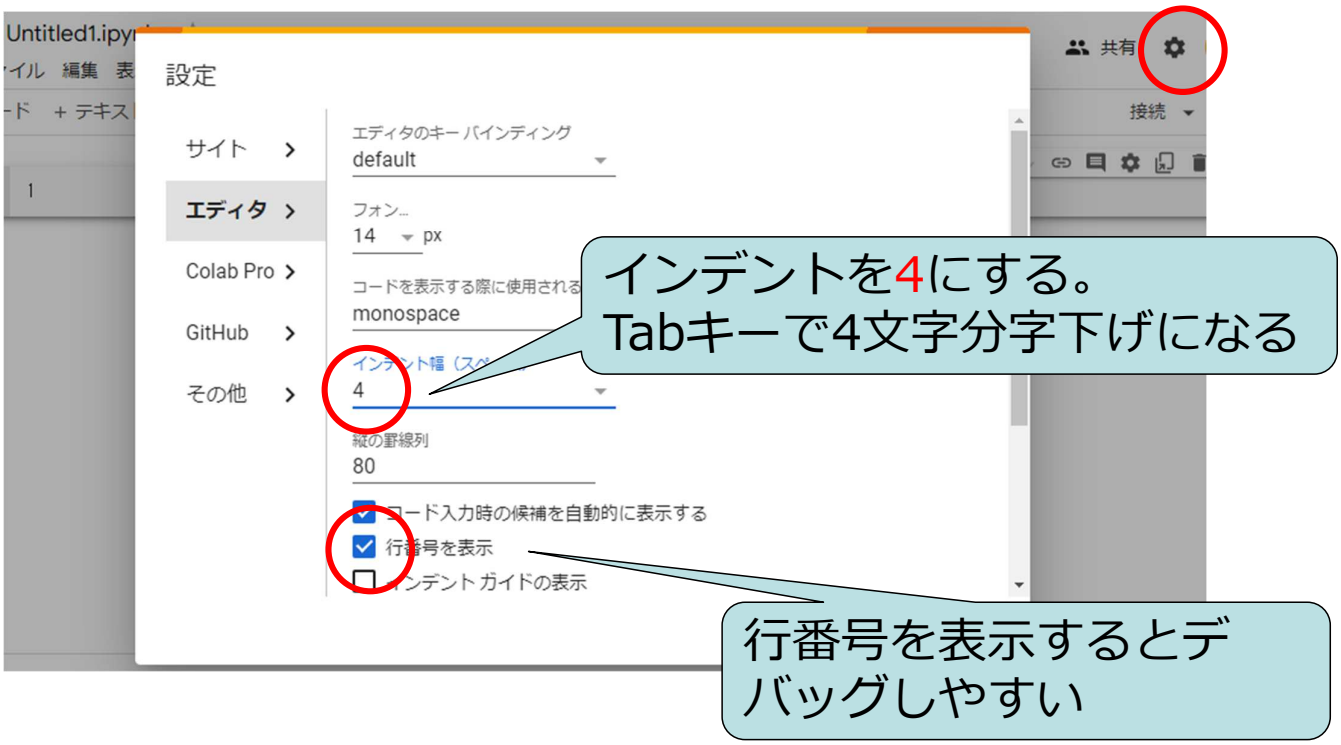
```
s = 0
for i in range(11):
    print(i)
    s = s + i
print("Goukei")
print(s)
```

字下げは重要

一つのブロックとしてforの中身として実行

同一のレベルで順番に実行される

インデントの変更



課題5　開発1：入力した2個の数で四則演算

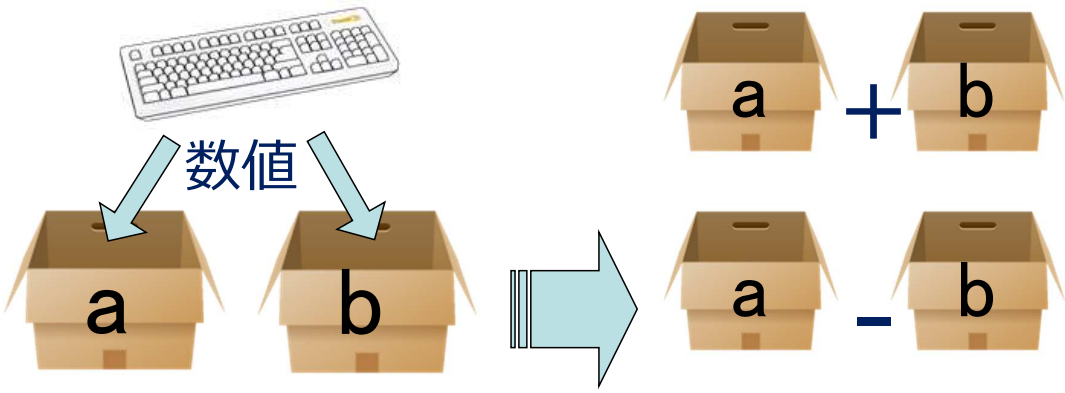
二つの数をキーボードから入力して、その足し算、引き算、掛け算、割り算をするプログラム

```
10
20
30
-10
200
0.5
```

課題1:
2個の数をキーボードから入力して、その足し算、引き算、掛け算、割り算の結果を順番に表示します。

このように表示されたら、キーボードから入力

次見て



課題1:
2個の数をキーボードから入力して、その 足し算、引き算、掛け算、割り算の結果を順番に表示します。

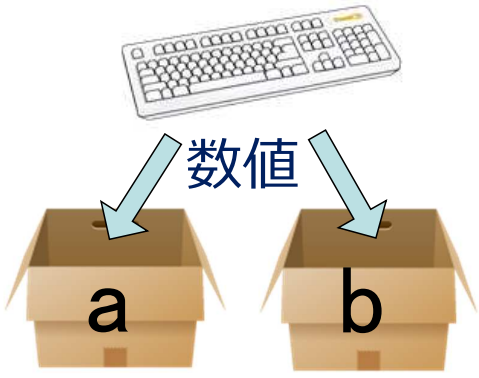
部品01　部品02　部品03

次見て

開発1: 入力した2個の数で四則演算(その1)

```
a = 6
b = 2
print(a + b)
print(a - b)
print(a * b)
print(a / b)
```

足し算、引き算、掛け算、割り算は課題3でやっています。
課題3ではプログラムの中で数値を設定しいていますが、これを入力します。



部品03

変数Aと変数Bに入力する方法は、部品カード03を見てください。
また課題4も数値の入力を行っています。

課題6　打ち込み3: 合格判断

```
print("Tokuten?")
a = int(input())
if a > 70:
    print("Goukaku")
```

そのまま打ち込む
プログラム

```
1 print("Tokuten")
2 a = int(input())
3 if a > 70:
4     print("Goukaku")
```

```
Tokuten
100
Goukaku
```

```
Tokuten
40
```

部品04　部品カード04も見てください。

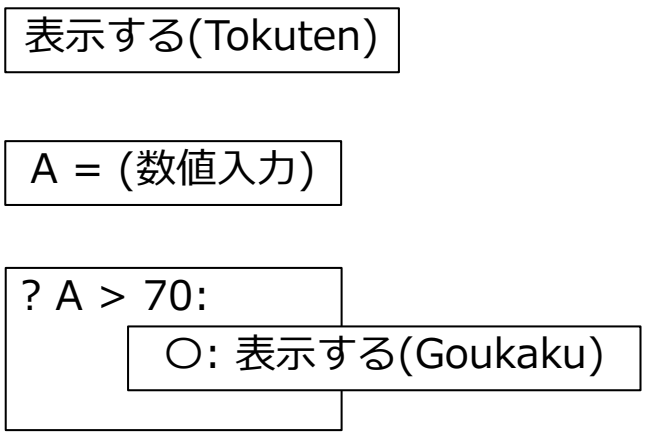
変数aに数値を入力して、70より大きければ(70は入らない)、Goukaku(合格)と表示する。

次見て

打ち込み3: 合格判断

```
print("Tokuten?")

a = int(input())

if a > 70:
    print("Goukaku")
```

この教材で使っている
プログラムの図式



重要　if a > 70:
----print("Goukaku")

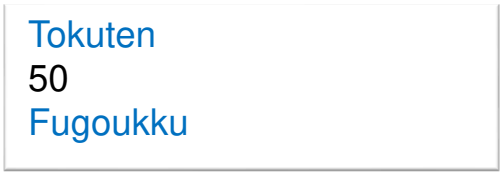
TPOINT

Tab

Pythonのルールでif命令の中身の命令は、字下げ(通常は半角の空白4文字)する。
TabキーでもOK

課題7　開発2: 合格不合格判断

変数aに数を入力して、
もし、70より大きい**なら**
(70は入らない)、
Goukaku(合格)と表示します。
そう**でなければ、**
Fugougaku(不合格)と表示します。



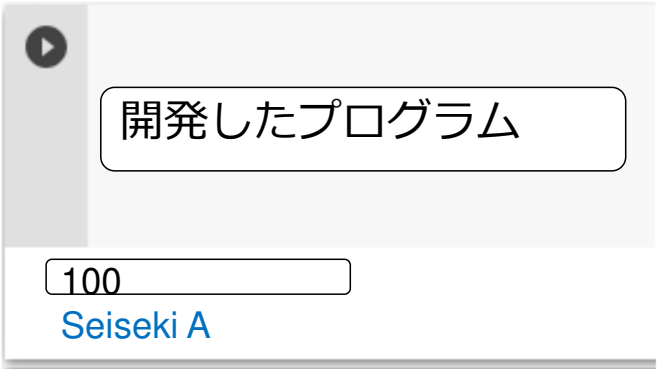
部品
05

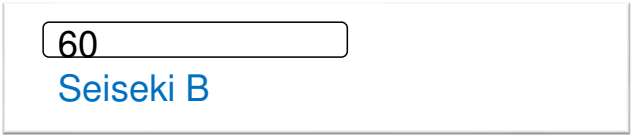
TPOINT

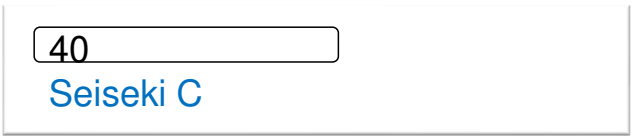
ヒント: 部品05を参考にして作ってみてください。
部品の内容そのままで使えるわけではありません。
課題6のプログラムを元にして作ると簡単です。

課題8　開発3: 成績A～C

変数Aに数を入力して、**もし**、
70より大きい**なら**(70は入らない)、
Seiseki A(成績A),
50より大きい**なら**(50は入らない)、
Seiseki B(成績B),
それ以外(50以下だったら)、
Seiseki C(成績C),
と表示します。







部品
06

ヒント: 課題7を流用して作る。
部品06を参考にして作ってみてください。

課題9　打ち込み4: 0から10までの数を表示

Range()で１０までを
指定する(11未満)

```
for i in range(11):
    print(i)
```

そのまま打ち込む
プログラム

```
1 for i in range(1,11):
2     print(i)

1
2
3
4
5
6
7
8
9
10
```

0,1,2,3,・・・,10
までの数を表示します。

部品
07

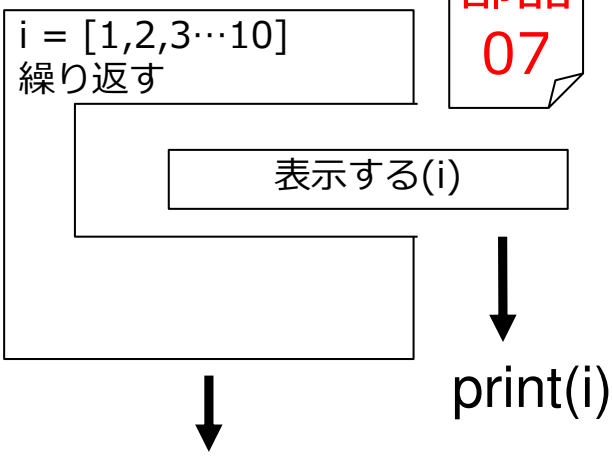
次見て

プログラムと図式

理解

プログラムもだんだん複雑になってきます。プログラムがどのような部品でくみあがっているか分かりやすくするため、今後ヒントで図式を示します。

部品
07




```
for i in range(11):
    print(i)
```

```
for i in range(11):
```

課題10　打ち込み5: 0から10までの合計を表示

```
s = 0
for i in range(11):
    print(i)
    s = s + i
print(s)
```

```
1 s = 0
2 for i in range(1,11):
3     print(i)
4     s = s + i
5 print(s)
```

```
1
2
3
4
5
6
7
8
9
10
55
```

そのまま打ち込む
プログラム

部品
07

1,2,3,・・・,10までの数を表示して、最後にその合計を表示する。

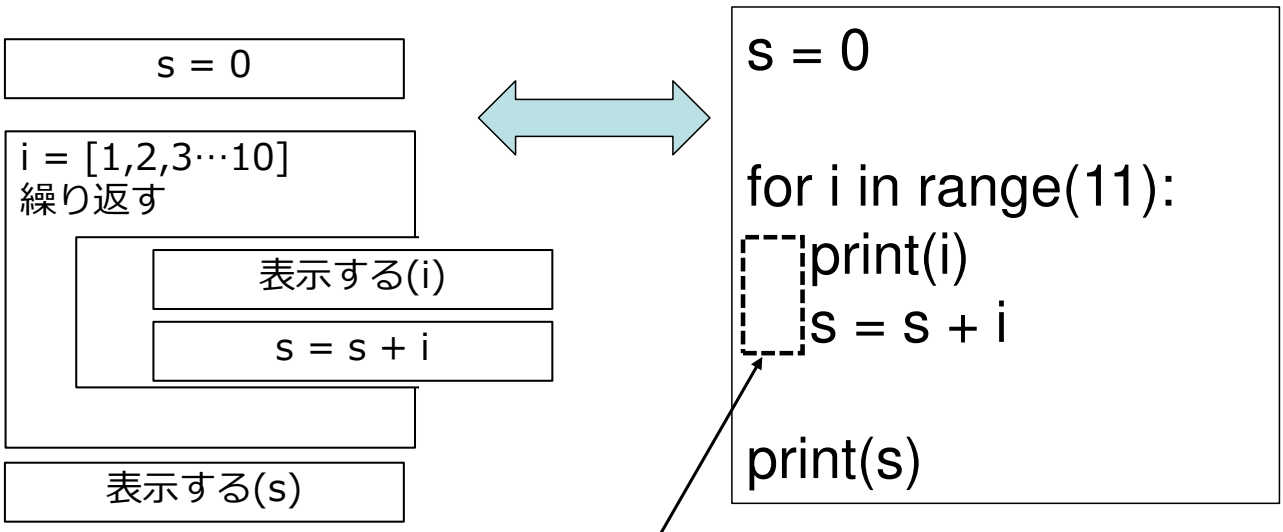
ヒント: 課題9
を流用して作る。

次見て

プログラムの図式

TPOINT

「 1から10までの合計」は次のような図式になります。



字下げしていることで、
for命令の中で、この二つの
命令が繰り返される。

課題11　開発4: 2からx未満までの偶数の合計

xの値を入力して、2からx未満偶数の合計を求めるプログラムを作成してください。

例: x = 7の場合、 2+4+6
　　x = 12の場合、 2+4+6+8+10

TPOINT

重要! 下記の順番で作って!!

課題10「0から10までの合計」のプログラムを流用してして作る。

手順1: 次のスライド

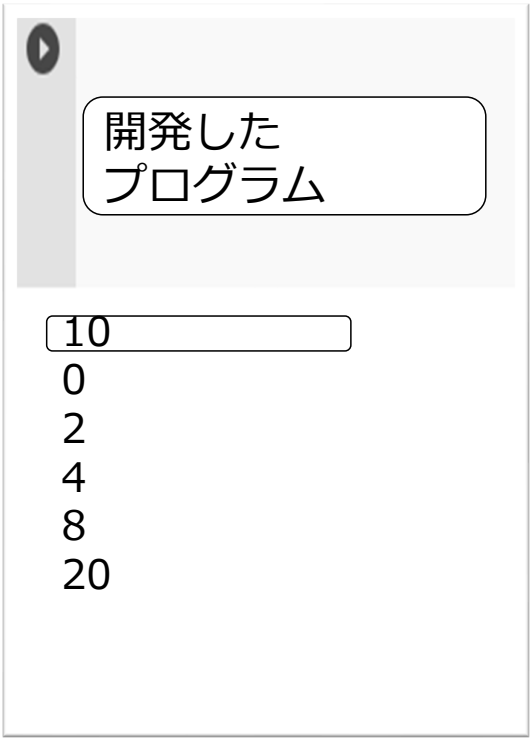
まず、2から10未満の偶数の合計を求めるプログラムを作ります。一か所だけ変更します。

手順2: 次の次のスライド

次に、xを入力して、x未満の合計を求めるプログラムを作ると楽かもしれません。

次見て

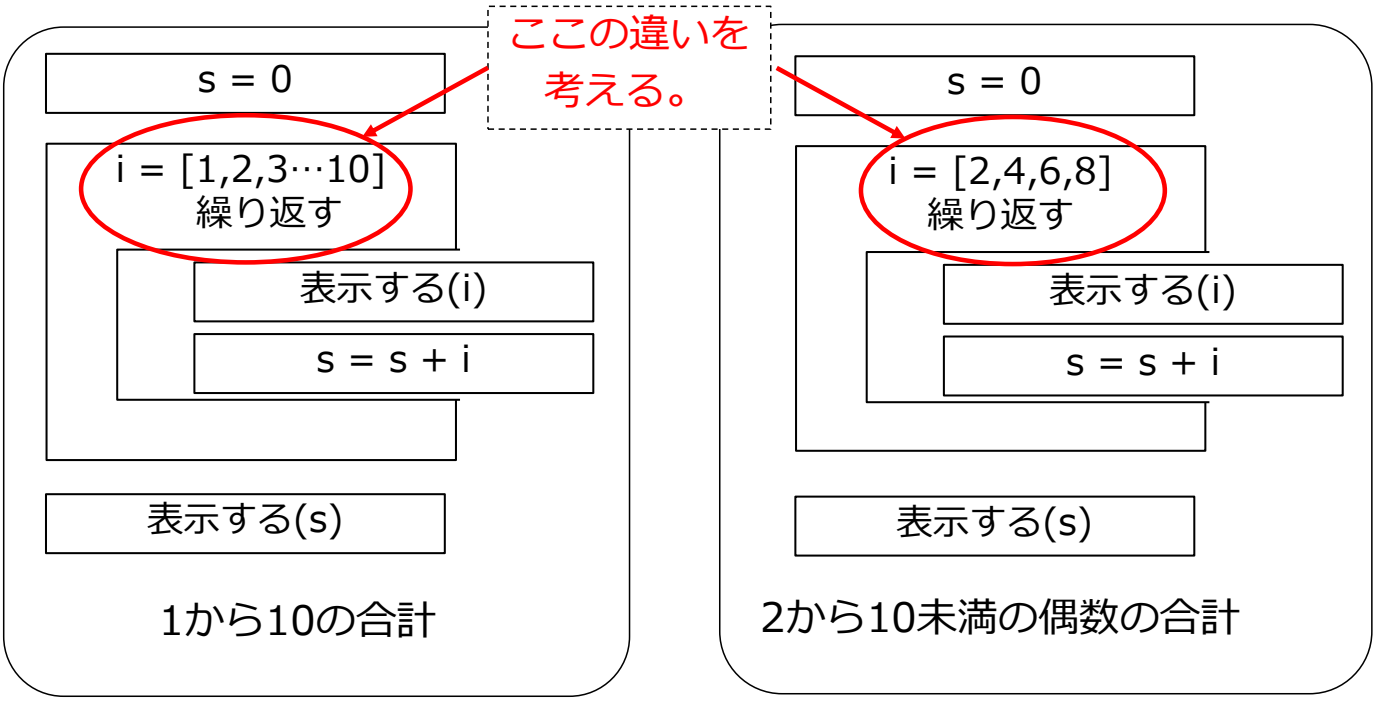
実行例



10と入力すると
0,2,4,8と表示して最後に
合計の20を表示する

次見て

ヒント2: 2から10未満の偶数の合計

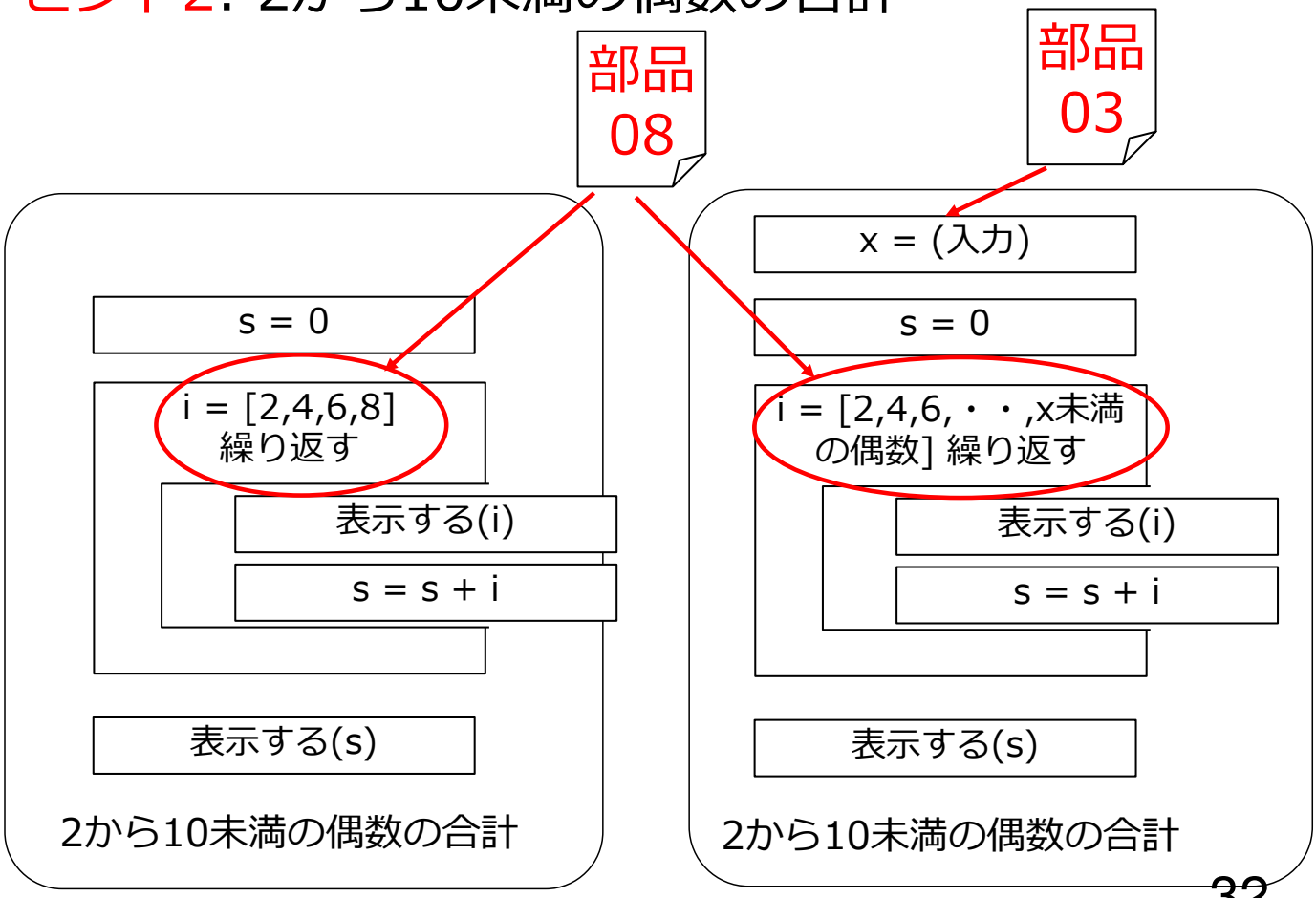


部品08

2, 4, 6,・・・となるように
部品08を少し変更して使います。

次見て

ヒント2: 2から10未満の偶数の合計



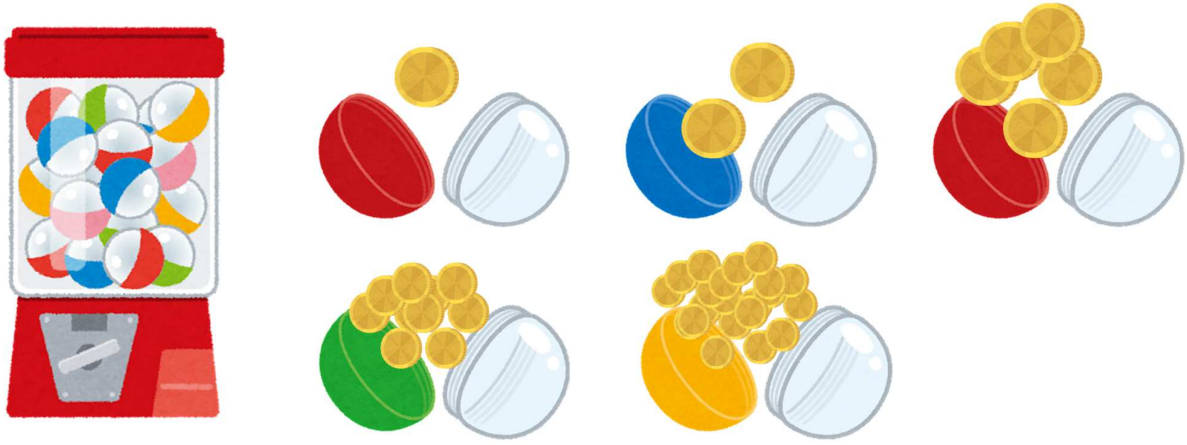
たくさんの数のの処理

次からたくさんの数を処理する
プログラムをつくっていきます。

今まで、数を入力するのに変数をつくっていきました。課題11では、x,s,iの3つの変数を使って合計を計算しました。
ただし、が、たくさんの数を入力するには、a,b,c,d,e・・・・と多くの変数を使うのは大変です。

こんな時にリスト(配列)使うと便利です。
次からリスト(配列)を学習していきます。

課題12　打ち込み6: コインガチャを作る



ランダムに、あらかじめ設定しておいたコインの数をランダムに表示するプログラムを作成します。

ランダムな数をつくる
import random

random.randint(開始番号、終了番号)

次見て

打ち込み6: コインガチャを作る

```
import random
d = [1, 2, 5, 10, 100]
i = random.randint(0, 4)
print("Coin=")
print(d[i])
```

そのまま打ち込む
プログラム

import randomで
random.randint()
が使えるようにします。

打ち込んだプログラム

```
Coin=
100
```

```
Coin=
5
```

```
Coin=
2
```

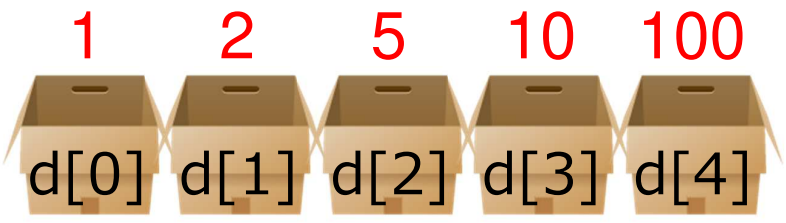
```
Coin=
100
```

次見て

打ち込み6: コインガチャを作る

リストはdに番号がついている、さくたんの箱のイメージです。
個々の箱のデータを使用する場合は、箱についた番号[0]～[5]
を指定します。

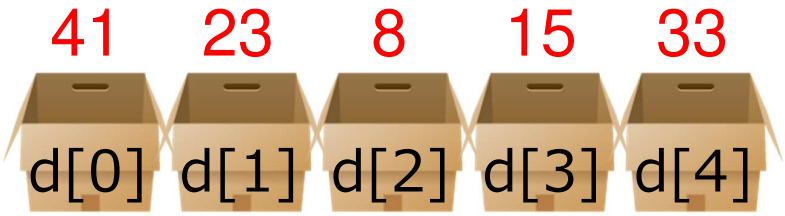
d = [1, 2, 5, 10, 100]



i = random.randint(0, 4)　← iに0～4のランダムな数か入る
d[i]　←　dのi番目の箱の内容

課題13 打ち込み7: リストを使った5つの数の合計

d = [41, 23, 8, 15, 33]



41+ 23+8+15+33 合計 120

いきなりプログラムが難しくなりました。あらかじめリストdに入れておいた5個の数の合計を計算します。

次見て

打ち込み7: リストを使った5つの数の合計

```
d = [41, 23, 8, 15, 33]
s = 0
for i in range(5):
    print(d[i])
    s = s + d[i]
print("Goukei")
print(s)
```

そのまま打ち込むプログラム

部品09

打ち込んだプログラム

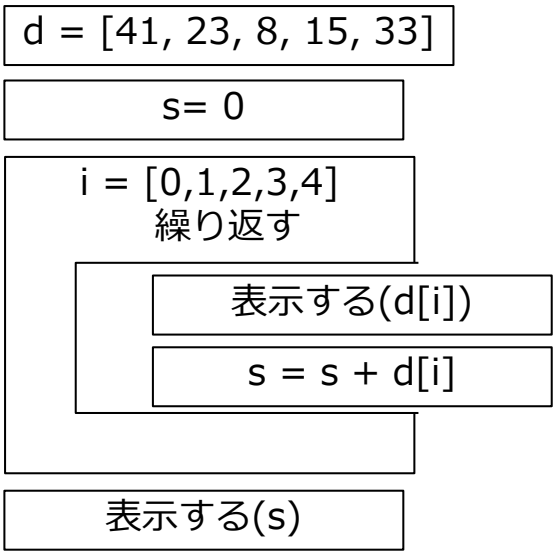
```
41
23
8
15
33
Goukei
120
```

ヒント: 課題10を流用して作る。

D[]の中の数をひとつづつ表示して、最後にその合計を表示する。

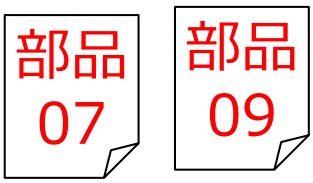
次見て

打ち込み7: リストを使った5つの数の合計

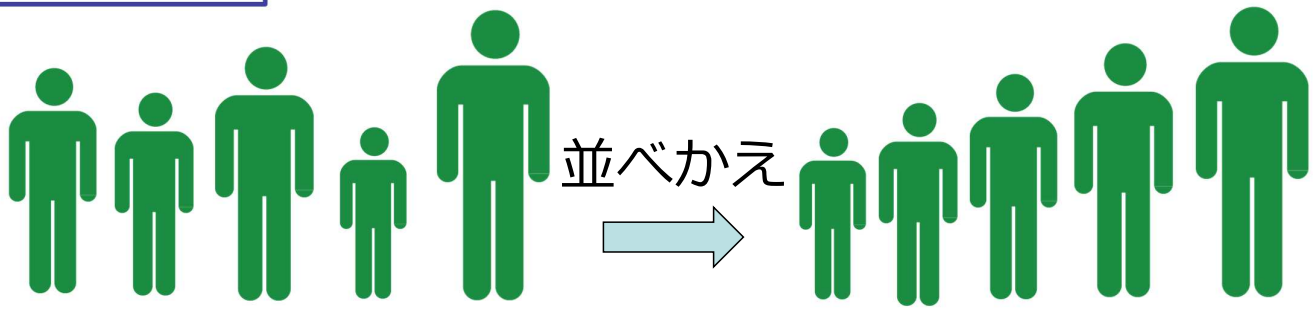



```
d = [41, 23, 8, 15, 33]
s = 0

for i in range(5):
    print(d[i])
    s = s + d[i]

print("Goukei")
print(s)
```

リストdに5個の数を入れて、変数Sに加えていって合計を計算



課題14　課題15～20の説明



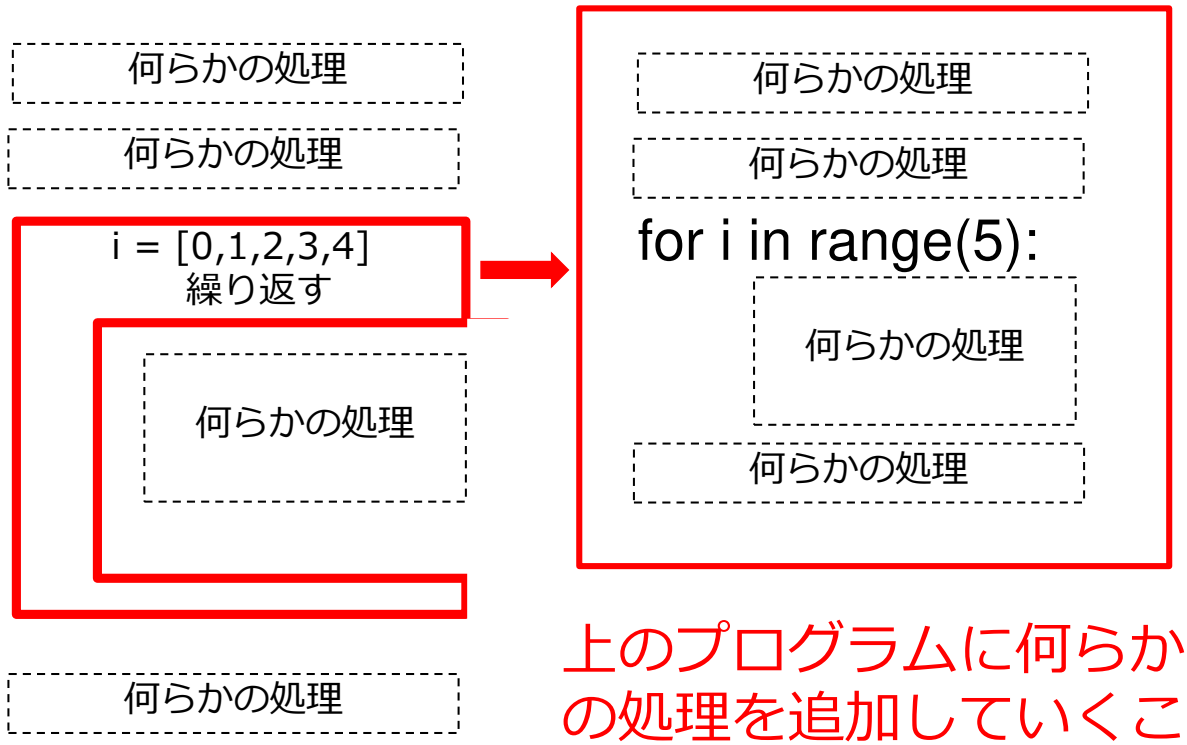
課題20では基本的なアルゴリズムである並び替え(ソート)のプログラムを作ります。いきなり作るのは難しいので、課題15～19をやっていきます。

課題15: リストの中から数を探す
課題16: リストの中の一番小さい数を見つける
課題17: 一番小さい数を配列の先頭に入れ替える
課題18: 二重繰り返しで九九に挑戦
課題19: 複雑な二重繰り返しに挑戦

次見て

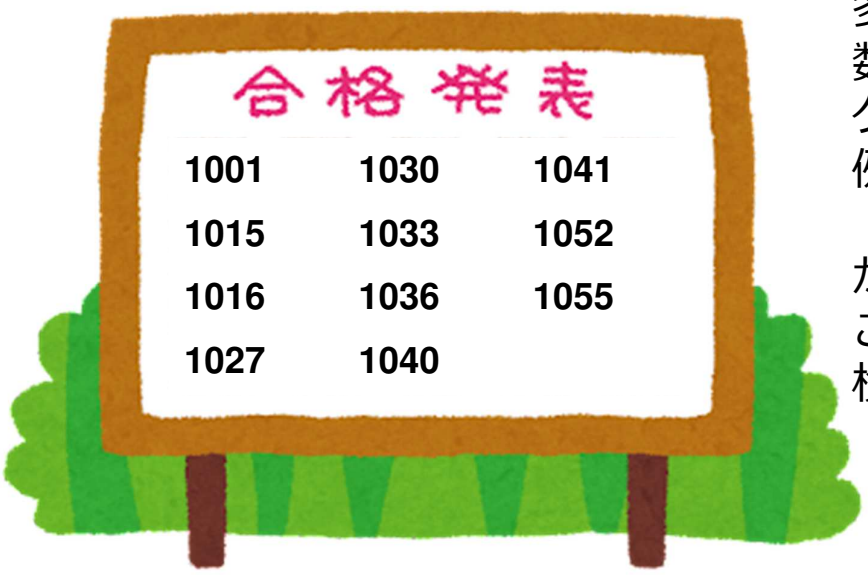
非常に重要　TPOINT

課題15～20までは、5個の箱のリストを順番に処理するため、次のような図式が基本になります。



上のプログラムに何らかの処理を追加していくことになります。

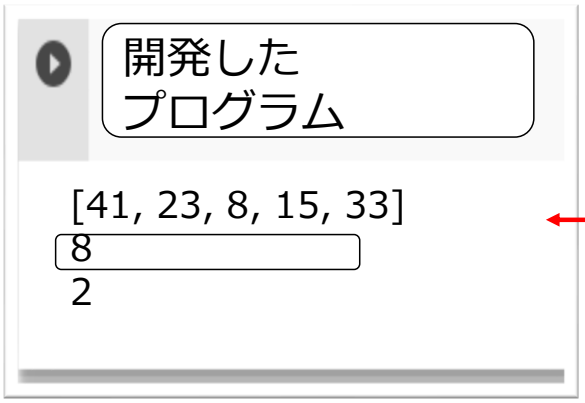
課題15　開発5: リストの中から数を探す



多くの数の中から、ある数があるか見つけるプログラムです。
例えば、合格発表の中に「1033」があるかどうか判断します。
このようなプログラムを検索といいます。

次見て

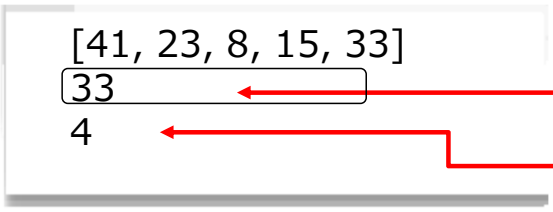
実行例



検索というこんな動作をするプログラムを作ります

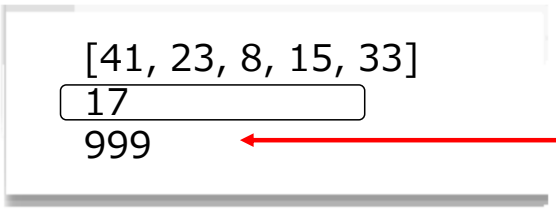
```
d = [41, 23, 8, 15, 33]
print(d)
```

あらかじめdに5つの数をいれておき、表示します。



変数aに探す数を入力します。

33は、d[4]に入っているで、4を表示



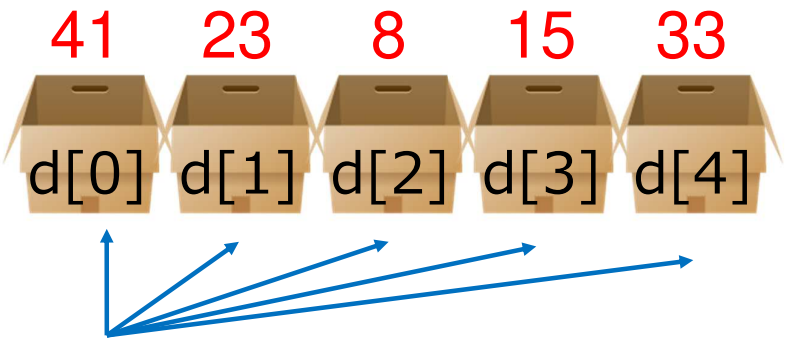
17はdに無いので、999を表示

次見て

開発5: リストの中から数を探す

以下のようにして数を探します。

d = [41, 23, 8, 15, 33]



変数aに探す数を入力します。

i を[0,1,2,3,4]で繰り返しながら、aとd[i]を比較します。

aとd[i]が等しい場合

x に iを入れます。見つからない場合に、初めに999を入れています。

次見て

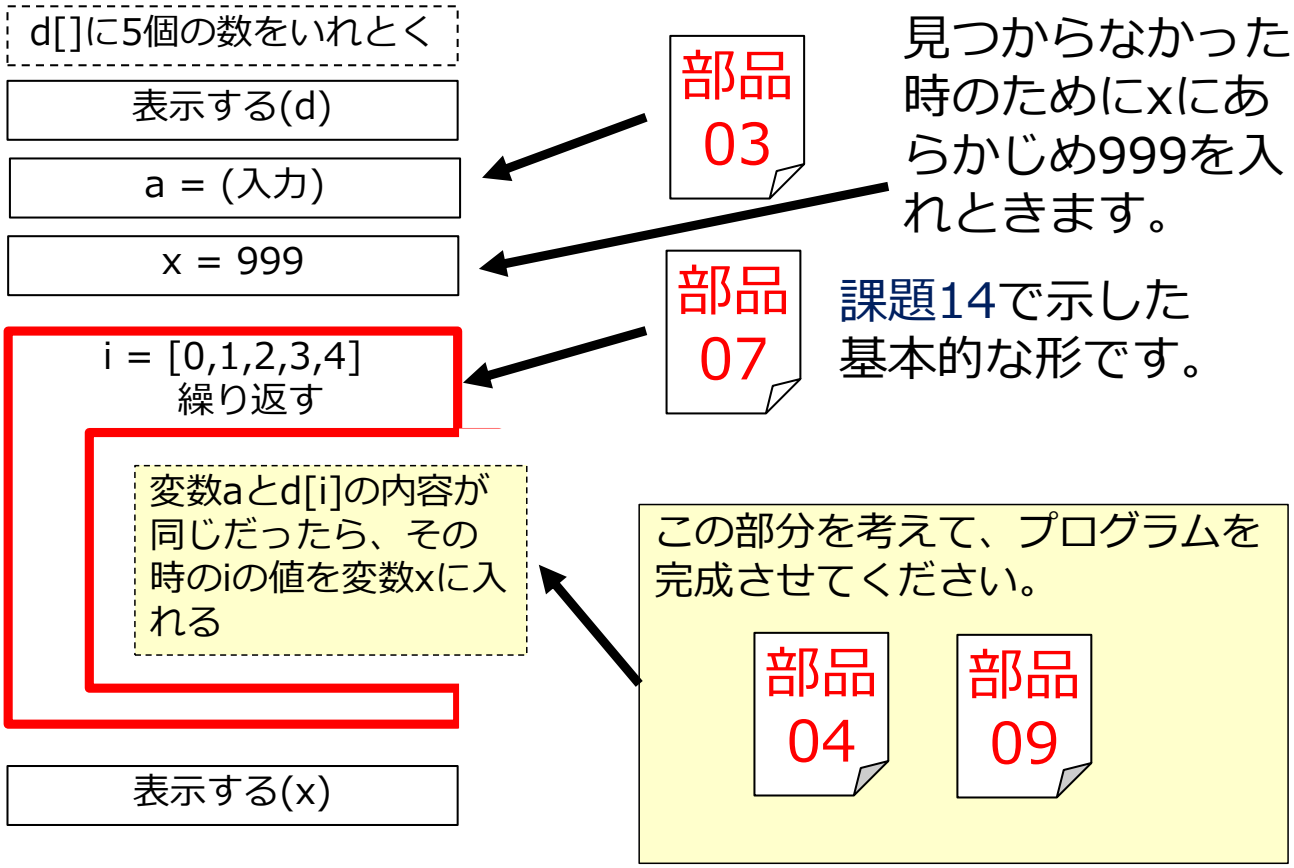
重要: if での条件指定リストの中から数を探す

部品
04

```
if x == y :
```

x == y	**x と y が等しい**
x != y	x と y が等しくない
x > y	x は y よりも大きい
x < y	x は y よりも小さい
x >= y	x は y と等しいか大きい
x <= y	x は y と等しいか小さい

開発5: リストの中から数を探す



課題16　開発6: リストの中の一番小さい数を見つける

開発した
プログラム

```
[41, 23, 8, 15, 33]
8
```

d = [41, 23, 8, 15, 33]
print(d)
あらかじめdに5つの数をいれておき、表示します。
一番小さい数をxに入れておいて表示しています。

```
[41, 23, 78, 15, 33]
15
```

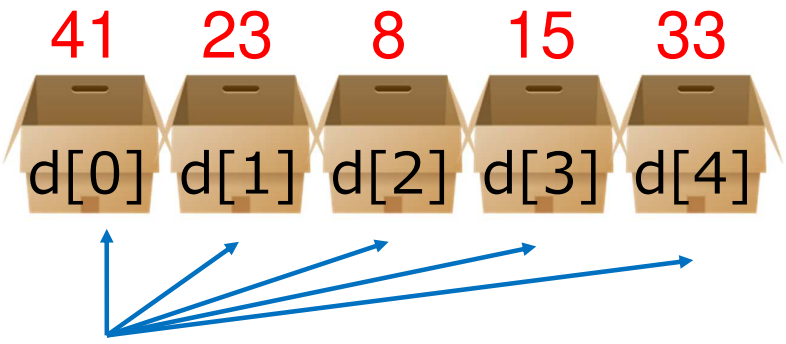
プログラムを変更して
d = [41, 23, 78, 15, 33]
にしているので、一番小さい15を表示しています。

次見て

開発6: リストの中の一番小さい数を見つける

以下のようにして数を探します。

d = [41, 23, 8, 15, 33]



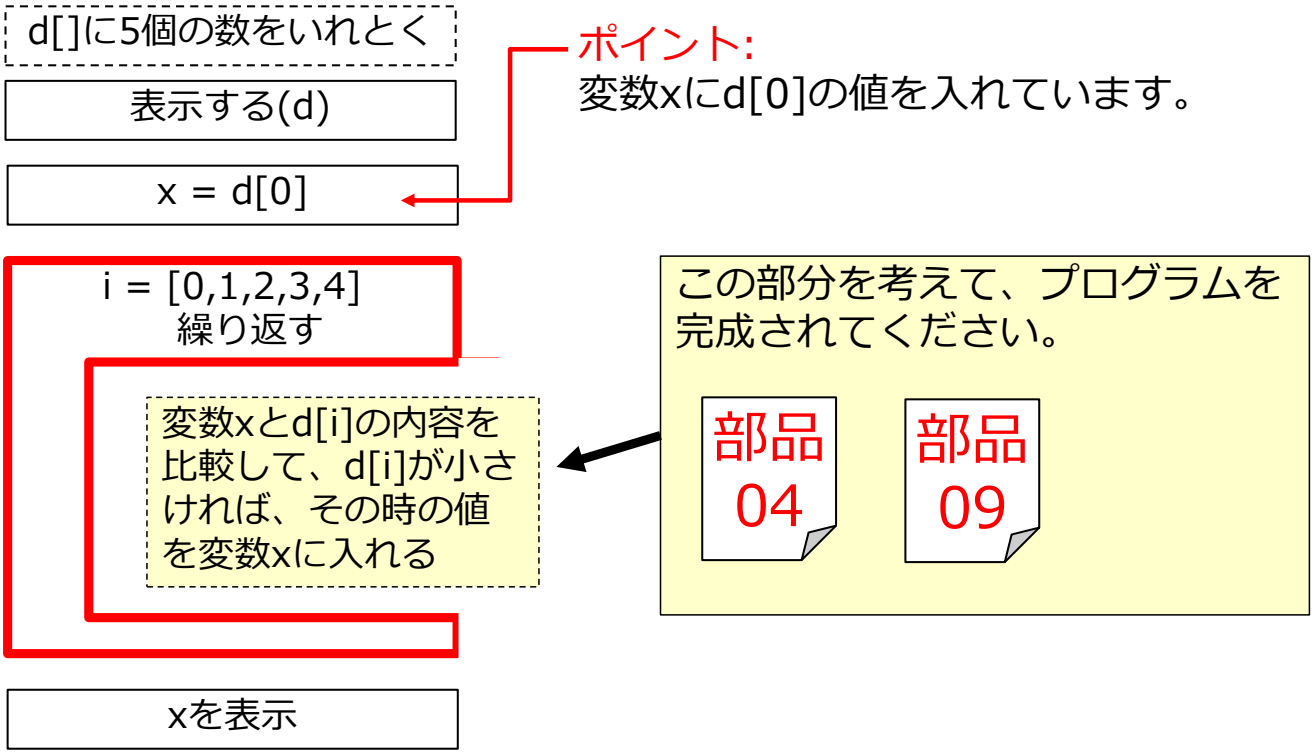
変数xに一番小さい数を入れます

i を[0,1,2,3,4]で繰り返しながら、xとd[i]を比較します。

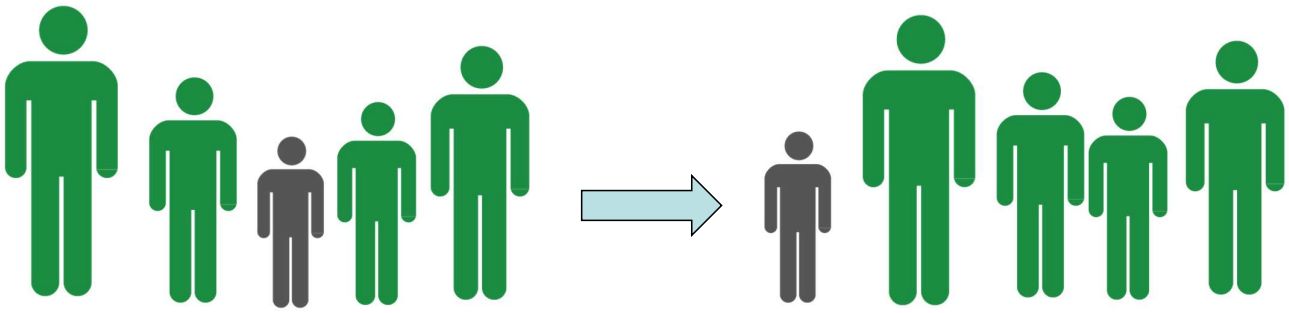
d[i]がxより小さい場合
Xの内容をd[i]に変えます。

次見て

開発6: リストの中の一番小さい数を見つける



課題17 開発7: 一番小さい数を配列の先頭に入れ替える



d = [41, 23, 8, 15, 33]
数がバラバラに入っている

d = [8, 41, 23, 15, 33]
配列の先頭に一番小さい数
が移動しているが、他の数
はすべて残っている

次見て

課題17 開発7: 一番小さい数を配列の先頭に入れ替える

開発した
プログラム

```
[23, 41, 8, 15, 33]
[8, 41, 23, 15, 33]
```

d = [41, 23, 8, 15, 33]
print(d)
あらかじめdに5つの数をいれておき、表示します。

処理後にprint(d)で表示。一番小さい8がd[0]に入っているが、すべての数は残っている。

```
[41, 23, 58, 15, 33]
[15, 41, 58, 23, 33]
```

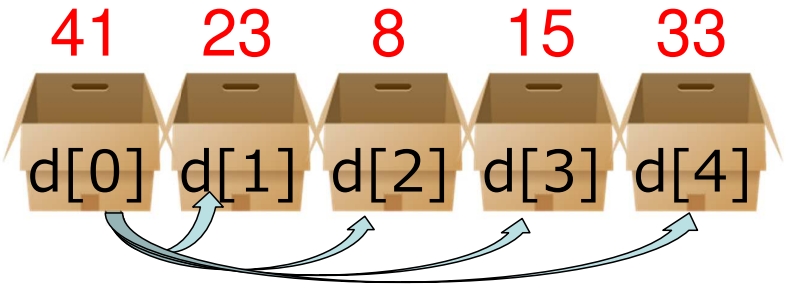
プログラムを変更して
d = [41, 23, 58, 15, 33]
にしているので、一番小さい15をd[0]に入れている

次見て

開発17: 一番小さい数を配列の先頭に入れ替える

以下のようにして数を入れ替えています。

d = [41, 23, 8, 15, 33]

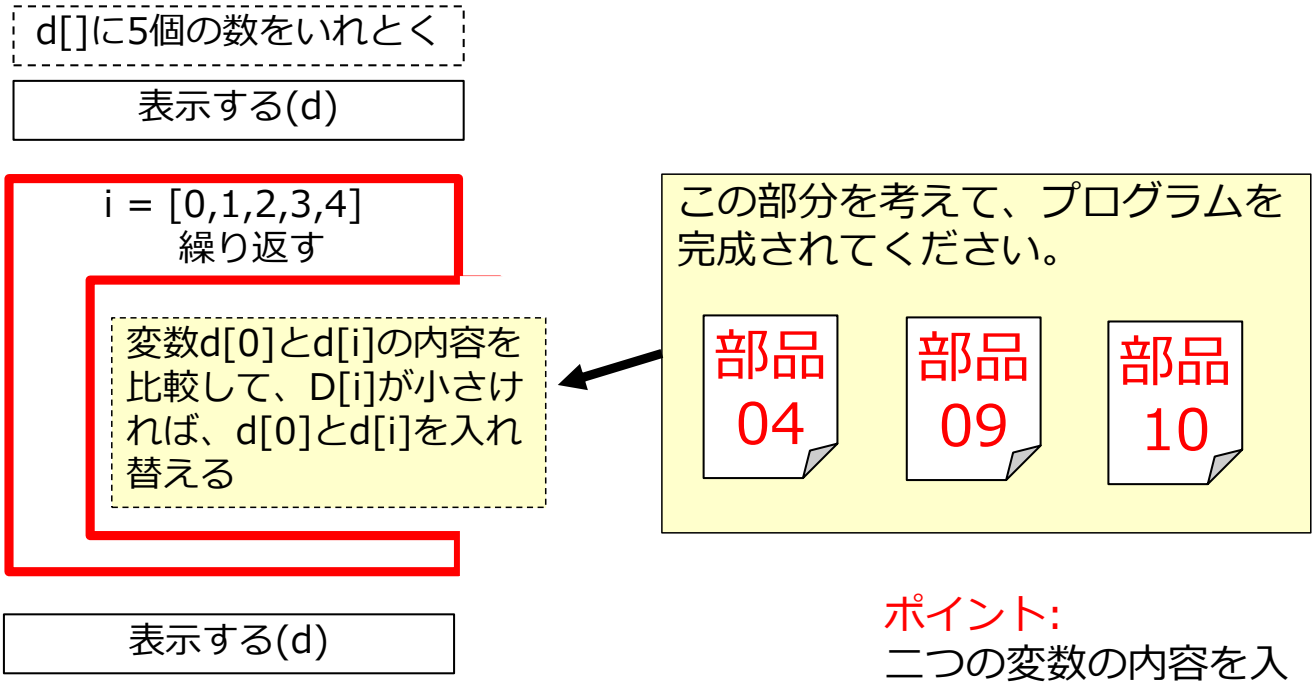


i を[0,1,2,3,4]で繰り返しながら、d[0]とd[i]を比較します。

d[i]がd[0]より小さい場合
d[i]とd[0]の内容を入れ替える。

次見て

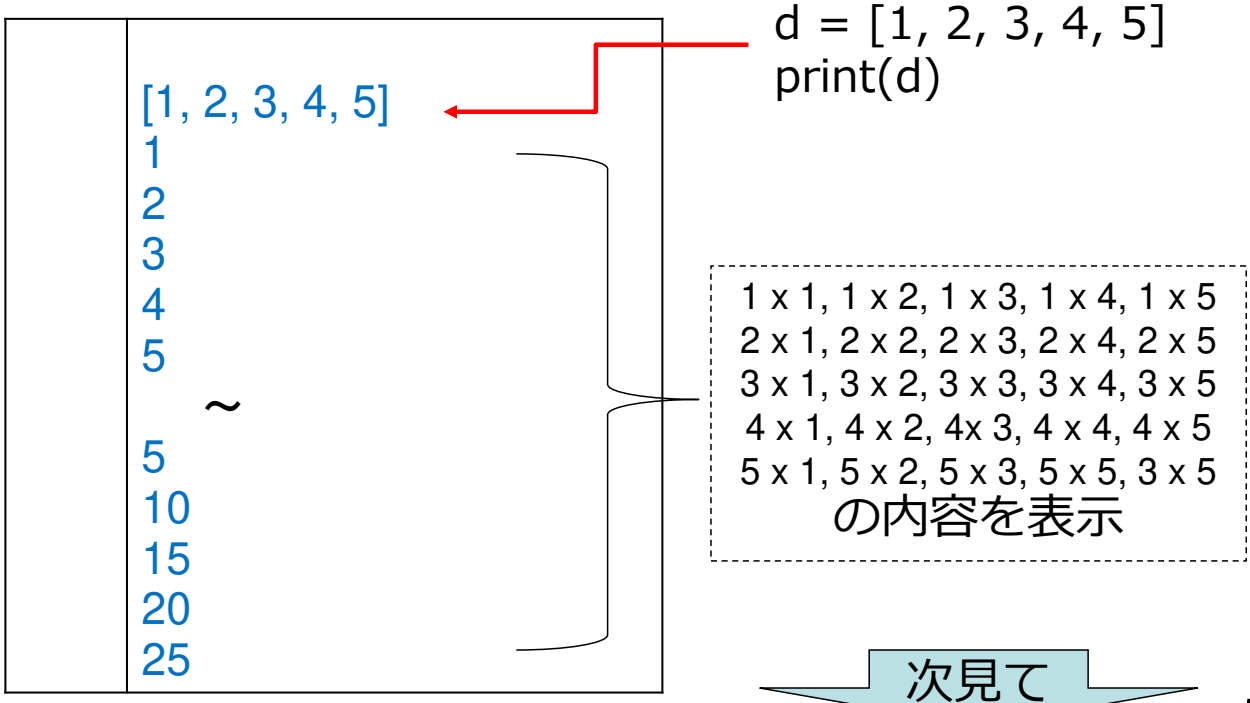
開発7: 一番小さい数を配列の先頭に入れ替える



ポイント:
二つの変数の内容を入れ替える時に注意が必要です。部品10を参照してください。

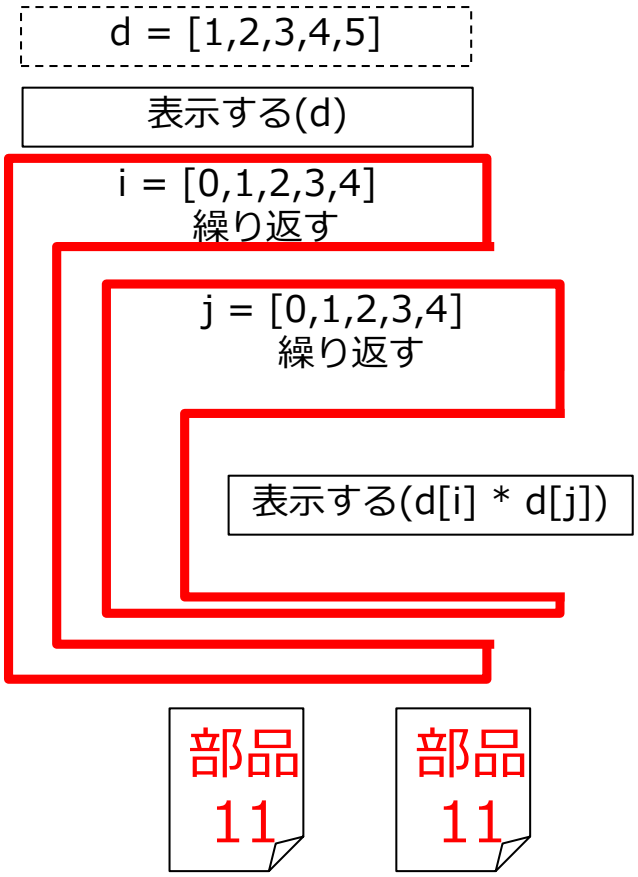
課題18　打ち込み8:二重繰り返しで九九に挑戦

二重繰り返しを使って、九九(1から５の段まで)を表示するプログラムを作ってみましょう



次見て

打ち込み8:二重繰り返しで九九に挑戦




```
d = [1, 2, 3, 4, 5]
print(d)

for i in range(5):
    for j in range(5):

        print(d[i] * d[j])
```

そのまま打ち込む
プログラム(行は空けない)

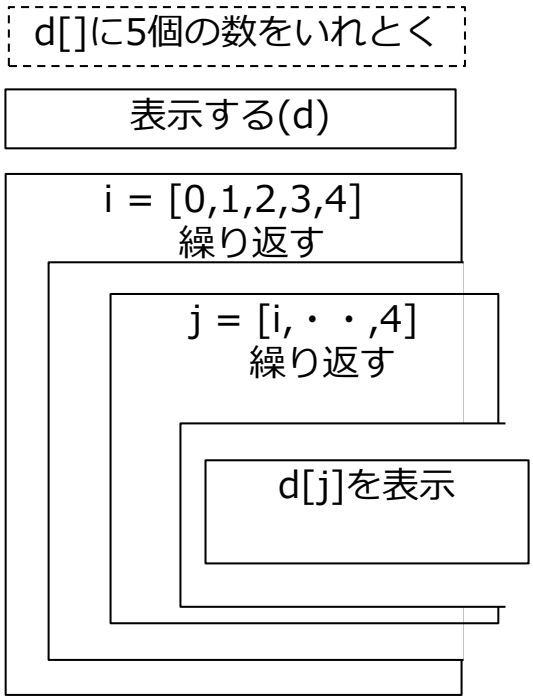
課題19　打ち込み7:複雑な二重繰り返しに挑戦

二重繰り返しですが、内側の繰り返しの始まりの数を変えます。
予めリストd[0]からd[4]まで数を入れておきます。
・初めにd[0]からD[4]までの数を表示
・次にd[1]からd[4]までの数を表示
・・・・
・最期にd[4]の数を表示する。

次見て

```
[41, 23, 8, 15, 33]
41
23
8          i = 0の時
15         j = [0,1,2,3,4]で
33         繰り返し
23
8          i = 1の時
15         j = [1,2,3,4]で
33         繰り返し
8
15         i = 2の時
33         j = [2,3,4]で
           繰り返し
15         i = 3の時
33         j = [3,4]で繰り返し
33         i = 4の時
```

打ち込み:複雑な二重繰り返しに挑戦




```
d = [41, 23, 8, 15, 33]
print(d)
for i in range(5):

    for j in range(i,5):

        print(d[j])
```

そのまま打ち込む
(行は空けない)

部品08でrange()の使い方を確認。iとjの内容については前のスライド見る

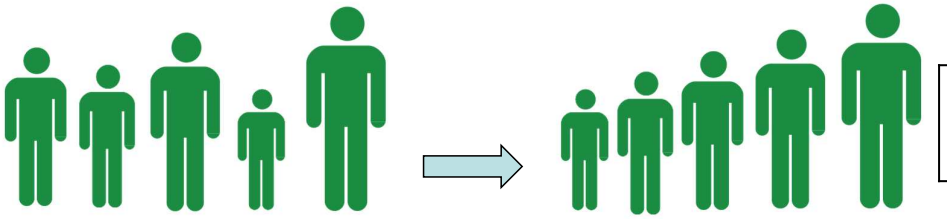
部品08

部品12

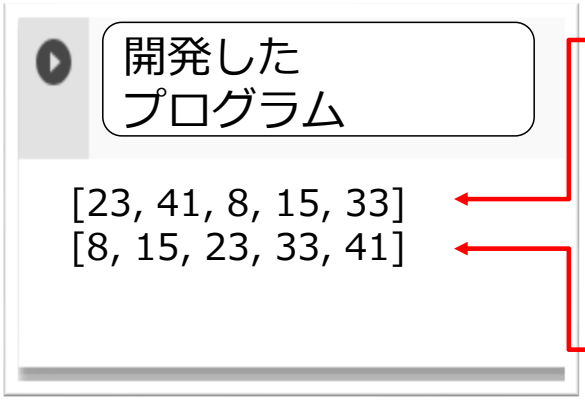
課題20

開発8:最後のチャレンジ: 数の並び替え

予めリストd[1]からd[4]まで数を入れておきます。この中の数を小さい順番に並び替えてください。



[41, 23, 8, 15, 33]
[8, 15, 23, 33, 41]



d = [41, 23, 8, 15, 33]
print(d)
初めのdの内容

print(d)
並べ方が終わった後の
dの内容

次見て

開発8:最後のチャレンジ: 数の並び替え

考え方: 「課題19:複雑な二重繰り返しに挑戦」
と「課題17:一番小さい数を配列の先頭に入れ替える」
を組み合わせてプログラムを作ります。

[41, 23, 58, 15, 33]
[15, 41, 58, 23, 33]
⇩
i = 0の時(課題17と同じ)
d[0]～d[4]の中で一番小さいものを
d[0]に入れ替える

[15, 41, 58, 23, 33]
[15, 23, 58, 41, 33]
⇩
i = 1の時
d[1]～d[4]の中で一番小さいものを
d[1]に入れ替える

[15, 23, 58, 41, 33]
[15, 23, 33, 58, 41]
⇩
i = 2の時
d[2]～d[4]の中で一番小さいものを
d[1]に入れ替える

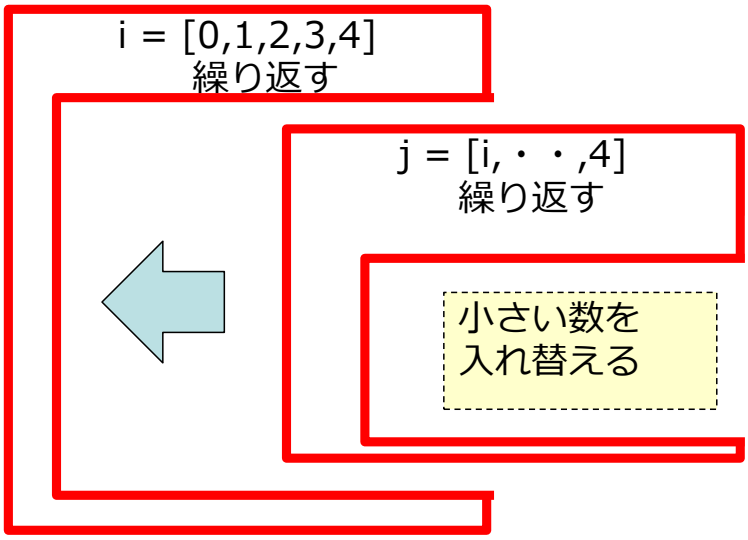
⇩

i = 4の時まで続ける

非常に重要

TPOINT

基本的な構造



課題17を
参考に

部品
12

今まで使っていた、変数iを使った繰り返しの中に、さらに繰り返しが入る形式
課題20の並び替えは、これが基本的な構造になります。

課題21　発展課題1: FizzBuss

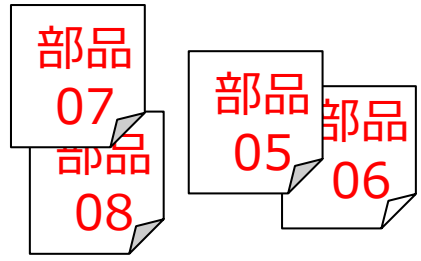
Fizz Buzzという遊びのプログラムを作ってみましょう。例えば、1から30までの数を表示しますが。

3の倍数の時は”Fizz”と表示。
5の場合の時は”Buzz”と表示。
3と5の倍数の時は、”Fizz Buzz”と表示。
それ以外は数字を言います。

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
```

```
16
17
Fizz
19
Buzz
Fizz
22
23
Fizz
Buzz
26
Fizz
28
29
FizzBuzz
```

ヒント



i % 3
余りを計算します。

課題22　発展課題2: バブルソート

今回作成したプログラムは選択ソートという方法を使ったもので、データを並び替える方法はいろいろあります。

次の並び替え方法の考え方やプログラムをWebで調べて作成してください。

バブルソート