１１回目　多次元配列

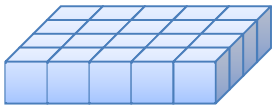
今日の講義で学ぶ内容

- ２次元配列とその使い方
- 不規則な２次元配列
- .length 修飾子

２次元配列

１次元配列　　**配列要素**が**直線的**に並ぶ**配列**です
次のように考えると分かりやすいでしょう

２次元配列　　**配列要素**が**平面的**に並ぶ**配列**です
次のように考えると分かりやすいでしょう



２次元以上の配列のことを**多次元配列**といいます

２次元配列の利用

２次元配列の利用手順　　**配列変数**の**宣言** → **配列要素**の**確保** → **配列要素**の**参照**

利用手順は１次元配列の場合と同様です

配列変数の宣言
型と**識別子**を指定して次のように行います

```
型　識別子[ ][ ];
```

または

```
型[ ][ ]　識別子;　　// Java での標準のスタイル
```

括弧[ ]の数が配列の**次元**を表します
たとえば、**int[][][] ary;** は int 型の３次元配列型の変数（配列変数）**ary** を宣言します

配列要素の確保

型と**配列要素**の**個数**を指定して次のように行います

```
識別子 = new 型 [ 行の配列要素の個数 ][ 列の配列要素の個数 ];
```

配列要素の参照

各**配列要素**の**参照**は配列変数の**識別子**と**添え字**を用いて次のようにします

```
識別子 [ 行の添え字 ] [ 列の添え字 ]
```

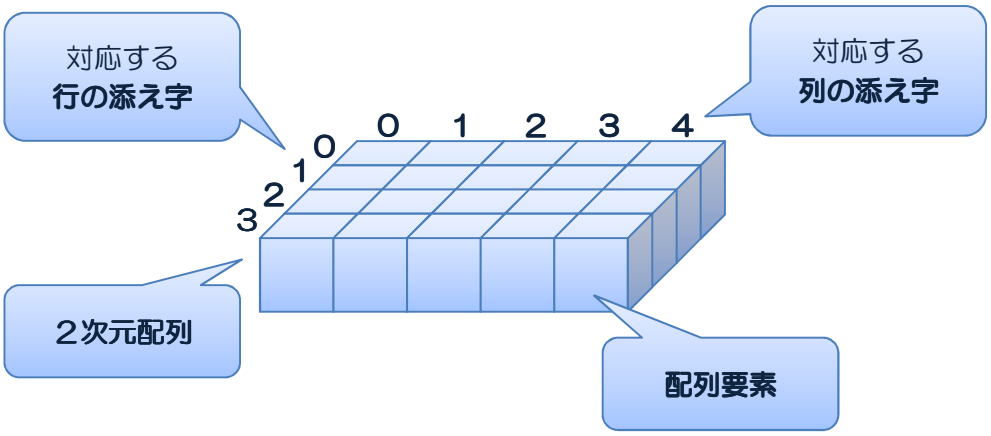
添え字には個々の**配列要素**の**位置（行方向**と**列方向）**を表す**０以上の整数**を指定します

・**行**の**添え字**には ０ ～ **行の配列要素の個数－１** までの**整数**を指定します

・**列**の**添え字**には ０ ～ **列の配列要素の個数－１** までの**整数**を指定します

添え字は、１次元の配列と同様に**０から始まる**ことに注意しましょう

例えば、**４行５列**の**配列要素**をもつ**２次元配列**の場合は次のようにして**添え字**を指定します



配列要素への**値**の**代入**は、各**配列要素**を**参照**して次のように行います

```
識別子 [ 行の添え字 ][ 列の添え字 ] = 値;
```

ソースコード例

ソースファイル名：Sample11_1.java

```
// 配列を用いて３人の学生（３行）の２科目の点数（２列）を管理する
class Sample11_1
{
    public static void main(String[] args)
    {
        // 配列変数の宣言
        int[][] test; // int test[][]; とも記述可能

        // 配列要素の確保
        test = new int[3][2];

        // 配列変数の宣言と配列要素の確保は同時に記述可能
        // int test[][] = new int[3][2];
        // int[][] test = new int[3][2];

        // 各配列要素へ値を代入
        // 添え字は０から（要素数）－１まで！！
        test[0][0]=80;  test[0][1]=50;
        test[1][0]=60;  test[1][1]=75;
        test[2][0]=22;  test[2][1]=90;

        // 各配列要素を順番に出力
        for(int i=0; i<3; i++)
        {
            System.out.println(i+"番目の学生の得点：");
            System.out.println("科目１："+test[i][0]+"¥t 科目２："+test[i][1]);
        }
    }
}
```

実行画面

```
>java Sample11_1
０番目の学生の得点：
科目１：80      科目２：50
１番目の学生の得点：
科目１：60      科目２：75
２番目の学生の得点：
科目１：22      科目２：90
```

２次元配列の初期化

２次元配列の初期化は、**配列変数**の**宣言時**に次のように行います
指定する値が指定する**行**と**列**に代入された**配列要素**をもつ**配列変数**が宣言されます

```
型　識別子[ ][ ] = { { ０行０列, ０行１列, … }, { １行０列, １行１列, … }, … { … } };
```

または

```
型[ ][ ]　識別子 = { { ０行０列, ０行１列, … }, { １行０列, １行１列, … }, … { … } };
```

括弧{ }が**入れ子**になっていることに注意しましょう

ソースコード例
ソースファイル名：Sample11_2.java

```
//  ２次元配列の初期化
class Sample11_2
{
	public static void main(String[] args)
	{
		//  ２次元配列の初期化
		int[][] test={{80, 50}, {60, 75}, {22, 90}};
		// int test[][]={{80, 50}, {60, 75}, {22, 90}};  とも記述可能

		//  各配列要素を順番に出力
		for(int i=0; i<3; i++)
		{
			System.out.println(i+"番目の学生：");
			System.out.println("科目１："+test[i][0]+"¥t 科目２："+test[i][1]);
		}
	}
}
```

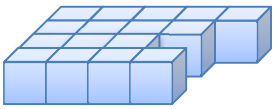
実行画面

```
>java Sample11_2
０番目の学生：
科目１：80　　科目２：50
１番目の学生：
科目１：60　　科目２：75
２番目の学生：
科目１：22　　科目２：90
```

配列の初期化では配列要素の数を指定しません
配列要素の数は{ }の入れ子や値の列より自動的に計算されます

不規則な２次元配列

不規則な２次元配列　　各行の配列要素の数がそれぞれ異なる**２次元配列**です



不規則な２次元配列の作り方
不規則な２次元配列は次の２通りの方法で作成することができます
・配列の**初期化**を利用して作成する方法
・配列変数の宣言と配列要素の確保の**利用手順**にしたがい作成する方法

配列の初期化を利用して作成する方法
ソースコード例
ソースファイル名：Sample11_3.java

```
// 不規則な２次元配列で初期化する
class Sample11_3
{
    public static void main(String[] args)
    {
        int i, j;

        // ２次元配列の初期化
        Int[][] test={{80, 60, 22}, {50, 75}, {72, 33, 75, 63}};

        // 各行の列数と要素を順番に出力します
        // .length は指定行の列数を値としてもちます（.length の詳細は次節）
        for(i=0; i<3; i++)
        {
            System.out.print(i+"行目の要素数は"+test[i].length+"で、");
            for(j=0; j<test[i].length; j++)
                System.out.print(test[i][j]+" ");
            System.out.println("です");
        }
    }
}
```

配列の**初期化**による**不規則な配列**
配列の列方向の配列要素を指定する内側の括弧{ }内で**列挙する値の個数を変えて**やればよいです

test[i].length
test[i].length は行方向の添え字 i で**指定される行**の**列数**を表します（.length 修飾子の詳細は次節で）

実行画面

```
>java Sample11_3
0 行目の要素数は 3 で、80 60 22 です
1 行目の要素数は 2 で、50 75 です
2 行目の要素数は 4 で、72 33 75 63 です
```

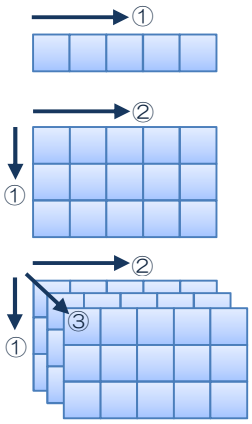
.length 修飾子とその利用

.length 修飾子　　**配列要素の数を得るための修飾子**

1 次元配列の場合：　① **配列変数.length**　　**配列要素の数**

2 次元配列の場合：　① **配列変数.length**　　**行数**
② **配列変数[i].length**　　第 i 行の**列数**

3 次元配列の場合：　① **配列変数.length**　　**行数**
② **配列変数[i].length**　　第 i 行の**列数**
③ **配列変数[i][j].length**　　第 i 行 j 列の**高さ**



4 次元以降も同様です

ソースコード例
ソースファイル名：Sample11_4.java

```
// .length 修飾子
class Sample11_4
{
    public static void main(String[] args)
    {
        // 配列の初期化
        int test[]={72, 33, 75, 63};
        System.out.println("配列要素の数は"+test.length+"です。");
    }
}
```

実行画面

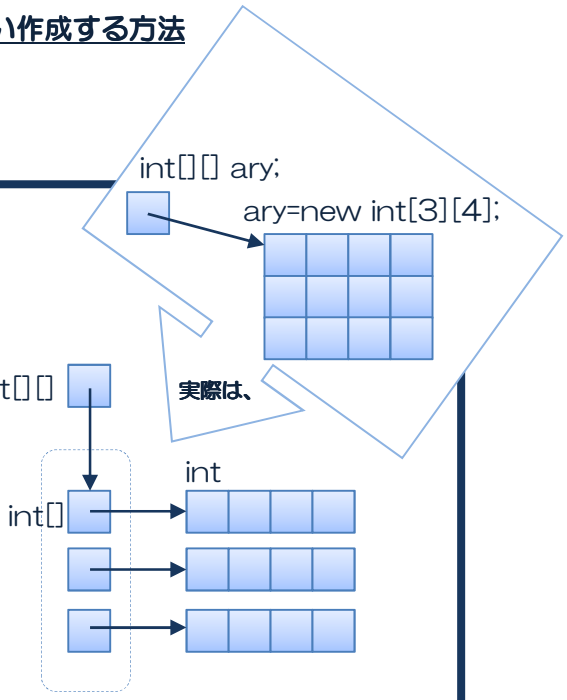
```
>java Sample11_4
配列要素の数は 4 です。
```

配列変数の宣言と配列要素の確保の利用手順にしたがい作成する方法

ソースコード例

ソースファイル名：Sample11_5.java

```
// 不規則な２次元配列を宣言する
class Sample11_5
{
    public static void main(String[] args)
    {
        // (ポイント)
        // ２次元配列は、１次元配列の配列　である

        // int 型の配列型の配列型の変数
        int[] test[];

        // 下記と同じ意味である
        // int test[][];
        // int[][] test;

        // 行数のみで部分的に２次元配列を作成
        test = new int[3][];  // 配列は３行からなる

        // 各行に列を作成して２次元配列は完成
        test[0] = new int[3]; // ０行目の配列要素は３つ
        test[1] = new int[2]; // １行目の配列要素は２つ
        test[2] = new int[4]; // ２行目の配列要素は４つ

        // 各行の列数（.length の詳細は前節）と要素を順番に出力
        System.out.println("配列の行数は"+test.length+"であり、");
        for(int i=0; i<test.length; i++)
            System.out.println(i+"行目の配列要素は"+test[i].length+"です");
    }
}
```



実行画面

```
>java Sample11_5
配列の行数は３であり、
０行目の配列要素は３です
１行目の配列要素は２です
２行目の配列要素は４です
```

参考：例題 **Sample11_5** の変数の振る舞いは図的に次のように理解できます