



# 情報処理演習II

## No. 8

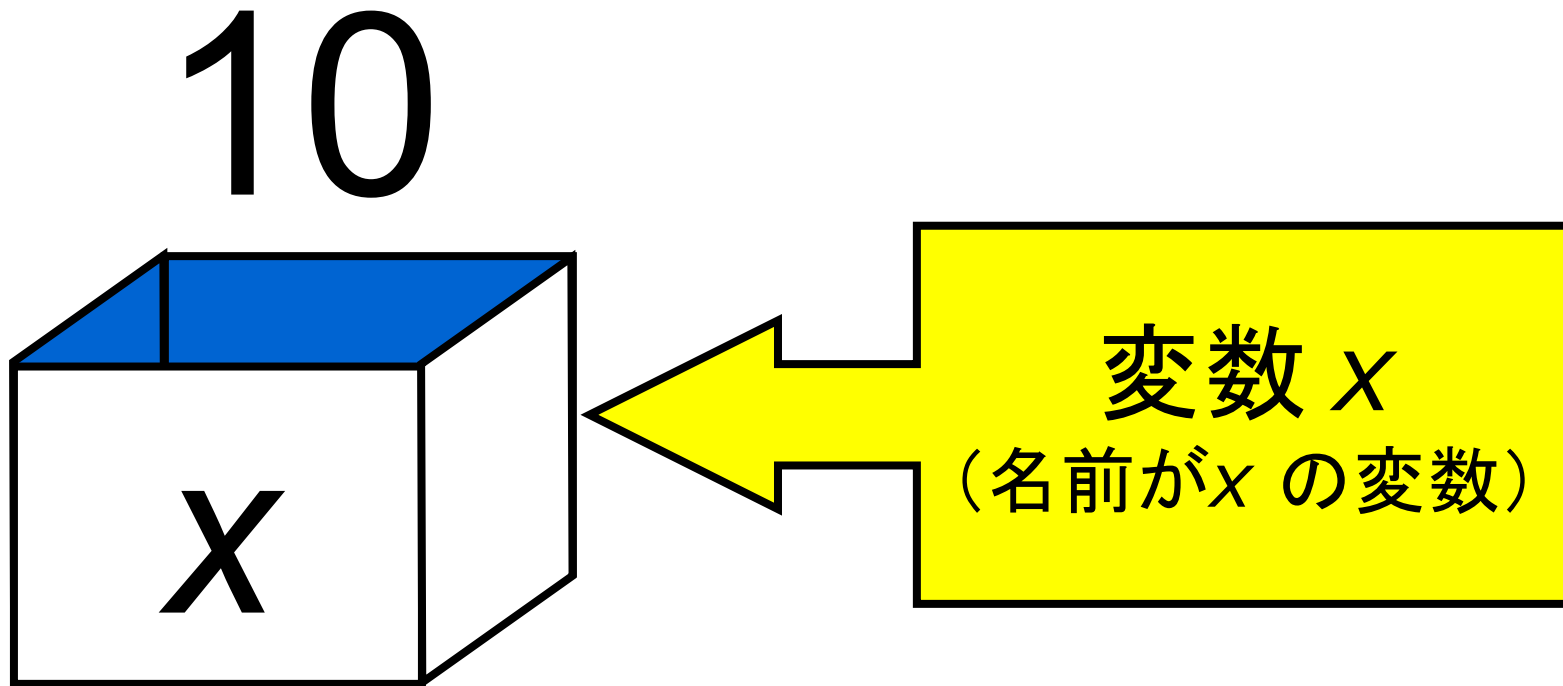
2023. 10. 17

芝浦工業大学 システム理工学部 機械制御システム学科

担当：桑原

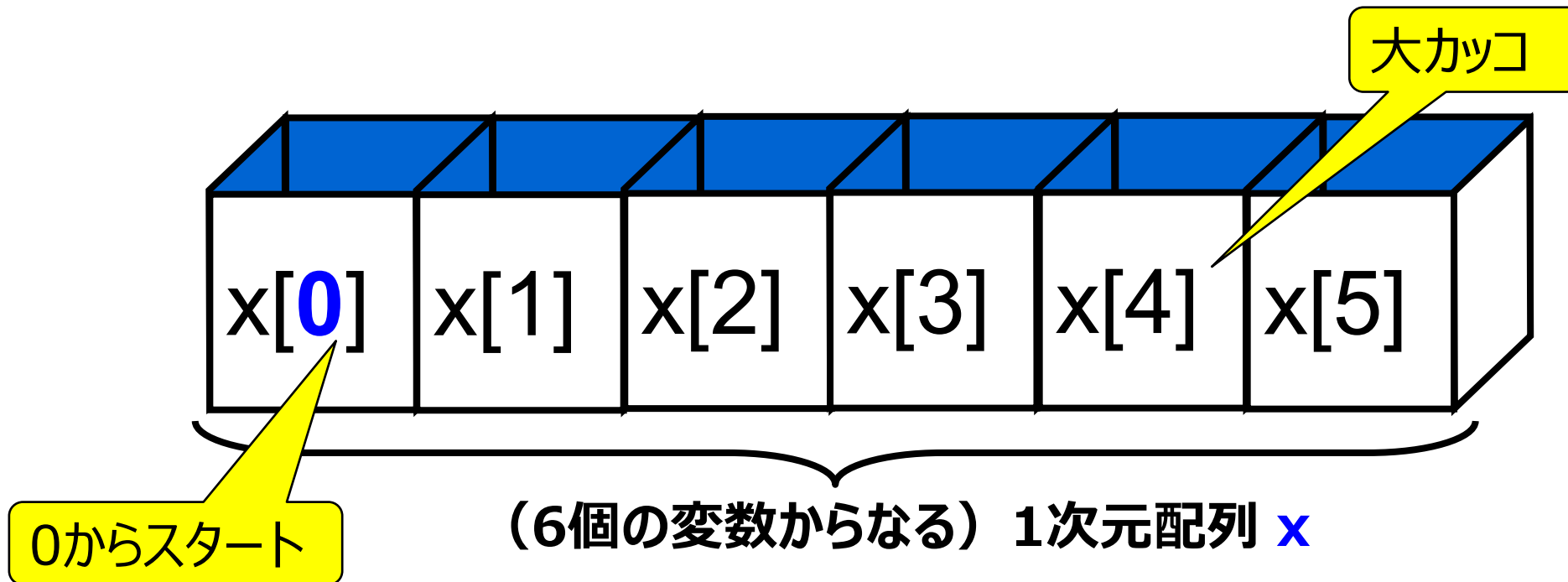
# 変数（復習）

- 変数とは：値（データ）を入れておくための箱。
- 名前がつけられている。
- 変数の値（箱の中身）は自由に変更できる。



# 配列とは? (1)

- **配列** = 同じ性質を持つ要素が一定の規則によって並んだ データの集合
- **複数の変数 (箱) を列や行に並べたもの**
- 変数と同様に、**配列に名前をつける (配列の宣言)**
  - 名前の付け方のルールは変数と同様
- 変数を一列に並べたものを**1次元配列**と呼ぶ

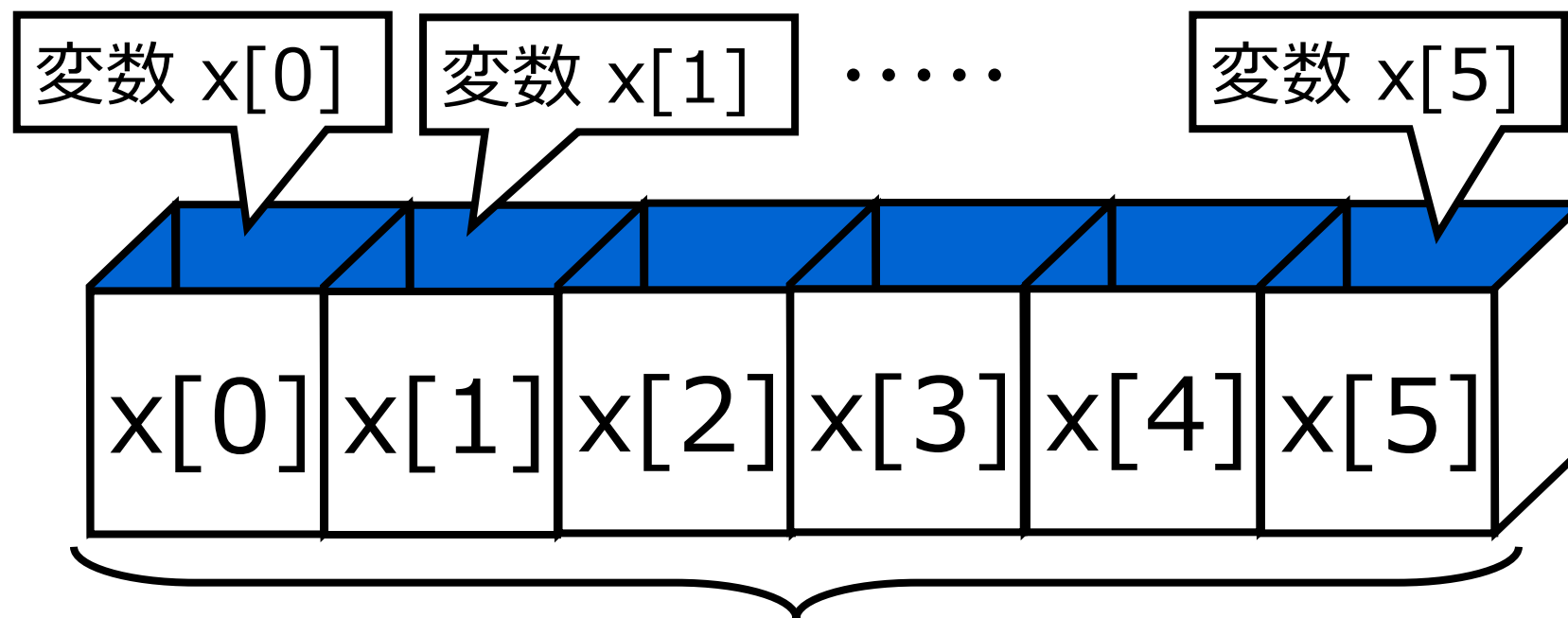


## 配列とは? (2)

- 配列の名前から、配列を構成する変数の名前が自動的に決まる。

1次元配列の場合：

**配列名[番号]** (番号は0から (要素数-1) まで)



(6個の変数からなる) 1次元配列 x

# 1次元配列の宣言

これが増えたことが  
これまでの  
"変数宣言"との違い

1次元配列の宣言：

(型宣言子) (配列名) [ (要素数) ] ;

- 例： `int array[5];`
  - **5**つの **int 型**変数を要素に持つ配列変数 **array** <sup>(アレイ)</sup> を宣言
  - 変数の要素の名前はそれぞれ：  
`array[0], array[1], array[2], array[3], array[4]`  
(括弧内の番号は 0から始まることに注意！)
  - 配列の各要素となる変数を「**配列要素**」, 各配列要素の括弧内の番号を「**配列の添え字**」と呼ぶ
- **配列宣言時の要素数に変数を用いることはできない**
  - ただし, 配列の添え字には変数を用いてよい

# 配列要素への値の代入

- 「各配列要素への値の代入」は、変数の場合と同様

例： `int array[5];` **1次元配列の宣言**

`array[0] = 1;`  
`array[1] = 2;` } **1つずつ要素（変数）に値を代入**

- まとめて代入する方法もある（注意）

例： `int array[5] = { 1, 2, 3, 4, 5 };`

（これは `array[0] = 1, array[1] = 2, array[2] = 3, array[3] = 4, array[4] = 5` という意味）

- 規則的に値を代入する場合は、繰返し制御文（for文, while文など）を使っても良い

# サンプルプログラム (1)

## サンプルプログラム 8.1

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    int i;
```

```
    int array[5];
```

```
    array[0] = 10;
```

```
    array[1] = 20;
```

```
    array[2] = 30;
```

```
    array[3] = 40;
```

```
    array[4] = 50;
```

```
    for (i = 0; i < 5; i++) {
```

```
        printf("%d 番目の要素変数の値は %d です\n", i+1, array[i]);
```

```
    }
```

```
}
```

```
int n=100;  
int array[n];
```

は誤り!!!

(配列宣言時の個数には**定数**しか使えない)

# サンプルプログラム (2)

## サンプルプログラム 8.2

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    int i;
```

```
    int array[5] = {10, 20, 30, 40, 50};
```


```
    for ( i = 0 ; i < 5 ; i++ ) {
```

```
        printf("%d 番目の要素変数の値は %d です¥n", i+1, array[i]);
```

```
    }
```

```
}
```

この代入は配列宣言時に有効で、かつデータは定数でないとNG





# サンプルプログラム (3)

## サンプルプログラム 8.3

```
#include <stdio.h>
```

```
void main()
```

```
{
```

```
    int  i;
```

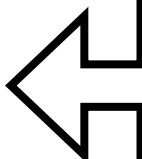
```
    int  array[100];
```

```
    /* 配列要素に値を代入 */
```

```
    for ( i = 0; i < 100; i++ ) {  
        array[i] = 10 * ( i + 1 );  
    }
```

```
    for ( i = 0; i < 100; i++ ) {  
        printf("%d 番目の要素変数の値は %d です¥n", i+1, array[i]);  
    }
```

```
}
```



**代入する値に規則性を持つなら、  
繰り返し制御文**を利用すると便利  
(要素数が多いときに有効)

# 配列のコピー

- 変数のコピー

正しい例：

```
int  x, y;  
x = 10;  
y = x;
```

**変数 y に変数 x の値が代入される**

(つまり、変数 x の内容を変数 y にコピーする)。  
よって変数 y の値は 10 である！

- 配列のコピー

誤った例：  

```
int  aa[3] = { 1, 3, 5 };  
int  ab[3];  
ab = aa;
```

配列 aa の内容 (各要素の値)  
を配列 ab にコピーしたい。しかし、  
**「ab = aa」ではダメ！**  
**要注意！**

- 要注意：** 代入演算子「=」によって配列を代入することはできない。

**配列をコピーするときは、配列要素を 1 つずつコピーしないといけない！**

# サンプルプログラム（4）

## サンプルプログラム 8.4

```
#include <stdio.h>

main()
{
    int i;
    int aa[3] = { 1, 3, 5 };
    int ab[3];

    /* 配列 aa を配列 ab にコピー */
    for ( i = 0; i < 3; i++ ) {
        ab[i] = aa[i];      /* 各要素ごとのコピー */
    }

    for ( i = 0; i < 3; i++ ) {
        printf(" 配列 ab の %d 番目の要素は %d です。¥n", i+1, ab[i]);
    }
}
```

## 2次元配列

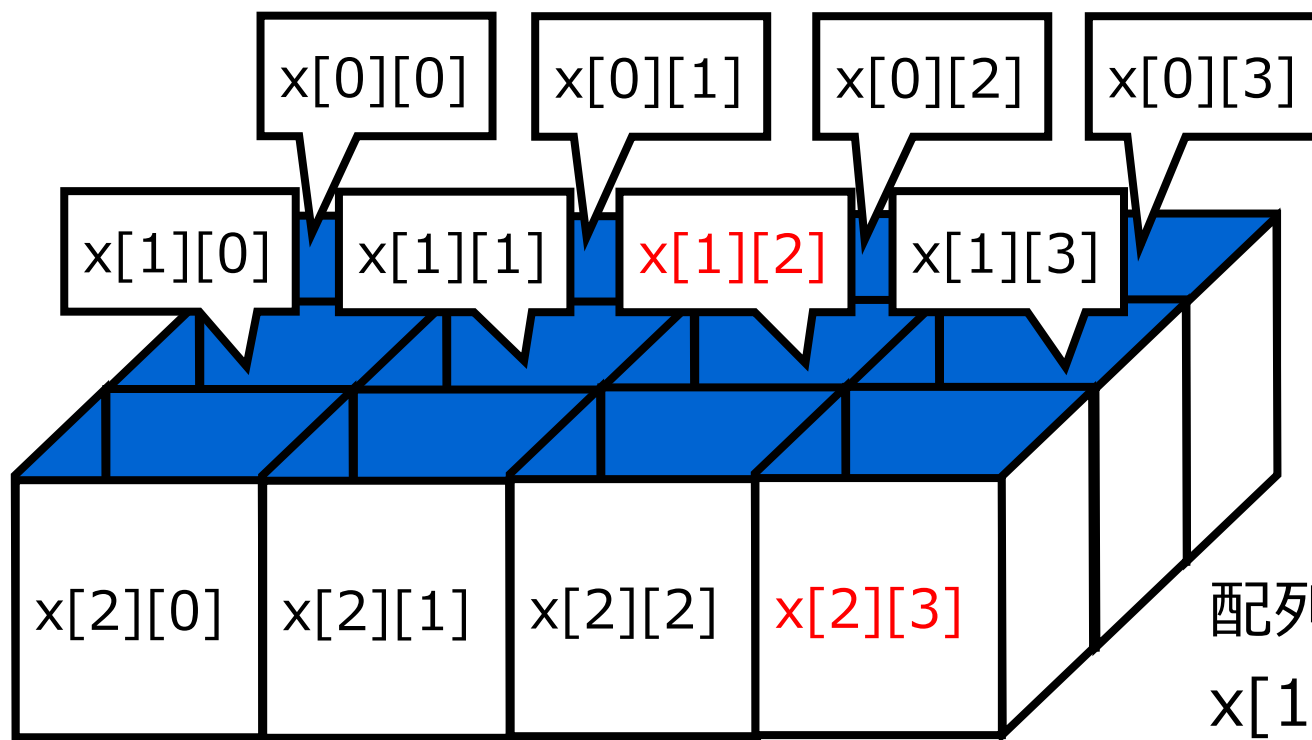
- 変数（データを格納する箱）を縦・横に並べて配列を作ることでもある = **2次元配列**
- 配列の添え字は、行・列を表すために2つ必要

# 2次元配列

(行 : タテ) (列 : ヨコ)

例 : `double x[3][4];`

→ 縦3×横4個のdouble 型の配列要素を持つ配列 x の宣言



配列要素の例 :

x[1][2], x[2][3], ...

## サンプルプログラム 各自で作成・実行してください。

キーボードから5つの小数データを配列に読み込み、  
入力後に表示するプログラムを作成せよ。

- ・配列変数の型は？ 名前は？ 個数は？
- ・scanf関数の利用 → 繰返し処理と組み合わせると便利

## 少なくとも出来るようになっておくべきこと

- ・配列変数に入っているデータの平均を求めるには？
- ・配列変数内のデータの最大値，最小値を求めるには？