守護神でもできるプログラミング講座

No.7

D3 村尾響

今日の目標

- リストの便利な宣言の仕方を覚えよう
- 多次元リストを使えるようになろう

OUTLINE

1. 復習
2. リストの宣言
3. 2次元リスト
4. リストの演算

1. 復習

リストを作成するには

数値や文字列を , で区切って[ ]で囲む

a = [1, 2, 3, 4]

とすると1や2や3や4といった数値(データ)がまとまって入ったリスト変数aが作成できる

1. 復習

x = [1, 2, 3, 4]
とすると
1が 0 番目　　2が 1 番目
3が 2 番目　　4が 3 番目
のデータになる

0から始まることに注意！

x[i]　　(この場合のiは0～3)とすると
xの中のi番目のデータを得ることができる

1. 復習

x = [1, 2, 3, 4]

print(x)
とすると
[1, 2, 3, 4]
のように
xの中のデータ全てが
表示される

print(x[2])
とすると
3
のように
xの中の2番目のデータ
(3)が表示される

2. リストの宣言

a = [1, 2, 3, 4]

としてリスト変数を宣言する場合、100個ほどのデータをリストにする場合は大変

数は気にせずにリストのデータの個数を指定したい場合、*****や**for**文を用いてリストを宣言することもある

2. リストの宣言

データの値は気にせず、データの数が10個のリスト変数を宣言したい場合、

list1 = [0] * 10

list2 = [0 for i in range(10)]

と書くことができる

この場合、どちらもすべてのデータの値を0にしている(0でなくても全く問題ない)

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

2. リストの宣言

同じように値は気にしないデータの数がn個のリスト変数を宣言したい場合は以下のようになる

list1 = [ANYNUM] * n

list2 = [ANYNUM for i in range(n)]

ANYNUMには好きな数字や文字列を、nには好きな自然数を入れる

2. リストの宣言

*やfor文を用いたリストの宣言は、データの数(要素数)の多いリストの宣言に向いているが、

自由にデータ(要素)の値を決めることができないという欠点がある

リスト変数をどう宣言するかは場合によって考えよう

演習1 リストの宣言

- 以下のようにリスト変数を宣言した時、リストの中身がどうなるか確認してみよう

[i for i in range(7)]

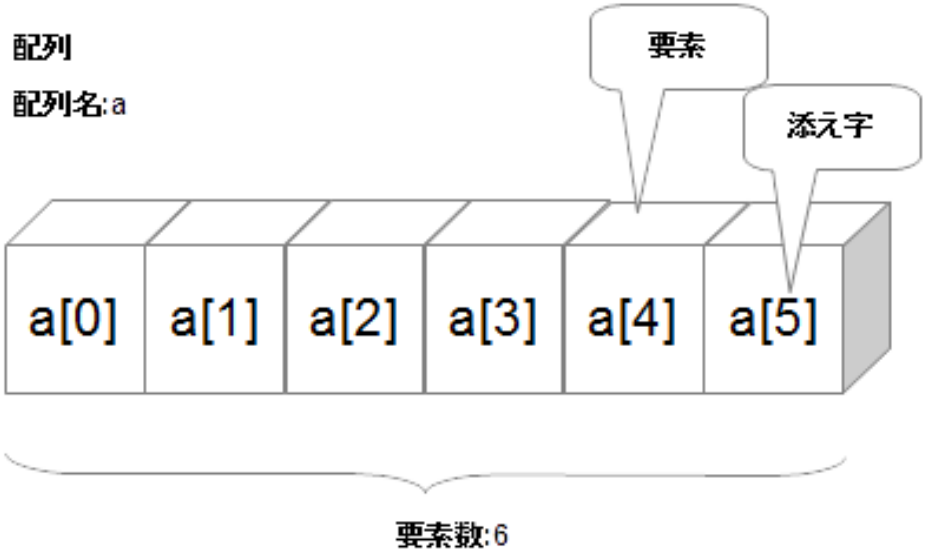
- for文によるリスト変数宣言を用いてリスト変数Li = [-1, -2, -3, -4, -5]を作成しよう

3. 2次元リスト

1次元リスト(今までのリスト)は図にすると以下のような図のイメージになる

いくつかのデータを1列にまとめてるイメージ

(配列は頭でリストに
置換してください)



3. 2次元リスト

2次元リスト1次元のリスト(横に1列になっているもの)をさらに縦に並べたものというイメージになり、九九の表のような配置になっている

array[0][0]	array[0][1]	array[0][2]	array[0][3]	array[0][4]
array[1][0]	array[1][1]	array[1][2]	array[1][3]	array[1][4]
array[2][0]	array[2][1]	array[2][2]	array[2][3]	array[2][4]

3. 2次元リスト

下のような2次元リストを宣言するには以下のようになる(値を指定したい場合)

array = [[0,1,2,3,4], [5,6,7,8,9], [10,11,12,13,14]]

array[0][0]	array[0][1]	array[0][2]	array[0][3]	array[0][4]
array[1][0]	array[1][1]	array[1][2]	array[1][3]	array[1][4]
array[2][0]	array[2][1]	array[2][2]	array[2][3]	array[2][4]

3. 2次元リスト

array = [[0,1,2,3,4], [5,6,7,8,9], [10,11,12,13,14]]
とすると

0,1,2,3,4というデータが入っている**array[0]**

5,6,7,8,9というデータが入っている**array[1]**

10,11,12,13,14というデータが入っている**array[2]**

というリスト変数が作成され、さらに

array[0],array[1],array[2]というデータが
入っている**array**というリスト変数が作られる

3. 2次元リスト

array = [[0,1,2,3,4], [5,6,7,8,9], [10,11,12,13,14]] のとき

array[0] = [0, 1, 2, 3, 4] arrayの0番目のリスト

array[2] = [10, 11, 12, 13, 14] arrayの2番目のリスト

array[0][0] = 0 array[0]の0番目

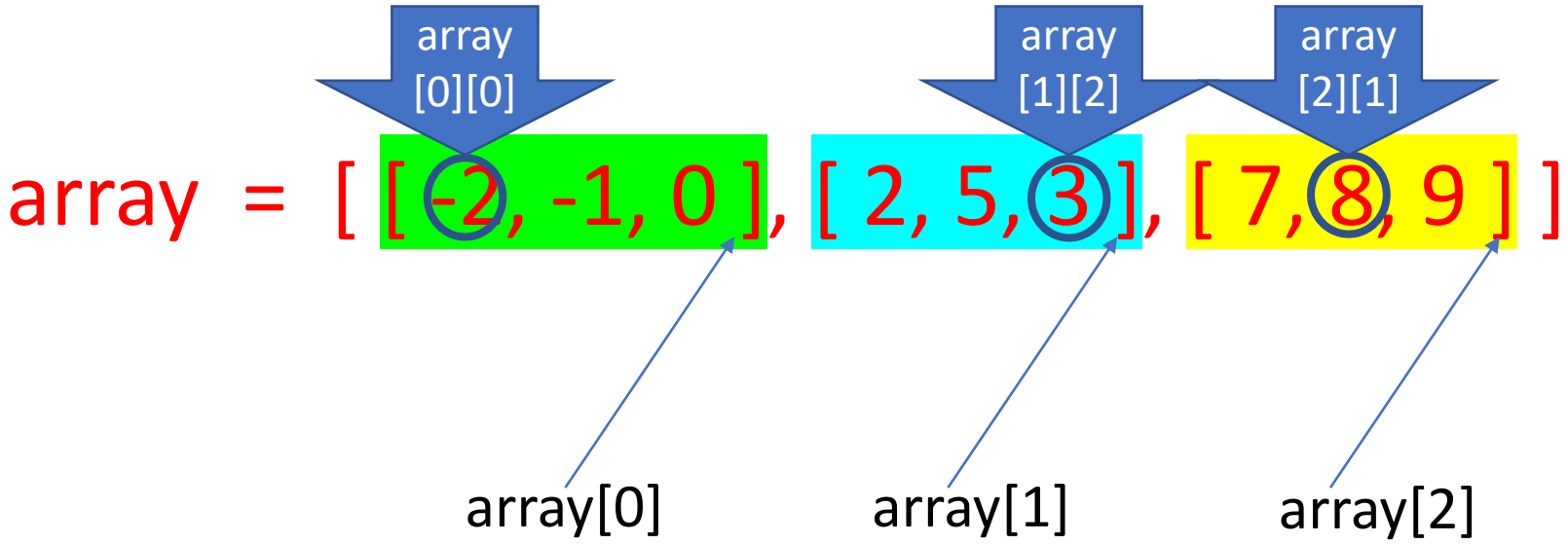
array[0][2] = 2 array[0]の2番目

array[1][4] = 9 array[1]の4番目

array[2][1] = 11 array[2]の1番目

array[2][3] = 13 array[2]の3番目

3. 2次元リスト


array
[0][0]
array
[1][2]
array
[2][1]
array = [[-2, -1, 0], [2, 5, 3], [7, 8, 9]]
array[0]
array[1]
array[2]

3. 2次元リスト

list = [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]

のように、リストの要素にすべてに同じ値を入れたり、要素の値は気にせずに要素数だけあらかじめ決めておきたい場合は

list = [[0] * 4] * 3

list = [[0 for i in range(4)] for j in range(3)]

とすることができる

3. 2次元リスト

補足　多次元リスト

list = [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]

のように、2次元リストをさらに要素にした(並べた)リストを作ることができ、これを3次元リストという

同じようにn-1次元リストを並べたn次元リストを作ることができる

2次以上のリストを多次元リストという

3. 2次元リスト

補足　多次元リスト

list = [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]　のとき

list[0] = [[1, 2], [3, 4]]

list[0][1] = [3, 4]

list[1][0][1] = 6

リストから数値1つを取り出したい場合、n次元リスト変数の場合変数の右に[]がn個必要になる

3. 2次元リスト

補足　多次元リスト

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

のように、2次元リストを作成するときに小さなリストlist[0]とlist[1]の要素数が違ってもOK

また2次元リストの要素がリストでなくてもOK

a[1] = [3]

a[1][0] = 3

a[2] = 4

a[2][0]などは存在しない

演習2 2次元リスト

for文によるリスト変数宣言を用いて2次元リスト変数

L1 = [[0, 1, 2], [0, 1, 2]]

L2 = [[0, 1, 2], [3, 4, 5]]

を作成しよう

4. リストの演算

a = 3, b = 4　のとき
a + b = 7　となる　(四則演算)

リスト a = [2, 3], b = [5, 7] に対して
a + b　を行うと
[2, 3, 5, 7]
というリストができる　(リストの要素の追加)
aの要素の後にbの要素が追加されている

4. リストの演算

リスト a = [2, 3]
の各要素に対して2を足して[4, 5]にしたいときは
forを用いる

```
a = [ 2, 3 ]
for i in range(2):
    a[i] += 2
print(a)
```

4. リストの演算

リスト a = [2, 3, 5, 6]
の各要素に対して3を足した新たなリスト
c = [5, 6, 8, 9]を作成したいとき

```
a = [ 2, 3, 5, 6 ]
c = [ 0 ] * 4
for i in range(4):
    c[i] = a[i] + 3
print(c)
```

```
a = [ 2, 3, 5, 6 ]
c = [ i + 3 for i in a ]
print(c)
```

c = [a[i] + 3 for i in range(4)]
でもOK

4. リストの演算

リスト a = [2, 2, 3], b = [1, 2, 3]
の各i番目の要素a[i]とb[i]を足し合わせたリスト
c = [3, 4, 6]を作成したいとき

```
a = [ 2, 2, 3 ]
b = [ 1, 2, 3 ]
c = [ 0 ] * 3
for i in range(3):
    c[i] = a[i] + b[i]
print(c)
```

```
a = [ 2, 2, 3 ]
b = [ 1, 2, 3 ]
c = [ i + j for (i, j) in zip(a, b) ]
print(c)
```

c = [a[i] + b[i] for i in range(3)]
でもOK

演習3 リストの演算

- リスト変数 a = [3, 5, 6] と b = [3, 4, 2] の各i番目の要素を掛け合わせたリスト変数cを作成しよう

- リスト変数 a = [2, 3, 1] と b = [5, 3, 8] の各i番目の要素にある演算をしてリスト変数 c = [25, 27, 8] を作成しよう