

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

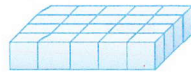
1次元配列

配列要素が直線的に並び配列です
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び配列です
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[][]; 2D
```

または

```
型[][] 識別子; // Javaでの標準のスタイル
```

```
コード例 | int[][] ary;
```

括弧[]の数が配列の次元を表します

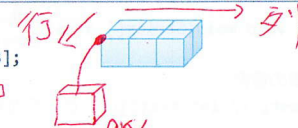
たとえば、int[][][] ary; はint型の3次元配列型の変数(配列変数)aryを宣言します

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];
```

```
コード例 | int[][] ary;
           | ary = new int[2][3];
```



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

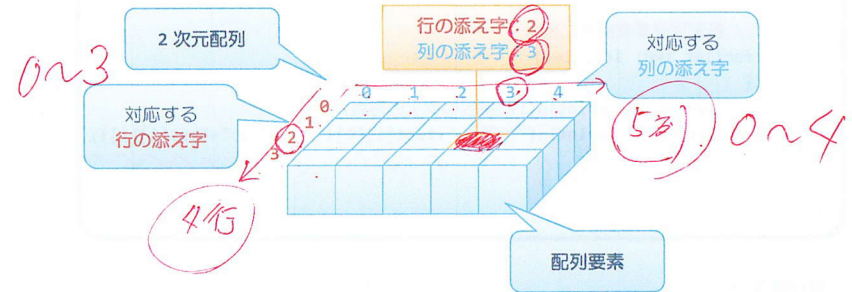
```
識別子 [行の添え字][列の添え字] = 値;
```

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

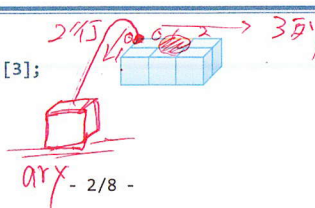
例えば、4行5列の配列要素をもつ2次元配列の場合は次のようにして添え字を指定します



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;
```

```
コード例 | int[][] ary;
           | ary = new int[2][3];
           | ary[0][1] = 1;
```



ソースコード例
 ソースファイル名: Sample11_1.java

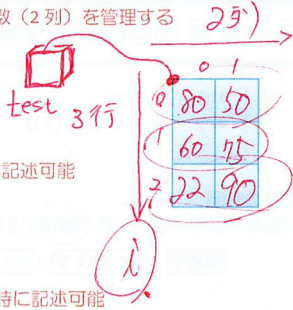
```
// 配列を用いて3人の学生(3行)の2科目の点数(2列)を管理する
class Sample11_1
{
  public static void main(String[] args)
  {
    // 配列変数の宣言
    int[][] test; // int test[][]; とも記述可能

    // 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]+"科目 2:"+test[i][1]);
    }
  }
}
```



実行画面

```
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
```

= {0, 0, 0, 0}

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

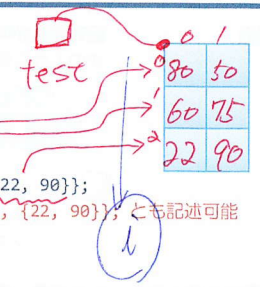
```
型[][] 識別子 = { {0行0列, 0行1列, ...}, {1行0列, 1行1列, ...}, ... };
```

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]+"科目 2:"+test[i][1]);
    }
  }
}
```



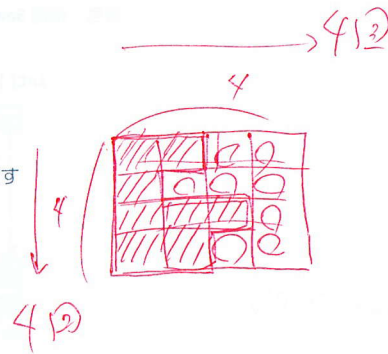
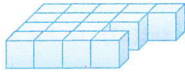
実行画面

```
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

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

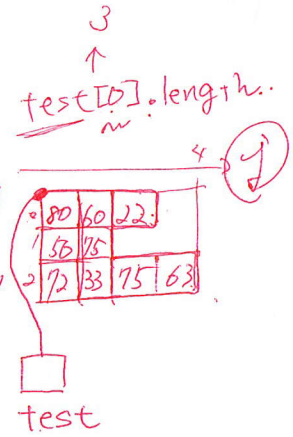
```
// 不規則な 2 次元配列で初期化する
class Sample11_3
{
    public static void main(String[] args)
    {
        int i, j;

        // 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

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 配列要素の数を取得するための修飾子です

- 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

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

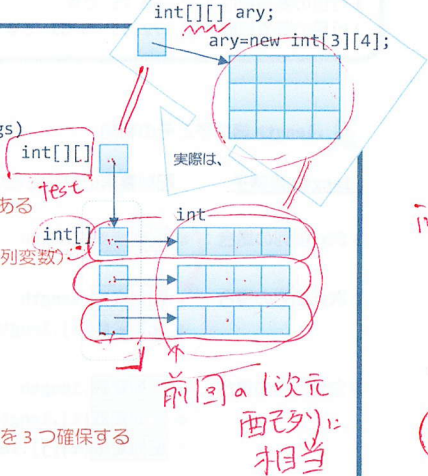
```
// 不規則な 2 次元配列を宣言する
class Sample11_5
{
    public static void main(String[] args)
    {
        // (ポイント)
        // 2 次元配列は"1 次元配列の配列"である
        // int 型の配列型の配列型の変数 (配列変数)
        int[] test[];

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

        // int 型の配列型の変数 (配列変数) を 3 つ確保する
        test = new int[3][];

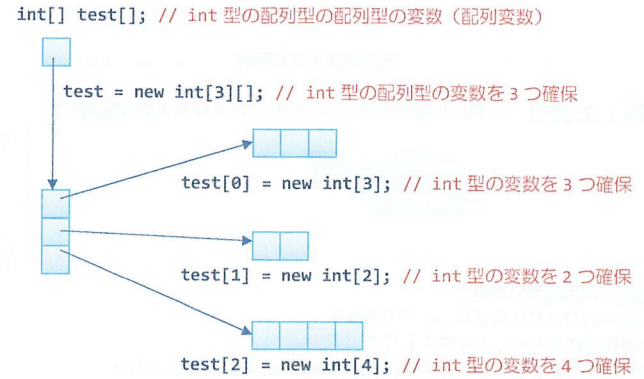
        // 各配列変数に int 型の変数をそれぞれの個数だけ確保して 2 次元配列は完成
        test[0] = new int[3]; // 0 行目の配列要素は 3 つ
        test[1] = new int[2]; // 1 行目の配列要素は 2 つ
        test[2] = new int[4]; // 2 行目の配列要素は 4 つ

        // 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");
    }
}
```



int[] ary;
int[] test
int[] test

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



今日の講義のまとめ

- 1 次元配列は配列要素が直線的に並び配列のことです。2 次元配列は配列要素が平面的に並び配列のことです。一般に、2 次元以上の配列を多次元配列といいます。
- 2 次元配列の配列要素を指定するとき、行方向と列方向の 2 つの添え字が必要です。
- 2 次元配列の初期化は、行単位で値を列挙することにより行います。
- 各行の列数が異なる不規則な 2 次元配列を作成することができます。
- .length 修飾子を用いることにより、配列要素の数を知ることができます。

実行画面

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