情報 I

4-1　　Python作法1

4-1-11　配列

(11)配列　複数の値を一括で扱うことができる番号付きの値収納箱

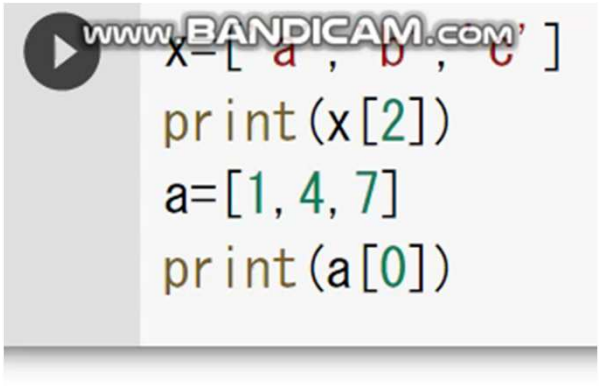
(11-1)　基本　　配列名=［値1,値2,値3,　･･･］

```
x=['a','b','c']                    # 配列x
print(x[2])                        #

a=[1,4,7]                          # 配列a
print(a[0])                        #
```

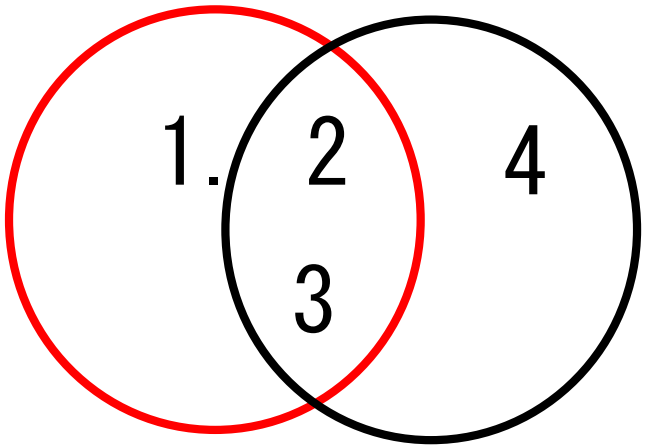
行＼列	X[0]	X[1]	X[2]
list	a	b	c

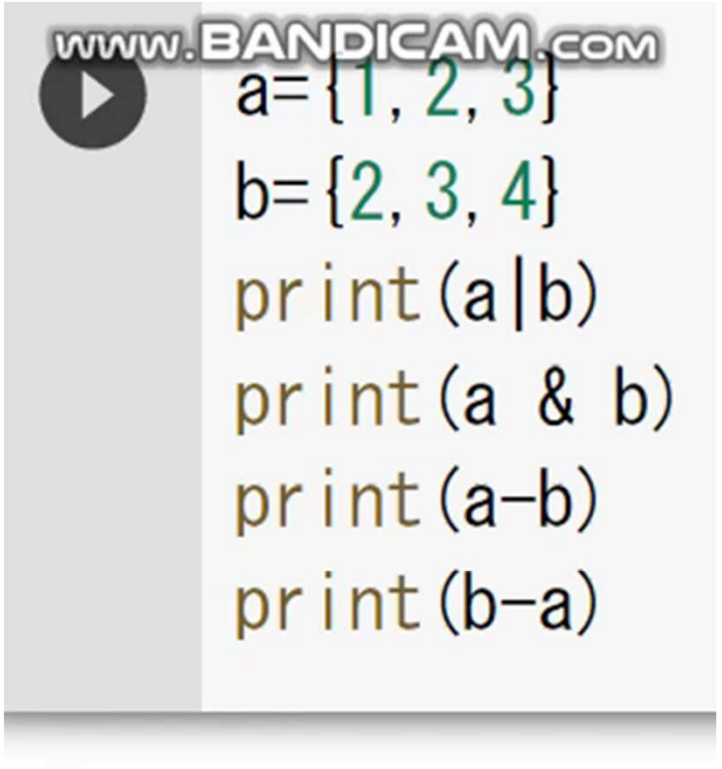
行＼列	a[0]	a[1]	a[2]
list	1	4	7



(11-2)　配列集合

```
a={1,2,3}                # 配列
b={2,3,4}                # 配列
print(a|b)               # 和集合
print(a&b)               # 積集合
print(a-b)               # 差集合
print(b-a)               # 差集合
```





(11-3)　リスト　list

```
list_x=[2,5,8]     # 整数型配列
print(list_x)      # 配列ｘを表示
list_y=[1,1.5,3, ‘love’ ]
                   # 整数型、小数型、文字型混合配列
print(list_y)      # 配列ｙを表示
list_y[0]=0.6      # 配列yの「0番にデータ0.6に変更」
print(list_y)      # 配列ｙを表示
list_y.insert(2,’yes’)
                   # 配列yの「2番にデータ’yes’を挿入」
print(list_y)      # 配列ｙを表示
```

コードを上から順番に処理している･･･順次処理

```
list_x=[2,5,8]
print(list_x)

list_y=[1,1.5,3,'love']
print(list_y)

list_y[0]=0.6
print(list_y)

list_y.insert(2,'yes')
print(list_y)
```

(11-5)　ソート（並べ替え）

```
list = [4,5,2,1,3]          # 配列list
list.sort()                 # listを並べ替え（デフォルト：昇順）
print(list)                 #  表示する
list.sort(reverse=True)     # listを並べ替え（降順）
print(list)                 # 表示する
```

(11-6)　サーチ（探索）

```
list = [4,5,2,1,3]
result = 6 in list      # 配列に6があるかの結果
print(result)           # 結果表示

list = [4,5,2,1,3]
result = 5 in list      # 配列に5があるかの結果
print(result)           # 結果表示

list = [4,5,2,1,3,5,3,5,1]
cnt = list.count(5)     # 配列にある5の個数
print(cnt)

index = list.index(3)   # 配列にある3の場所
print(index)
```



```
list = [4,5,2,1,3]
result = 6 in list
print(result)

list = [4,5,2,1,3]
result = 5 in list
print(result)

list = [4,5,2,1,3,5,3,5,1]
cnt = list.count(5)
print(cnt)

index = list.index(3)
print(index)
```

```
list = [4,5,2,1,3,5,3,5,1]

a=max(list)    # 配列の最大値
print(a)

b=min(list)    # 配列の最小値
print(b)

c=sum(list)    # 配列の合計値
Print(c)

d=len(list)    # データの個数
print(d)
```


```
list = [4,5,2,1,3,5,3,5,1]

a=max(list)
print(a)

b=min(list)
print(b)

c=sum(list)
print(c)

d=len(list)
print(d)
```

(11-7)　文字列サーチ（部分探索）➡for文：シーケンスを反復処理

①リスト内包表記：配列から、サーチにより新しいリストを作成する方法

```
list = ['albert', 'carol', 'ester','jane','peter',
'george','william','Catherine','john','richard','Diana']
match = [a for a in list if "er" in a]
print(match)
```

```
list = ['albert', 'carol', 'ester','jane','peter', 'george','william','Catherine','john','richard','Diana']
match = [a for a in list if "er" in a]
print(match)
```

```
['albert', 'ester', 'peter', 'Catherine']
```

② if文を forループ内で使用➡配列内の er を含む文字列を検索

```
list = ['albert', 'carol', 'ester','jane','peter',
'george','william','Catherine','john','richard','Diana']
new_list =[]
for x in list:
    if "er" in x:
        new_list.append(x)
print(new_list)
```

```
list = ['albert', 'carol', 'ester','jane','peter', 'george','william','Catherine','john','richard','Diana']
new_list =[]
for x in list:
  if "er" in x:
    new_list.append(x)
print(new_list)
```

```
['albert', 'ester', 'peter', 'Catherine']
```

(11-9)　二次元配列


```
list_123 = [
    [1, 2, 3, 4, 5, 6, 7, 8, 9],
    [10, 20, 30, 40, 50, 60, 70, 80, 90],
    [100, 200, 300, 400, 500, 600, 700, 800, 900]
    ]
print(list_123)
print(list_123[1][5])
print(list_123[2][7])
```

```
print(list_123)
print(list_123[1][5])
print(list_123[2][7])
```

行＼列	Col[0]	Col[1]	Col[2]	Col[3]	Col[4]	Col[5]	Col[6]	Col[7]	Col[8]
Row[0]	1	2	3	4	5	6	7	8	9
Row[1]	10	20	30	40	50	60	70	80	90
Row[2]	100	200	300	400	500	600	700	800	900

(11-10) モデル化とシミュレーション(乱数、配列による待ち時間問題)

	A	B	C	D	E	F	G	H	I
1	課題	待ち時間問題：到着時間間隔を1~5分(乱数)、サービス作業時間を５~10分(乱数)とする。							
2	条件	受付窓口１か所の場合							
3	客到着順	到着間隔	到着時刻	開始時刻	作業時間	終了時刻	待ち時間		
4	1	0	0	0	7	7	0		
5	2	2	2	7	10	17	5		
6	3	4	6	17	6	23	11		
7	4	3	9	23	5	28	14		
8	5	1	10	28	10	38	18		
9	6	1	11	38	6	44	27		
10	7	5	16	44	10	54	28		
11	8	3	19	54	10	64	35		
12	9	2	21	64	7	71	43		
13	10	5	26	71	6	77	45		
14	11	5	31	77	9	86	46		
15	12	4	35	86	6	92	51		
16	13	3	38	92	7	99	54		
17	14	3	41	99	6	105	58		
18	15	1	42	105	8	113	63		
19	16	2	44	113	6	119	69		
20						平均待ち	35.4		

筆者・当サイト表計算によるモデル化参照
http://strnun.fool.jp/pov-ray_strnun/Waiting%20time%20problem_ans.xlsx

・到着時刻 = 前客到着時刻 + 間隔

・到着時刻>=前客終了時刻のとき　開始時刻 =到着時刻
・到着時刻 < 前客終了時刻のとき　開始時刻 = 前客終了時刻

・終了時刻 = 開始時刻 + 作業時間

・待ち時間 = 開始時刻 -到着時刻

番目　時刻	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1	1	2	3	4	5	6	7																					
2			1	2	3	4	5	1	2	3	4	5	6	7	8	9	10											
3							1	2	3	4	5	6	7	8	9	10	11	1	2	3	4	5	6					
4										1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5
										4番目待ち時間														4番目作業時間				

② コーディング実装

```
import statistics                  # 統計用モジュール
import random                      # 乱数モジュール
ar=0                               # 到着時刻arrive
st=0                               # レジ会計・治療など作業開始時刻start
fin=0                              # レジ会計・治療など作業終了時刻finish
se=random.randint(5,10)            # レジ会計・治療など作業経過時間service
wt=[]                              # 到着から開始までの待ち時間配列を設定waiting time
for i in range(20):                # 20人分反復
  ar=ar+random.randint(1,5)        # 到着時刻は先着者の後1～5分後で乱数
  if fin<=ar:                      # 到着時刻が先客の終了時刻より後ならば
    st=ar                          # 開始時刻は到着時刻
  else:                            # そうでないなら
    st=fin                         # 開始時刻は先客の終了時刻
  fin=st+se                        # 終了時刻は開始時刻に作業時間を加える
  wt.append(st-ar)                 # 待ち時間配列に(開始時刻-到着時刻)を追加
  print(i+1,ar,wt[i],st,fin)       # 1～20番、到着時刻、待ち時間、作業開始時刻、終了時刻を表示
  i=i+1                            # 次番を処理
print(average(wt))                 # 待ち時間の平均値を表示
```

```
import statistics
import random
ar=0
st=0
fin=0
se=random.randint(5,10)
wt=[]
for i in range(20):
  ar=ar+random.randint(1,5)
  if fin<=ar:
    st=ar
  else:
    st=fin
  fin=st+se
  wt.append(st-ar)
  print(i+1,ar,wt[i],st,fin)
  i=i+1
print(average(wt))
```

1～20番、到着時刻、待ち時間、作業開始時刻、終了時刻

```
1 5 0 5 12
2 8 4 12 19
3 10 9 19 26
4 11 15 26 33
5 14 19 33 40
6 15 25 40 47
7 18 29 47 54
8 21 33 54 61
9 26 35 61 68
10 30 38 68 75
11 35 40 75 82
12 39 43 82 89
13 40 49 89 96
14 41 55 96 103
15 45 58 103 110
16 48 62 110 117
17 49 68 117 124
18 54 70 124 131
19 55 76 131 138
20 57 81 138 145
40 45
```