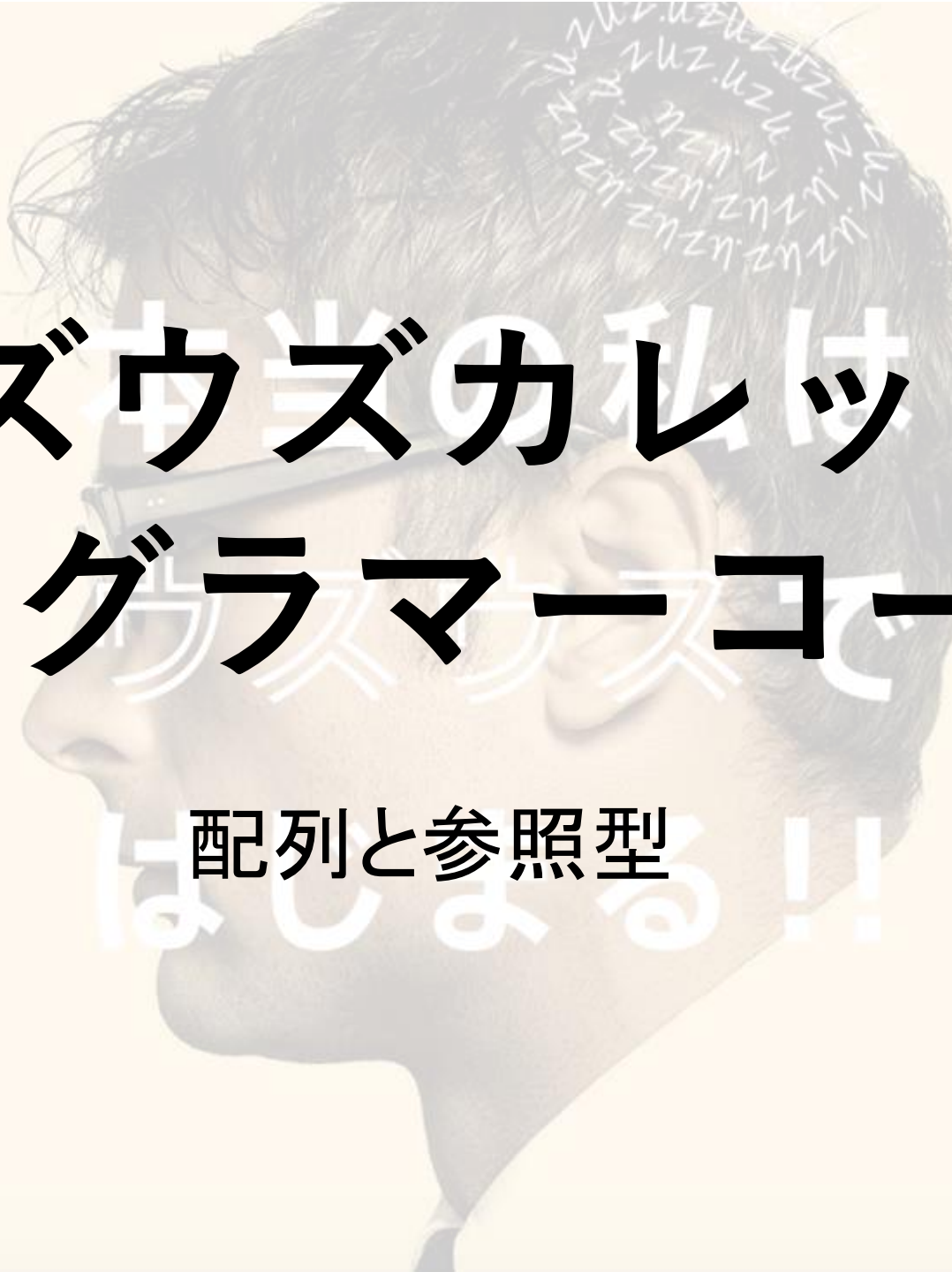
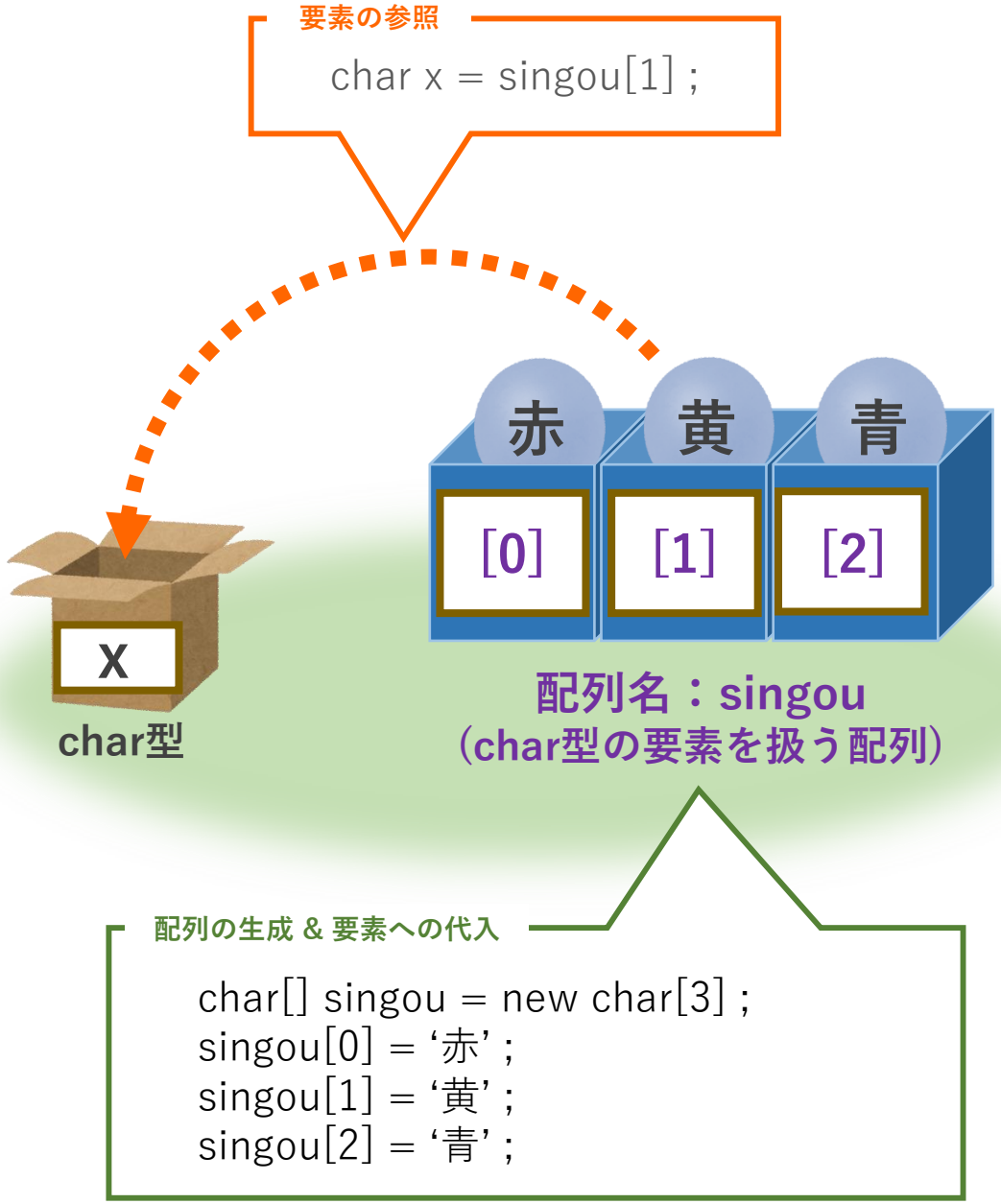


ウズウズカレッジ
プログラマーコース

配列と参照型



≪配列①≫

□**配列**は**同じ型の変数**を複数まとめた集合体のことです。
一つ一つの変数のことを**要素**と呼びます。

□配列を生成するには以下のように記述します。
『 型名[] 』の部分を**配列宣言**と言い、例えば char[] と書くと
char型の要素を扱う配列であることを表します。

型名[] 配列名 = new 型名[要素数] ;

□一つ一つの要素には**インデックス（添え字）**と呼ばれる番号が付与されており、**0から順に**番号が振られています。

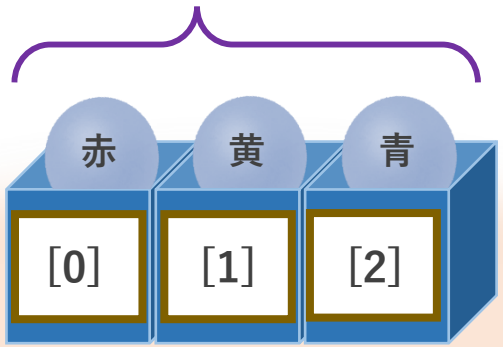
□参照・代入など、要素を使用したい場合は
『 **配列名[インデックス]** 』の書式で使用する要素を指定することができます。
指定方法が特殊なだけで、扱い方は変数と特に変わりません。

（例：要素への代入）
singou[0] = ‘赤’ ;

（例：要素の参照）
char x = singou[1] ;

一度に配列を初期化する方法

```
char[] singou = new char[] {‘赤’, ‘黄’, ‘青’} ;
char[] singou = {‘赤’, ‘黄’, ‘青’} ;
```



配列名：singou

要素数

要素数を取得する方法

```
int count = singou.length ;
```

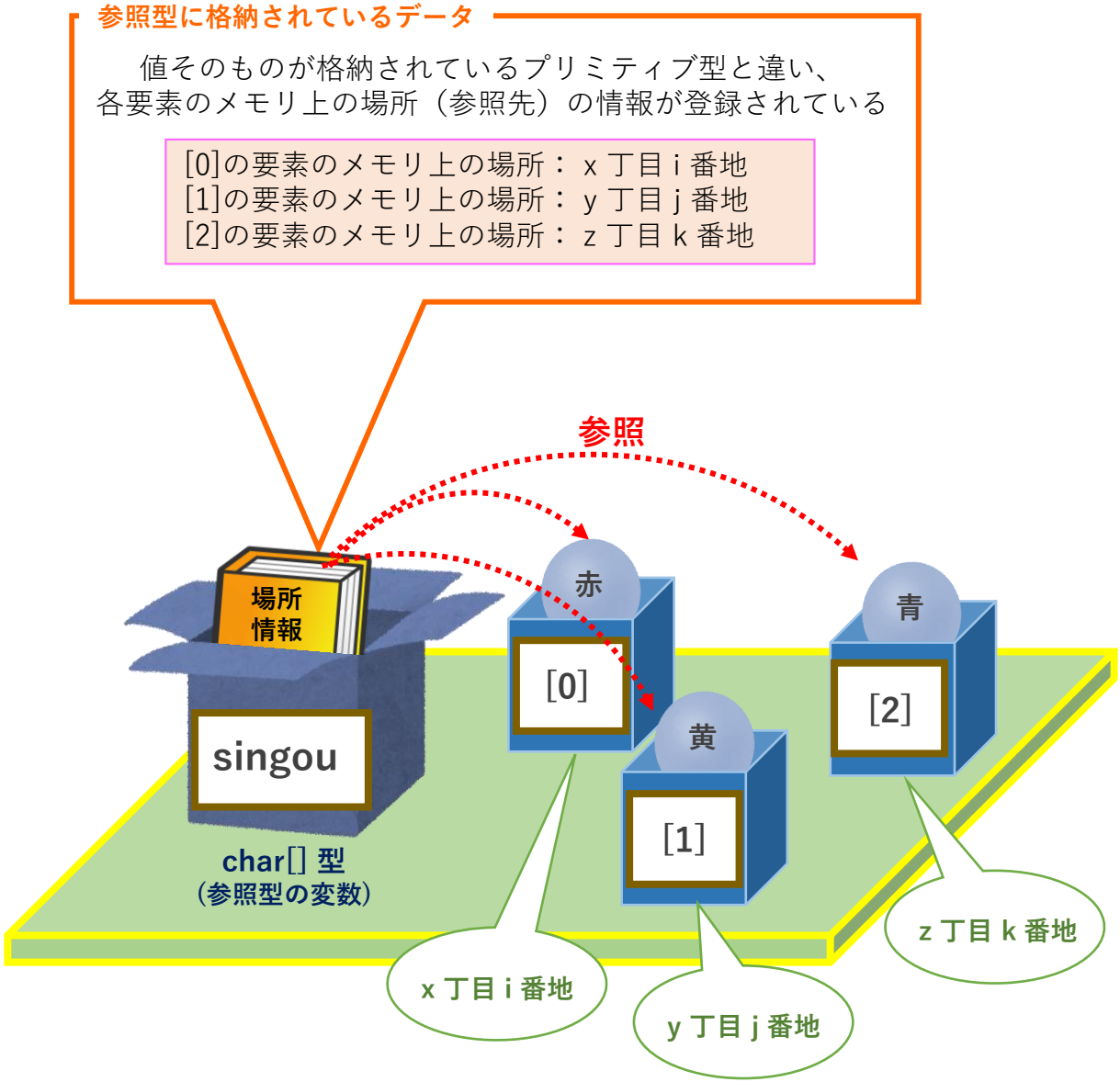
≪配列②≫

□配列の宣言と同時に複数の要素を初期化することも可能です。
一度に代入する際は { } を使用します。
以下のどちらでも初期化が可能です。

```
char[] singou = new char[] {‘赤’, ‘黄’, ‘青’} ;
char[] singou = {‘赤’, ‘黄’, ‘青’} ;
```

□配列の要素数を以下の方法で取得できます。頻繁に使うことになるので覚えておきましょう。

```
配列名.length
```



≪配列のしくみ（参照型）≫

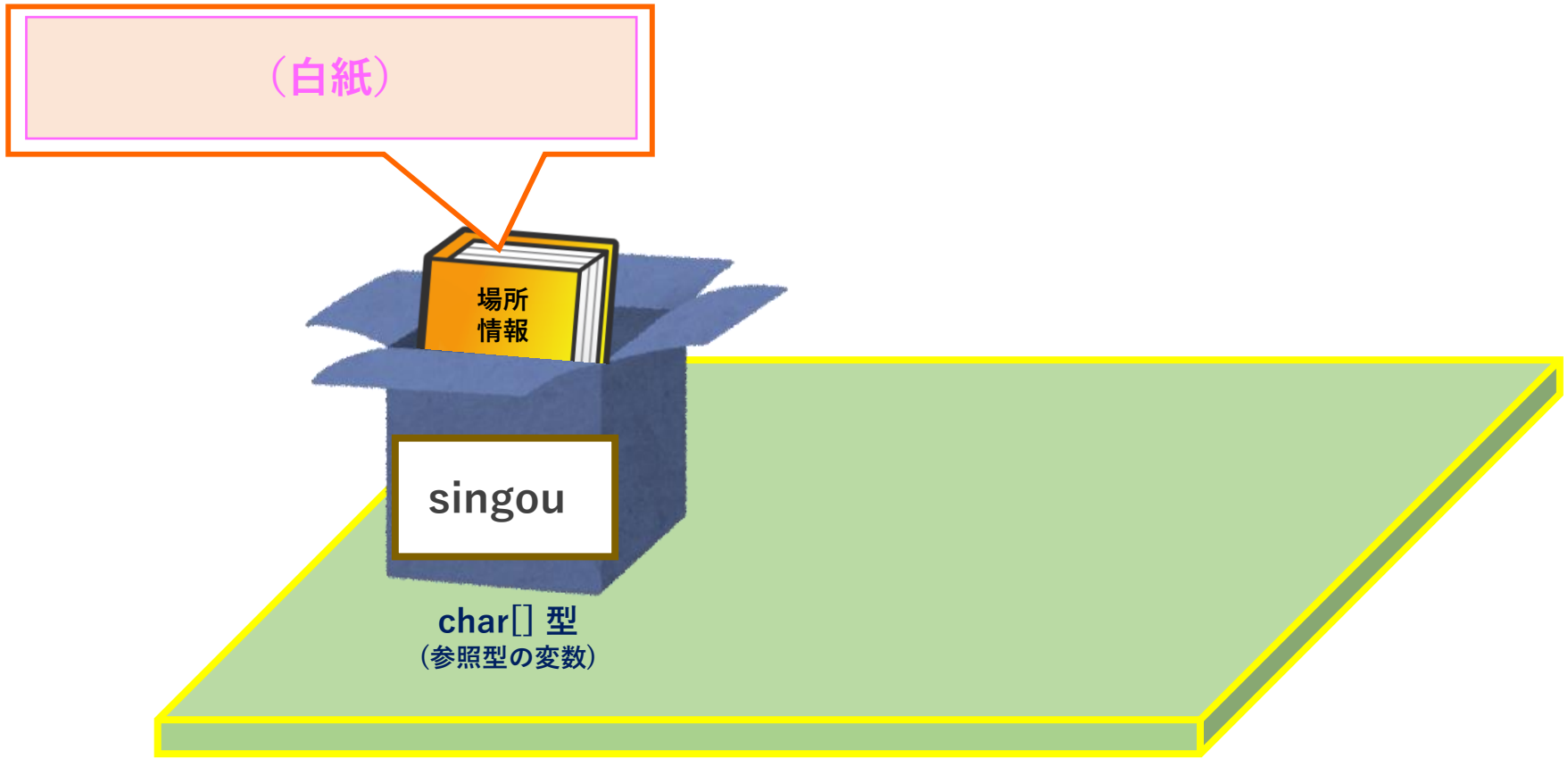
□実は変数には『**プリミティブ型（基本データ型）**』と『**参照型**』の2種類が存在します。
プリミティブ型はこれまで勉強してきた変数のことで、つまり値を1つだけ入れられる箱のようなもののことです。

□参照型はプリミティブ型のように値を直接保持しているわけではありません。**実際の値は別のメモリ領域で管理し、そのメモリ上における場所情報**を保持するのが参照型です。
参照型はこの場所情報を複数保持することが可能で、これらをアクセス先（参照先）として複数のデータを自身と結びつけることができます。

□配列は参照型に該当します。
1つ1つの要素はプリミティブ型であり、それらの場所情報を参照型変数として保持しています。

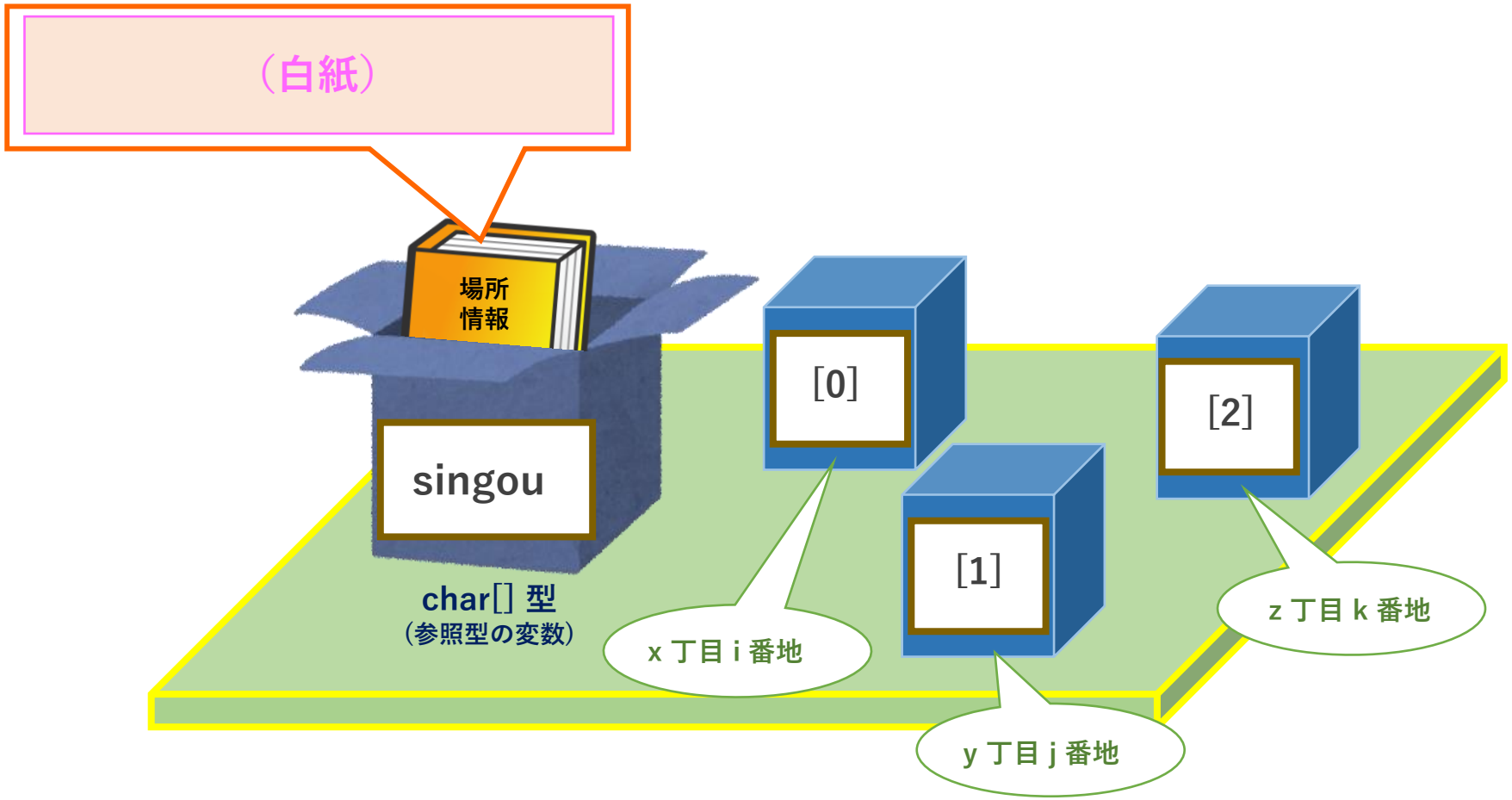
～配列が生成されるまで～

```
char[] singou = new char[3] ;
singou[0] = '赤' ;
singou[1] = '黄' ;
singou[2] = '青' ;
```



～配列が生成されるまで～

```
char[] singou = new char[3] ;
singou[0] = '赤' ;
singou[1] = '黄' ;
singou[2] = '青' ;
```

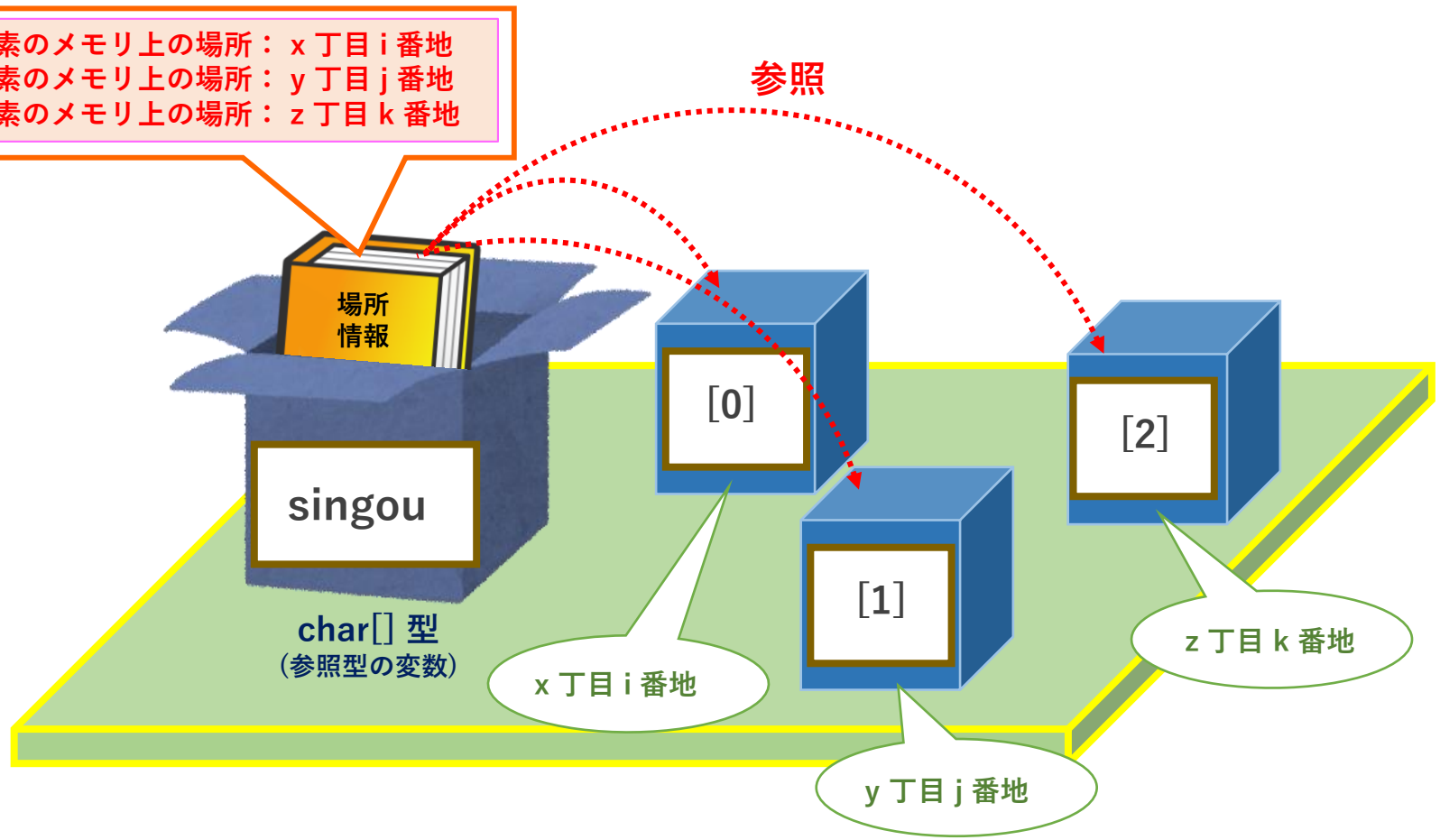


～配列が生成されるまで～

```
char[] singou = new char[3] ;
singou[0] = '赤' ;
singou[1] = '黄' ;
singou[2] = '青' ;
```

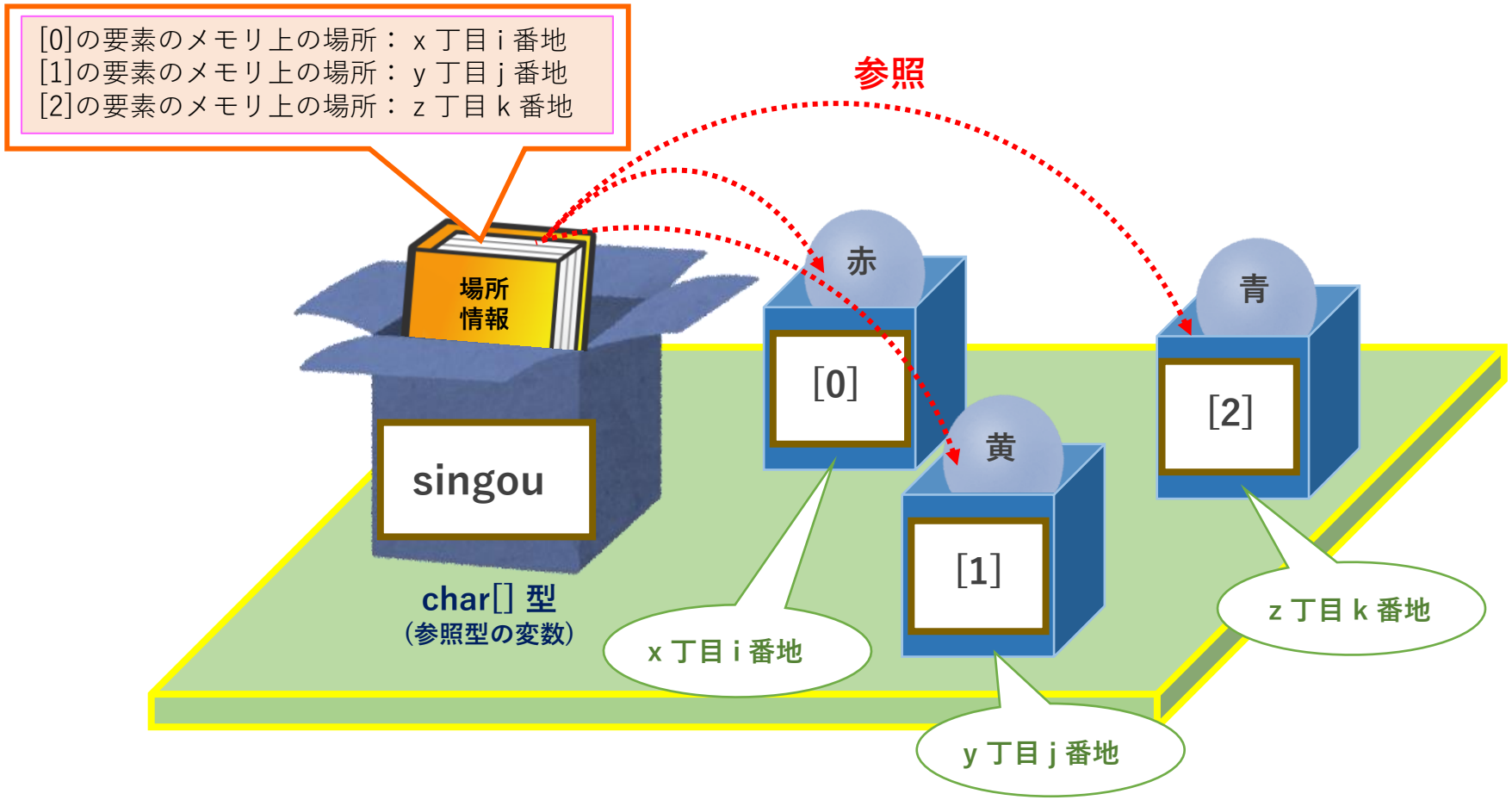
代入されているのは
データではなく場所情報

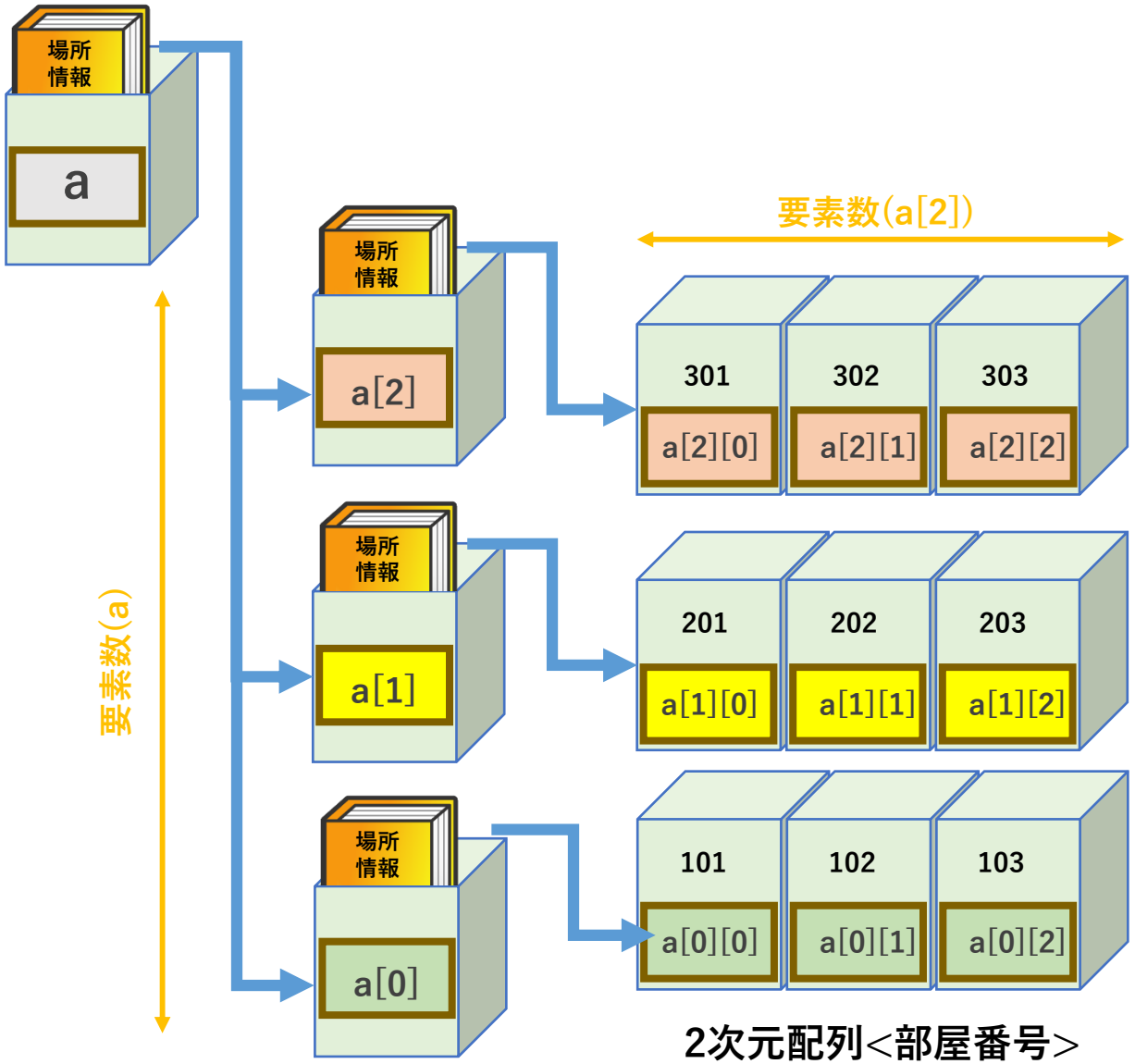
[0]の要素のメモリ上の場所： x 丁目 i 番地
[1]の要素のメモリ上の場所： y 丁目 j 番地
[2]の要素のメモリ上の場所： z 丁目 k 番地



～配列が生成されるまで～

```
char[] singou = new char[3] ;
singou[0] = ‘赤’ ;
singou[1] = ‘黄’ ;
singou[2] = ‘青’ ;
```





2次元配列<部屋番号>

≪多次元配列≫

□配列には**次元**の概念が存在し、2次元はもちろん3次元や4次元の多次元配列が存在します。

□配列宣言において下記の要領で [] を増やすことで多次元配列を作成することが可能です。

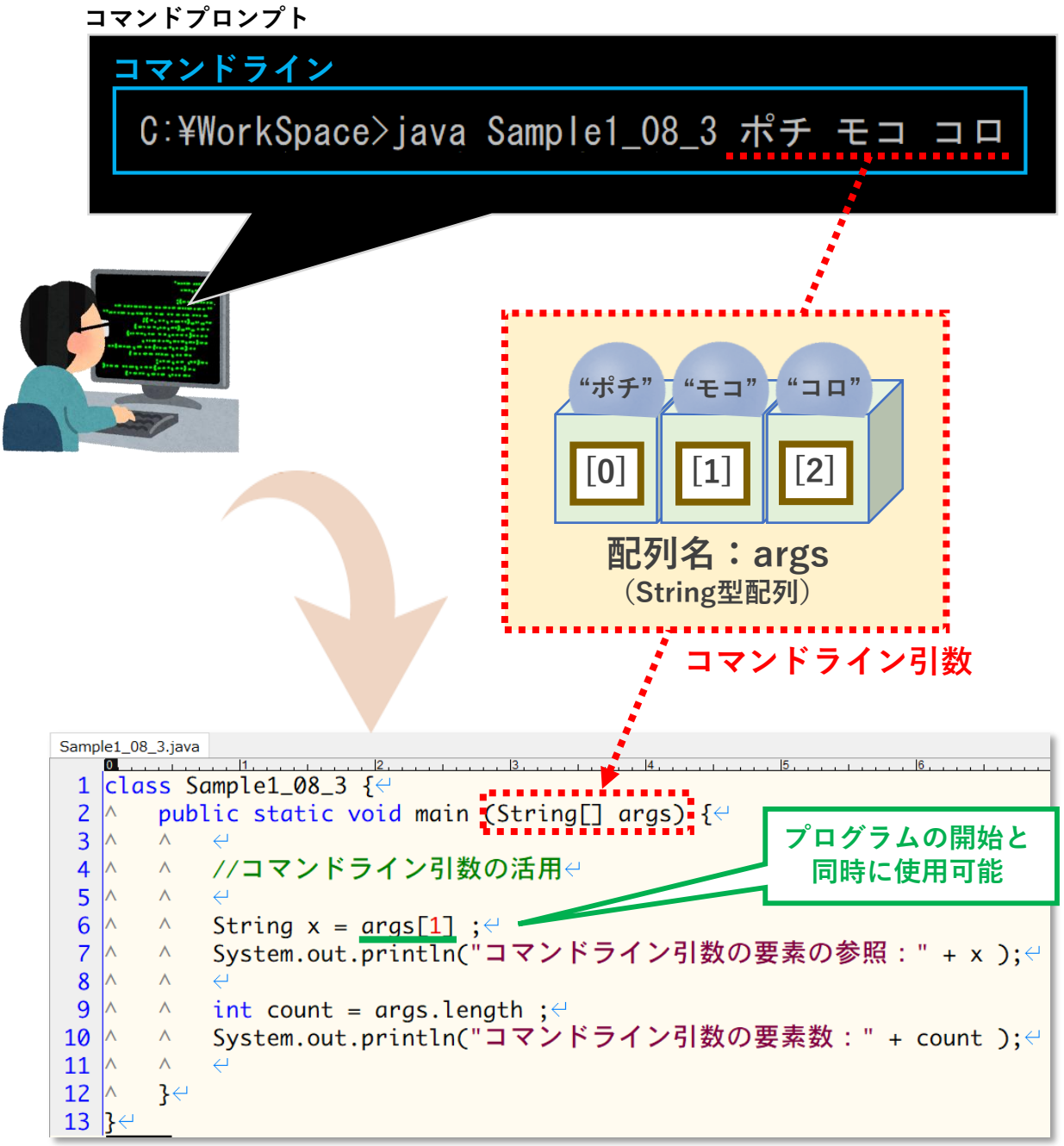
（2次元配列） int[][] a = new int[3][3] ;

（3次元配列） int[][][] b = new int[3][3][3] ;

□左図の2次元配列の初期化は以下のように行います。

int[][] a = {{101, 102, 103}, {201, 202, 203}, {301, 302, 303}} ;

□『配列名.length 』は厳密には**参照先の数**を取得しています。
左図の2次元配列の場合、aの参照先はa[0], a[1], a[2]の3つあるため、a.lengthで取得できる数値は3です。
a[0]の参照先はa[0][0], a[0][1], a[0][2]の3つであるため、a[0].lengthで取得できる数値はやはり3です。



≪コマンドライン引数≫

□**コマンドライン引数**は**コマンドライン（コマンドプロンプトの行のこと）**を通して**人間から受け取る特殊な配列**です。
コマンドプロンプト上でプログラム実行命令『Java クラス名』を打ち込んだ後に、半角スペース区切りでデータを入力するとJVMがそれらを**String型配列**に詰めて実行先のプログラムに渡してくれます。
この配列の名前は**args**と定められており、プログラムの処理開始と同時に使用可能な状態になっています。

□**引数**とはプログラムが外部から受け取るデータのことです。
（詳しい理解は後ほど行います）

＜演習：Ex1_08_1＞

```
/*-< 演習：Ex1_08_1 >-----------------------------------
コメントの内容に従ってプログラムを作成してください。
----------------------------------------------------*/
class Ex1_08_1 {
    public static void main (String[] args) {
        
        /* (1) 以下の配列を準備してください。
        **      配列名     ：names
        **      保持する値：「taro」「takeshi」「hanako」「moco」「satoshi」
        */
        
        
        
        
        /* (2) 以下のアンダーバーを埋め、
        **      names配列から「moco」の値を表示するプログラムに修正してください。
        */
        String printName = ______ ;          //(21行目)アンダーバーに適切な処理を埋めてください
        System.out.println( printName );     //(22行目)変更しないでください
        
    }
}
```

＜演習：Ex1_08_2＞

```
/*-< 演習：Ex1_08_2 >-----------------------------------
コメントの内容に従ってプログラムを作成してください。
----------------------------------------------------*/
class Ex1_08_2 {
    public static void main (String[] args) {

        /* 以下の配列を準備し、「トイプードル」が表示されるプログラムを作成してください。
        **      配列名      ：animals
        **      保持する値
        **          animals[0][0]：アメリカンショートヘア
        **          animals[0][1]：マンチカン
        **          animals[1][0]：ジャンガリアンハムスター
        **          animals[1][1]：ハリネズミ
        **          animals[2][0]：オカメインコ
        **          animals[2][1]：ブンチョウ
        **          animals[3][0]：ゴールデンレトリーバー
        **          animals[3][1]：トイプードル
        */

    }
}
```

＜演習：Ex1_08_3＞

```
/*-< 演習：Ex1_08_3 >--------------------------------
コマンドライン引数として３つの数値を受け取り、その合計値を
表示するプログラムを作成してください。
<例>
コマンドライン入力値：5 8 10
表示される値：23
-------------------------------------------------*/
class Ex1_08_3 {
    public static void main (String[] args) {
        int input1 = _____________ ;                    //(11行目)アンダーバーに適切な処理を埋めてください
        int input2 = _____________ ;                    //(12行目)アンダーバーに適切な処理を埋めてください
        int input3 = _____________ ;                    //(13行目)アンダーバーに適切な処理を埋めてください

        System.out.println( input1 + input2 + input3 );  //(15行目)変更しないでください

    }
}
```