1 基礎

1.1 知識

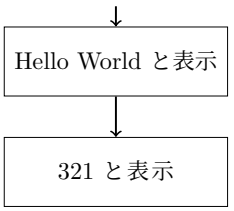
(1) 表示・入力・代入

print( ) で, () の中身を表示することができる. また, 文字列の場合は, ' ' でくくる.

また, 人が入力した文字を読み取る場合は, input() を使う.

プログラミングにおいて, 変数を設定し数を代入して, 演算処理を行うことができる. = を用いることで, 数値代入をすることができる.

```
1 print('Hello World')
2 print(321)
```

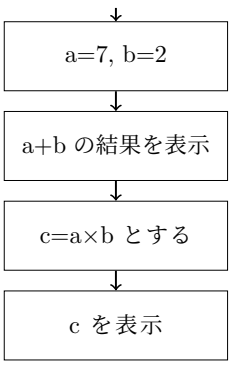


(2) 演算

演算子は以下の通り. 例は, $a = 5, b = 2$ とする.

	演算子	記入例	結果
足し算	$+$	$a + b$	7
引き算	$-$	$a - b$	3
掛け算	$*$	$a * b$	10
割り算	$/$	a/b	2.5
商	$//$	$a//b$	2
余り	$\%$	$a\%b$	1
累乗	$**$	$a**b$	25

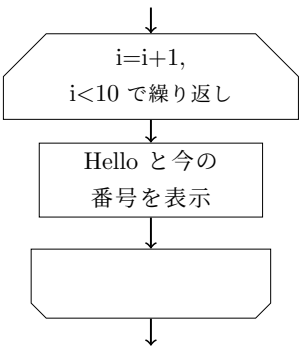
```
1 a=7;b=2
2 print(a+b)
3 c=a*b
4 print(c)
```



(3) 繰り返し

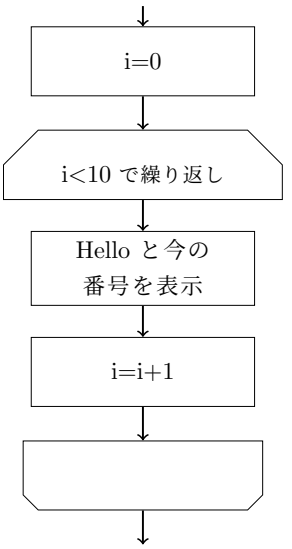
同じ操作を繰り返す際に用いる. for 文, while 文などがある.

```
1 for i in range(10):
2  print('Hello', i)
```



1 行目の意味は, 「変数 i を 0 から 10 未満の間で, 1 ずつ増やしながら繰り返す」
2 行目の意味は, 「Hello」という文字列と, 今の数字を表す「i」を出力.
繰り返すコードは, 1 マス開けて書くということに注意.

```
1 i=0
2 while i<10:
3  print('Hello', i)
4  i=i+1
```



1 行目で i の初期値設定, 4 行目 (while 文内) で i の処理が必要.
2 行目の意味は, 「変数 i が 10 未満であれば繰り返す」
3 行目の意味は, 「Hello」という文字列と, 今の数字を表す「i」を出力.
繰り返すコードは, 1 マス開けて書くということに注意.

(4) 条件分岐

条件によって実行内容を変えるときに利用.

1 行目は, 変数 a に入力された数字を代入. 入力された文字・数字は全て「文字列」として扱われるため, 「整数」に変換する必要がある. その関数として int を用いている.

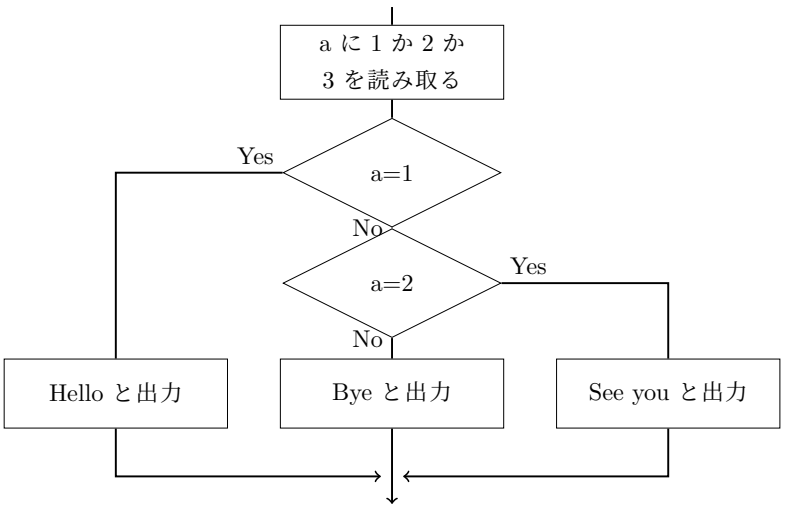
2 行目は, もし a が 1 だったら...

4 行目は, そうでなくもし a が 2 だったら...

6 行目は, そうでなければ...

それぞれの条件下で実行するプログラムについては, 字下げを行う.

```
1 a=int(input('INPUT 1 or 2 or 3: '))
2 if a==1:
3  print('Hello')
4 elif a==2:
5  print('See you')
6 else:
7  print('Bye')
```

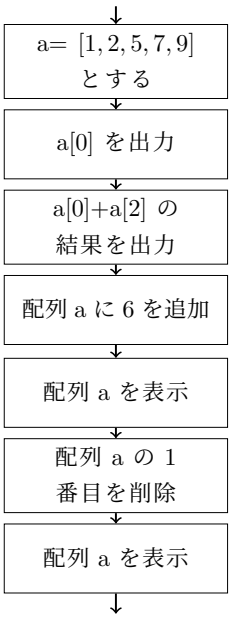


(5) リスト

リストを用いることで, いくつかの数字をまとめて扱うことができる.

注) リストの先頭は 0 番目.

```
1 a=[1, 2, 5, 7, 9]
2 print(a[0])
3 print(a[0]+a[3])
4 a.append(6)
5 print(a)
6 a.pop(1)
7 print(a)
```



1 行目でリストの定義, 2-3 行目はリストの要素の扱い方である.

4 行目で, リストの最後に 6 を追加.

6 行目で, リストの 1 番目の数字を削除.

1.2 練習問題

(1) 自分の名前を出力させよ. (in English)

(2) a, b に好きな数字を代入し, その 2 数に対して全ての四則演算を実行し, 結果を表示させよ.

(3) 0 から 10 回繰り返し, Hello と 今の番号の 2 倍の数を表示させよ.
つまり, 以下のような結果となってほしい.

```
Hello, 0
Hello, 2
Hello, 4
      ( 中略)
Hello, 16
Hello, 18
```

(4) 1 ～ 4 のいずれかを実行後に入力し, 読み取った数字によって違う英単語を出力させるプログラムを作れ.

(5) 自由にプログラムを組み, いろいろと試してみる.

2　**平均・分散** etc

2.1 知識

x	25	21	18	17	21	26	23	21	20	18
y	28	19	30	13	27	12	30	13	15	23

x, y について, 以下の問いに答えよ.

(1) 合計を求めよ.

(2) 平均を求めよ.

(3) 分散を求めよ.

(4) 相関係数を求めよ.

2.2 練習用コード

```
import math
X=[25 , 21 , 18 , 17 , 21 , 26 , 23 , 21 , 20 , 18]
Y=[28 , 19 , 30 , 13 , 27 , 12 , 30 , 13 , 15 , 23]
sum_x=0
sum_y=0
for i in range(len(X)):
 sum_x=sum_x+X[i]
 sum_y=sum_y+Y[i]
print("sum of X = ", sum_x)
print("sum of Y = ", sum_y)
ave_x=sum_x/len(X)
ave_y=sum_y/len(Y)
print("average of X = ", ave_x)
print("average of Y = ", ave_y)

sum_xx=0
sum_yy=0
for i in range(len(X)):
 sum_xx=sum_xx+(X[i]-ave_x)*(X[i]-ave_x)
 sum_yy=sum_yy+(Y[i]-ave_y)*(Y[i]-ave_y)
var_x=sum_xx/len(X)
var_y=sum_yy/len(Y)
print("variance of X = ", var_x)
print("variance of Y = ", var_y)
stdev_x=math.sqrt(var_x)
stdev_y=math.sqrt(var_y)

sum_xy=0
for i in range(len(X)):
 sum_xy=sum_xy+(X[i]-ave_x)*(Y[i]-ave_y)
cov_xy=sum_xy/len(X)
r=cov_xy/(stdev_x*stdev_y)
print("correlation coefficient = ", r)
```

2.3 実行結果

上のコードの実行結果はどのようになったか.

2.4 構造

左のコードの構造を分解して理解しよう.

2.5 新しく得た知識等

忘れないためにも, メモ.

3　ジャンケンで学ぶ乱数

3.1 知識

乱数を使うためには, コマンドが必要である. 以下の 1 行目のように書くことで, ランダム関数を使うことができる. この記載がないと, エラーとなる.

```
1 import random
2 a=random.randint(3, 6)
3 print(a)
4 b=random.randrange(8)
5 print(b)
```

1 行目は, 乱数機能を追加するコマンド.
2 行目は, 3 以上 6 以下のランダムな整数値を与える関数.
4 行目は, 0 以上 8 未満のランダムな整数値を与える関数.
与えられる乱数値は, 実行のたびに変化する.

3.2 いろいろな練習

(1) 4 以上 10 以下のランダムな整数値を出力してみよう.

(2) 0 以上 100 以下のランダムな整数値を出力してみよう.

(ネット検索も用いて OK. どのようにしたらいいかはメモしておこう.)

(3) 0 以上 1 以下のランダムな小数値を出力してみよう.

(4) 0 以上 10 以下のランダムな小数値を出力してみよう.

3.3 練習用コード

```
import random
print("Your hand? 1:Rock, 2:Scissors, 3:Paper")
user_hand=int(input("INPUT 1 or 2 or 3 : ", ))
cpu_hand=random.randint(1,3)
if cpu_hand==1:
 print("CPU is Rock")
 if user_hand==1:
  print("draw")
 elif user_hand==2:
  print("you lose")
 else:
  print("you win")
elif cpu_hand==2:
 print("CPU is Scissors")
 if user_hand==2:
  print("draw")
 elif user_hand==3:
  print("you lose")
 else:
  print("you win")
else:
 print("CPU is Paper")
 if user_hand==3:
  print("draw")
 elif user_hand==1:
  print("you lose")
 else:
  print("you win")
```

3.4 実行結果

上のコードの実行結果はどのようになったか.

3.5 構造

左のコードの構造を分解して理解しよう.

3.6 新しく得た知識等

忘れないためにも, メモ.

4　関数

4.1 知識

同じ機能を持つプログラムを何度も書くと, コードが膨大になり読みにくい. それを避けるために, 関数を作成し整理することができる.

```
1 def circle(r):
2  return r**2*3.14
3
4 r=int(input('radius='))
5 area=circle(r)
6 print("AREA=", area)
```

数学の関数同じように, $f(x)$ の形で表す.
f を関数名, x を引数, x を入れると返ってくる値を返り値 (戻り値) という. 上の例では, circle が関数名, r が引数, $3.14 \times r^2$ が返り値である.

4.2 その他関数

- 組み込み関数
  あらかじめ用意されている関数のこと. python には, 豊富に用意されている.
- API
  プログラムからソフトウェアを操作するためのインタフェース.
- WebAPI
  外部から呼び出して利用できる API のこと. 必要なデータを引数として送ることで, 返り値としてサービス・データが利用できる.

4.3 練習問題

(1) 長さを入力したら, その長さが 1 辺となる立方体の体積を返す関数を作る.

(2) 長さを入力したら, その長さが半径の球の体積を返す関数を作る.

(3) 3 辺の長さを入力して, その 3 辺の直方体の体積を返す関数を作る. 【Try !】

5　健二くんと学ぶモジュロ演算

5.1 知識

サマーウォーズにて, 健二が生年月日から曜日を暗算で求めていた. 彼は「モジュロ演算」をしただけである. このモジュロ演算を実際に学び, コードを組んでみよう.

(1) ツェラーの方法 (簡略化 ver.)

(a) 1 月, 2 月の場合, その前の年の 13 月, 14 月として扱う.

(b) $A =$ 西暦の下 2 桁 とする.

(c) $B =$ 西暦の下 2 桁を 4 で割った商

(d) 何月かにより, 以下の表に対応する値を C とする.

3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月	12 月	13 月	14 月
3	6	1	4	6	2	5	0	3	5	1	4

(e) $D =$ 日付 とする.

(f) $X = A + B + C + D$ を求める. もし, 1900 年台の場合は 1 を足す.

(g) X を 7 で割ったあまりを求める. 以下の表に対応して曜日を求めることができる.

余り	1	2	3	4	5	6	0
曜日	日	月	火	水	木	金	土

(2) あなたが生まれたのは何曜日か計算してみよう.

5.2 練習用コード

```
year=int(input("Year= "))
month=int(input("Month= "))
day=int(input("Day= "))
if month==1:
 newyear=year-1
 month=13
elif month==2:
 newyear=year-1
 month=14
else:
 newyear=year
a=newyear%100
b=(newyear%100)//4
c_list=[3,6,1,4,6,2,5,0,3,5,1,4]
c=c_list[month-3]
d=day
x=a+b+c+d
if (newyear//100)==19:
 x=x+1
w=x%7
week=["Sataday","Sunday","Monday","Tuesday","Wednesday","Thursday","Friday"]
print(day,"/",month,"/",year,"is...")
print(week[w])
```

5.3 実行結果

上のコードの実行結果はどのようになったか.

5.4 構造

左のコードの構造を分解して理解しよう.

5.5 新しく得た知識等

忘れないためにも, メモ.

6　31 ゲーム

6.1　知識

ルール

(1) 2 人で, 1 から 31 までの数字を順番に言って, 最後の 31 を言ったら負け.

(2) 一度に一人 3 つまでの数字を言うことができる.

例えば..

A さん「1, 2, 3」
B さん「4, 5」
A さん「6」

⋮

A さん「29, 30」
B さん「31」

の場合, B さんの負け.

6.1.1　実際にやってみる

対戦相手を探してやってみよう. (教員でも可)

6.1.2 **実は**...

必勝法がある．検討してみよう．

6.2 練習用コード

```
import random
print("31-game start")
print("You first")
i=0
while i<31:
 print("How many to increase ?")
 n=int(input(("1 or 2 or 3 : ")))
 i=i+n
 print("Now...", i)
 if i>=31:
  print("You lose")
 else:
  n=random.randint(1,3)
  print("CPU call...",n)
  i=i+n
  print("Now...", i)
  if i>=31:
   print("You win")
```

6.3 実行結果

上のコードの実行結果はどのようになったか.

6.4 構造

左のコードの構造を分解して理解しよう.

6.5 新しく得た知識等

忘れないためにも, メモ.

6.6 追加課題

取り組む順は自由.

(1) CPU の強さを変えたい (現状より強く).

(2) 先攻後攻の順をジャンケンで決定できるようにしたい.