

自由席

相方と隣り合って座ろう
PC41・PC42は使いません

発展的なプログラム1/2

23j1-209

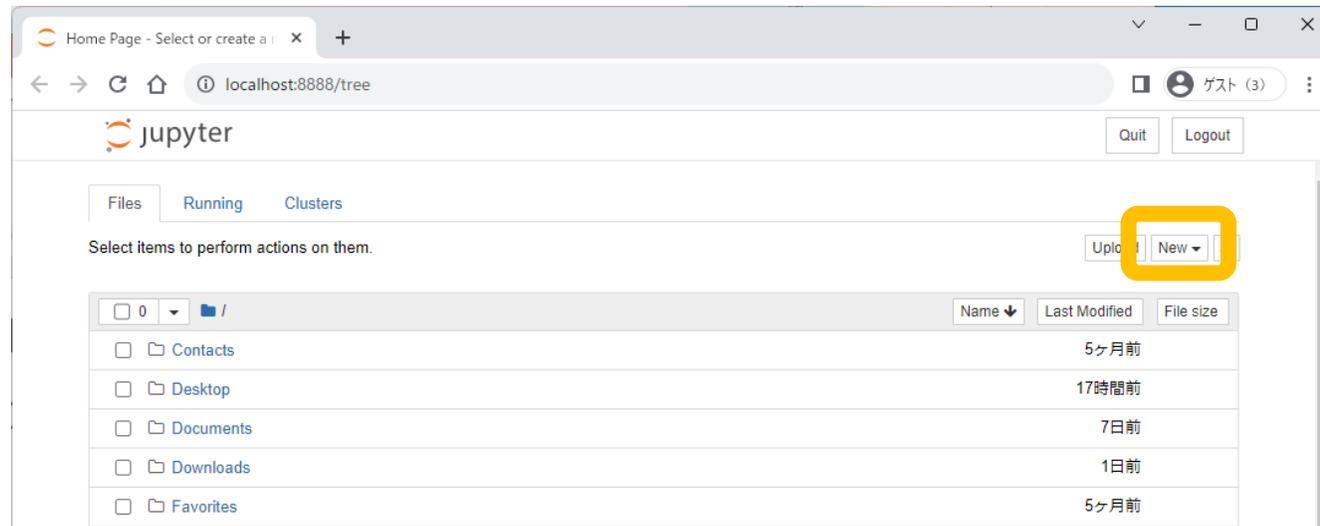
教科書P74-77

PCサインイン

Teamsサインイン

プログラミングを始めよう

- プログラムの開発環境を用意する
 1. [ スタート] → [Jupyter Notebooks]
 2. [New] → [Python3]



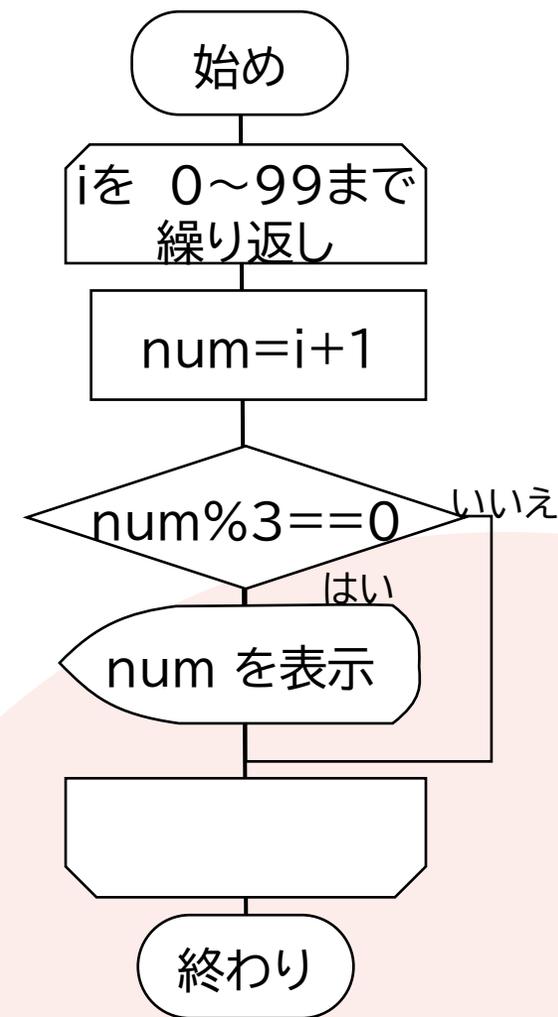
この時間の目標

- 発展的なプログラム
- S プログラム作成の工夫がよく理解でき、手順を工夫して活用しようと思った
- A プログラム作成の工夫がよく理解できた
- B プログラム作成の工夫が理解できた
- C プログラム作成の工夫が理解できなかった

制御構造の組み合わせ

- 分岐構造と反復構造の組み合わせ
 - 当然順次構造も組み合わせる

```
for i in range(100):  
    num=i+1  
    if num%3==0:  
        print(num)
```



リストと配列

- 複数の値を一つのまとまりとして扱う
 - Pythonではリストという

```
fruits=['apple', 'melon', 'orange', 'kiwi']
```

```
print(fruits) ①
```

```
print(fruits[3]) ②
```

```
fruits.append('banana')
```

```
print(fruits) ③
```

```
fruits[2]='mikan'
```

```
print(fruits) ④
```

① ['apple', 'melon', 'orange', 'kiwi']

② kiwi

③ ['apple', 'melon', 'orange', 'kiwi', 'banana']

④ ['apple', 'melon', 'mikan', 'kiwi', 'banana']

リストと配列

- 複数の値を一つのまとまりとして扱う
 - Pythonではリストという

```
name=['太郎', '次郎', '三郎']
```

```
kokugo=[80, 40, 92]
```

```
suugaku=[77, 98, 60]
```

```
eigo=[90, 82, 80]
```

```
for i in range(3):
```

```
    print(name[i], ' 国語 : ', kokugo[i], ' 点  数学 : ', suugaku[i], ' 点  英語 : ', eigo[i], ' 点')
```

太郎	国語 : 80 点	数学 : 77 点	英語 : 90 点
次郎	国語 : 40 点	数学 : 98 点	英語 : 82 点
三郎	国語 : 92 点	数学 : 60 点	英語 : 80 点

乱数

- 指定した範囲の数がランダムに発生する
 - シミュレーションやゲームなどに使用

```
#さいころ  
import random  
me=random.randint(1, 6)  
print(me)
```

乱数を使う時の
おまじない
プログラムの冒頭に
1回だけ書く

randomは特殊！
最後の数も入る

- Pythonの標準機能にはない
 - randomモジュールをimportして利用する

関数

- ひとつかたまりのプログラムを分割

```
import random
```

```
def sai():
```

```
    me=random.randint(1, 6)
```

```
    return me
```

さいころ関数

```
print(sai())
```

プログラム

関数とリスト

- 100回さいころを振ったときの出目の数

```
import random
```

```
def sai():  
    me=random.randint(1, 6)  
    return me
```

さいころ関数

```
A=[0, 0, 0, 0, 0, 0, 0, 0]  
for i in range(100):  
    me=sai()  
    A[me]=A[me]+1  
print(A)
```

プログラム

関数の活用

- 何が求まるのかな？

```
def factorial(n):  
    if n<2:  
        return 1  
    return n*factorial(n-1)
```

関数

```
n=int(input())  
print(factorial(n))
```

プログラム

プログラミングで使う技

フラグ

- 状態を表すのに使う変数のこと
 - 「フラグが立つ」と言ったりする

```
suu=int(input('Number?'))
```

```
sosu=True
```

```
for i in range(2, suu):
```

```
    if suu%i==0:
```

```
        sosu=False
```

```
if sosu==True:
```

```
    print('素数')
```

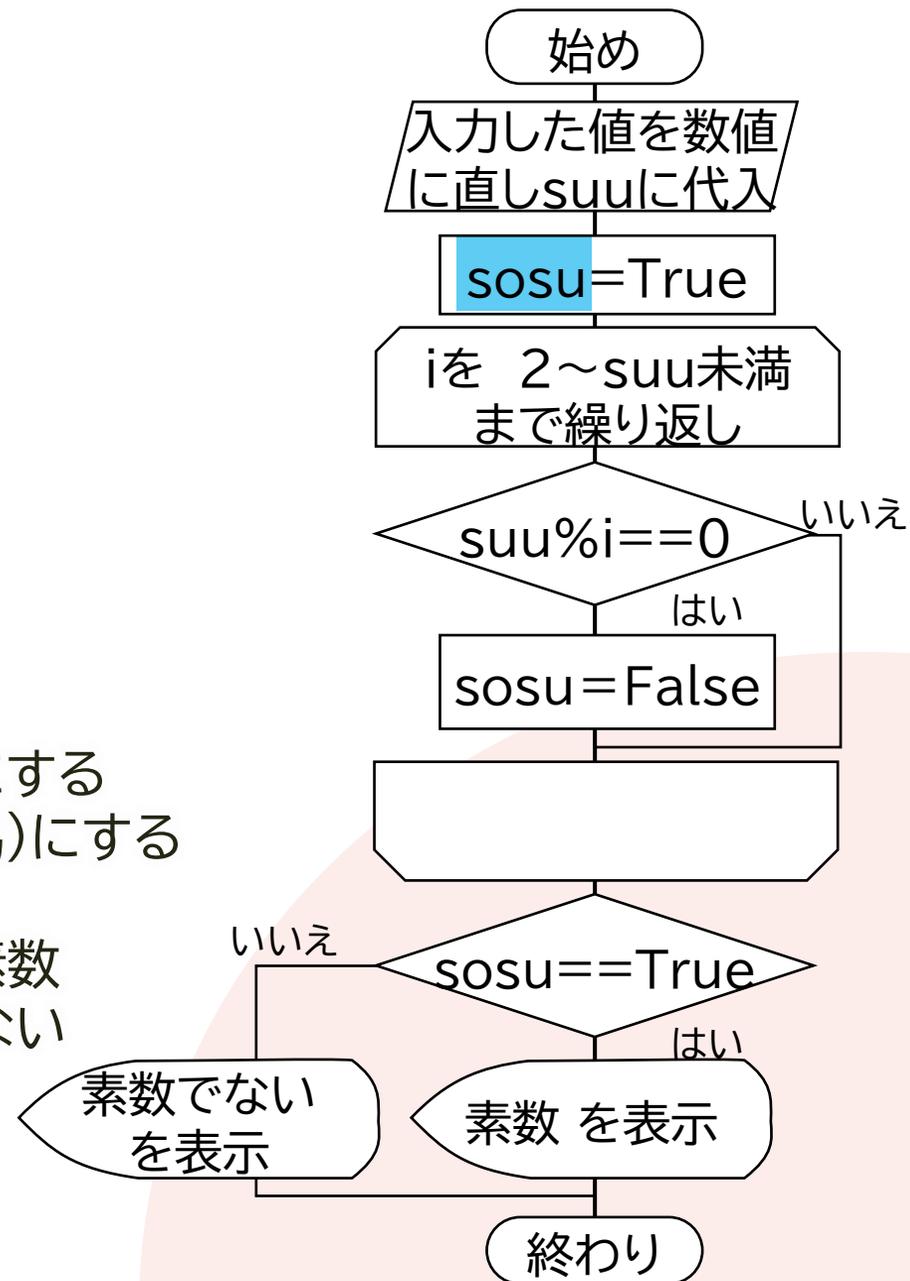
```
else:
```

```
    print('素数でない')
```

これがフラグ

とりあえず True(真)にする
違っていたら False(偽)にする

で、True のままなら素数
そうでなければ素数でない

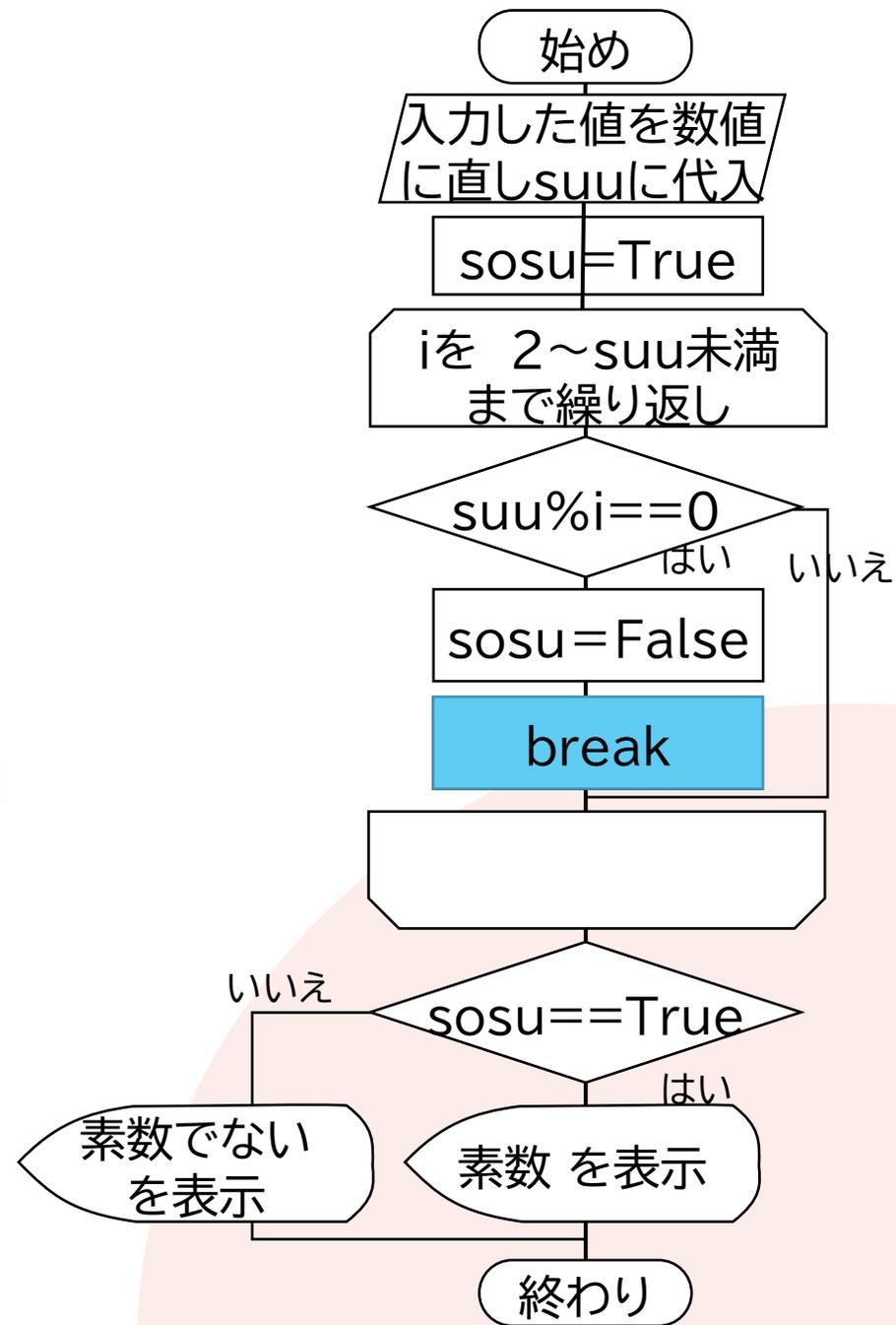


ループを抜ける

- ループを強制的に抜ける `break`
 - 結果がわかったら繰り返しを抜ける

```
suu=int(input('Number?'))
sosu=True
for i in range(2, suu):
    if suu%i==0:
        sosu=False
        break
if sosu==True:
    print('素数')
else:
    print('素数でない')
```

121212112を入力した時
breakあり 2秒
breakなし 15秒



カウンタ

- 数を数えるときに使う変数のこと
 - 1から100未満の数で
入力した数で割り切れる数の個数
(倍数が何個あるか?)

```
suu=int(input())
```

```
cnt=0
```

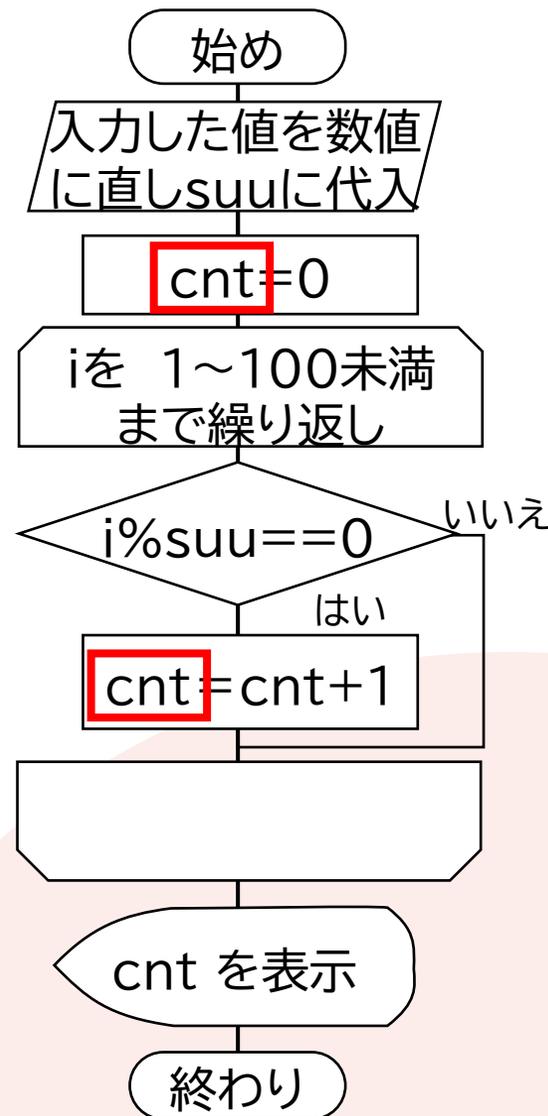
```
for i in range(1, 100):
```

```
    if i%suu==0:
```

```
        cnt=cnt+1
```

```
print(cnt)
```

とりあえずカウンタを0にして
1つ見つかったら1足す



Python プログラミング基礎テスト

Pythonプログラミング練習問題は何番まで終わったか確認してください
口外無用です

成績評価のためのテストです

1. 問題の条件に合うプログラムを作る
 - Jupyter Notebookを利用
2. 正しく動作することを確認する
 - 実行例があるときは、実行例どおりに動作するか確認
3. プログラムをフォームに貼り付ける
 - 正しく動かないものは減点です
4. 次の問題に取り組む

Pythonプログラミング 発展問題

Pythonプログラミング発展問題

- 以下の条件を満たすプログラムを自分で制作
 - 乱数を使ったゲーム的要素があるもの
 - 入力に対して反応があるもの
 - プログラムの説明ができること
 - 動作すること
- 何か一つ作り次々回授業で提出
 - ルールなども書き添える
 - 相互評価する
- 参考例
 - おみくじ/さいころ/数当て
 - 自分でルールや仕組みを考えても良い

自分で造る
が大前提

必須の関数
random・input・print
必須の構文
if・forかwhile

クイズ
計算問題
人工無能
など

おみくじ

- ランダムに大吉・吉・中吉・小吉・末吉・凶・大凶 がでる
- ランダムにメッセージが添えられる

- あまり発展しないかも

サイコロ

- サイコロの出た目を当てるゲーム
- 出た目を当てるゲーム-当たるまでバージョン
- サイコロを2個振ってゼロ目ができるまでの回数を表示
- さらに.....

数当て

- 10までの乱数を発生し、その数を当てるゲーム
- 10までの数を当てるゲーム-当たるまでバージョン
- 10までの数を当てる-ハズレなら大きく/小さくのヒント
- さらに.....

Pythonプログラミング発展問題

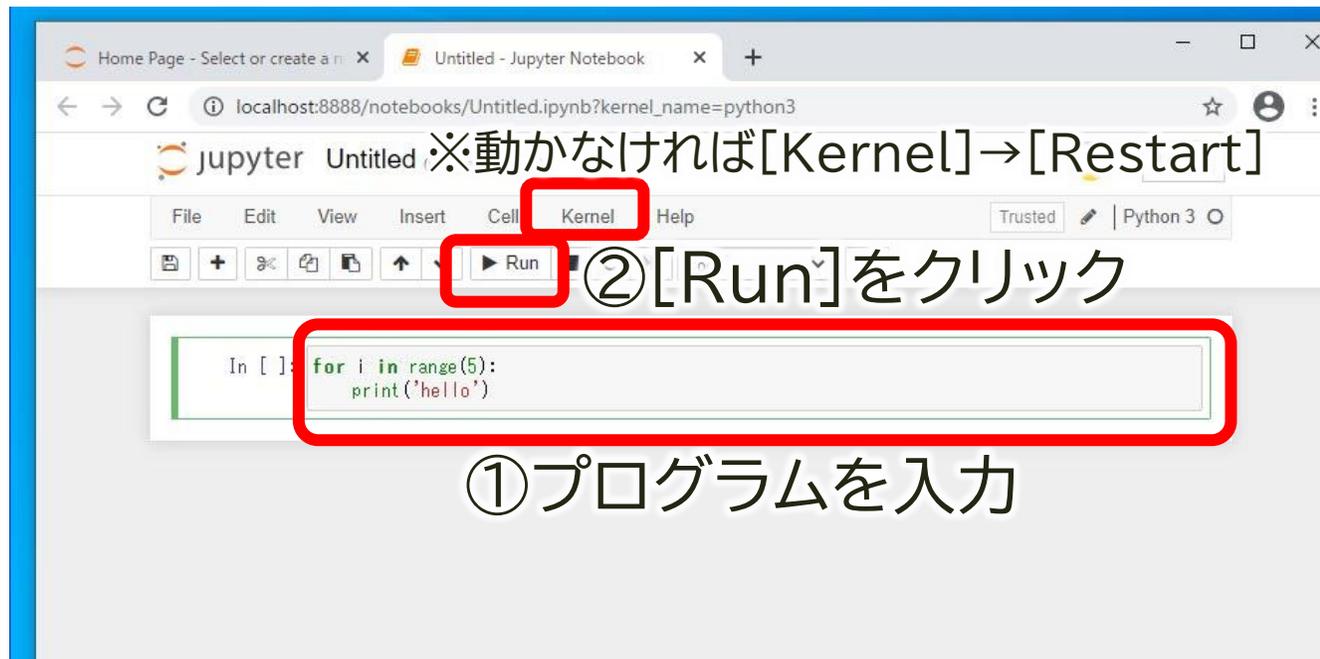
- 以下の条件を満たすプログラムを自分で制作
 - 乱数を使ったゲーム的要素があるもの
 - 入力に対して反応があるもの
 - プログラムの説明ができること
 - 動作すること
- 何か一つ作り次々回授業内で提出
 - ルールなども書き添える
 - 相互評価する
- 参考例
 - おみくじ/さいころ/数当て
 - 自分でルールや仕組みを考えても良い

自分で造る
が大前提

必須の関数
random・input・print
必須の構文
if・forかwhile

Jupyter Notebookを利用している人

- ①プログラムを入力
- ②[Run]をクリック
 - ※動かなければ
[Kernel]→[Restart]



- プログラムの保存
 - プログラムをコピー
 - Wordに貼付→OneDrive/Classiポートフォリオに貼付

Jupyter Notebookでプログラムの保存

- 保存するには
 - [File]→[Download as]→[Notebook(.ipynb)]
 - ファイル名を半角小文字英数でつける
 - 自分のドライブに保存
 - OneDriveなどにアップロード
- 続きからやるには
 - Jupyterの画面右上[Upload]
 - Notebookを開く前の画面
 - Googleコラボラトリーは[ファイル]→[ノートブックをアップロード]
 - ファイルを指定して[開く]
 - [Upload]をクリック

終了

授業の振り返り

- Classiアンケートに回答
[23情報 I 2学期 第9回 授業の振り返り
 - 授業実施日に送信
- パソコンはシャットダウン
- 次回は「27発展的なプログラム1/28発展的なプログラム2」
 - 教科書P74～77