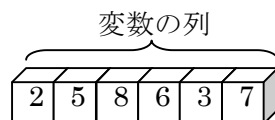


● 配 列

配 列

同じ型の複数の変数を一括して管理する機能
直観的には、



● 配列の準備

配列の準備手順

配列変数の宣言 → 配列要素の確保 → 配列要素の初期化

配列変数の宣言

配列を扱う変数（配列変数という）を用意する
配列を参照するときに使われる

型と識別子を指定して次のように行う。

```
型 識別子[];
```

または

```
型[] 識別子;
```

配列要素の確保

値を格納するための領域（配列要素という）を用意する
各配列要素は一つの変数としての機能をもつ

型と配列要素の個数を指定して次のように行う。

```
識別子 = new 型[配列要素の個数];
```

※new 演算子は、指定された個数の配列要素をコンピュータのメモリ上に確保する。

配列要素の初期化 配列要素を確保した際に適当な値を入れておくこと
(確保されたままの配列要素にはゴミが入っているため)

各配列要素は、配列変数の識別子と添え字（0から(要素数) - 1までの整数）を用いて参照できる。配列要素の初期化は、各配列要素を参照して次のように行う。

`識別子[添え字] = 値;`

ソースコード例

ソースファイル名 : Sample10_1.java

```
// 配列を用いて5人の学生の点数を管理する
class Sample10_1
{
    public static void main(String[] args)
    {
        int i;

        // 配列変数の宣言
        int test[];
        // int[] test; と同記述可能

        // 配列要素の確保
        test = new int[5];

        // 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[5];
        // int[] test = new int[5];

        // 各配列要素の初期化（兼、値の代入）
        test[0]=80;
        test[1]=60;
        test[2]=22;
        test[3]=50;
        test[4]=75;

        // 各配列要素（添え字は0から4まで）を順番に出力
        for(i=0; i<5; i++)
            System.out.println((i+1)+"番目の学生の点数は"+test[i]+"です。");
    }
}
```

実行画面

```
>java Sample10_1
1 番目の学生の点数は 80 です。
2 番目の学生の点数は 60 です。
3 番目の学生の点数は 22 です。
4 番目の学生の点数は 50 です。
5 番目の学生の点数は 75 です。
-- Press any key to exit (Input "c" to continue) --
```

ソースコード例

ソースファイル名 : Sample10_2.java

```
// 配列要素の動的な確保
import java.io.*;
class Sample10_2
{
    public static void main(String[] args) throws IOException
    {
        int i;
        int num;    // 学生数
        int[] test; // 学生の点数を保存する配列変数

        // キーボード入力の準備
        BufferedReader br;
        br = new BufferedReader(new InputStreamReader(System.in));

        // キーボード入力
        System.out.println("学生の人数を入力してください。");
        num=Integer.parseInt(br.readLine());

        // 学生数分の配列要素を確保
        test = new int[num];

        // キーボードから点数を配列要素へ順番に入力する
        for(i=0;i<num;i++)
            test[i]= Integer.parseInt(br.readLine());

        // 配列要素に入力されている点数を順番に出力する
        for(i=0;i<num;i++)
            System.out.println((i+1)+"番目の学生の点数は"+test[i]+"です。");
    }
}
```

実行画面

```
>java Sample10_2
学生の人数を入力してください。
3
89
75
95
1 番目の学生の点数は 89 です。
2 番目の学生の点数は 75 です。
3 番目の学生の点数は 95 です。
-- Press any key to exit (Input "c" to continue) --
```

● 配列の初期化

配列の初期化

配列要素を一度に初期化する

配列の初期化は識別子の宣言時に次のように行う。与えられた値を配列要素にもつ配列変数が宣言される。また、配列の初期化では配列要素の数を指定しない。

```
型 識別子[] = { 値1, 値2, 値3, . . . , 値n };
```

または

```
型[] 識別子 = { 値1, 値2, 値3, . . . , 値n };
```

ソースコード例

ソースファイル名 : Sample10_3.java

```
// 配列の初期化
class Sample10_3
{
    public static void main(String[] args)
    {
        int i;
        // 配列の初期化
        int test[]={80,60,22,50,75};
        // int[] test={80,60,22,50,75}; と同記述可能

        // 配列要素を出力
        for(i=0;i<5;i++)
            System.out.println((i+1)+"番目の学生の点数は"+test[i]+"です。");
    }
}
```

実行画面

```
>java Sample10_3
1 番目の学生の点数は 80 です。
2 番目の学生の点数は 60 です。
3 番目の学生の点数は 22 です。
4 番目の学生の点数は 50 です。
5 番目の学生の点数は 75 です。
-- Press any key to exit (Input "c" to continue) --
```

● 配列変数と参照型変数

変数には 基本型変数

値として“値”そのものをもつ
例えば、int 型変数、double 型変数など

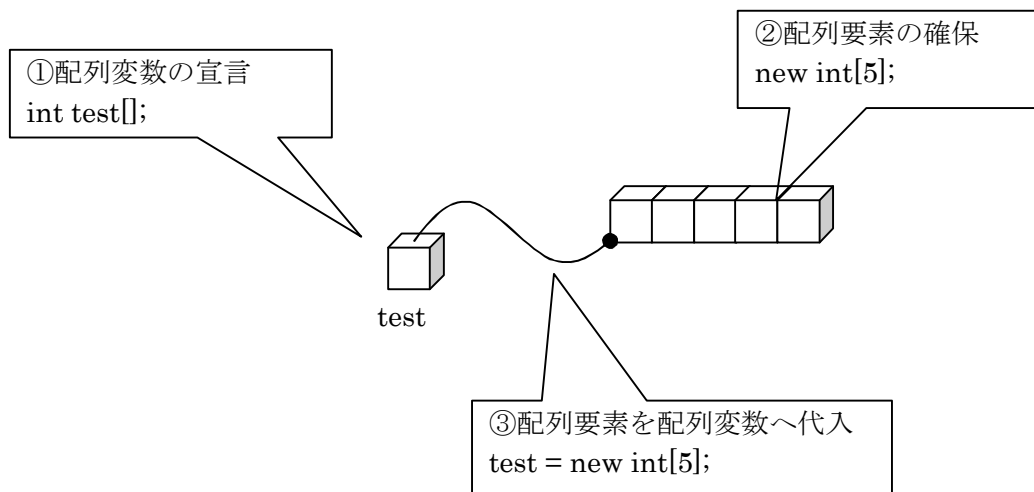


参照型変数

値として“値のある場所”をもつ
例えば、配列変数、クラス変数など



参考：例題 Sample10_1 での配列変数の宣言と配列要素の確保は次のように理解できる



ソースコード例

ソースファイル名 : Sample10_4.java

```
// 配列変数へ代入するということは？
class Sample10_4
{
    public static void main(String[] args)
    {
        int i;
        int n1=1, n2;
        int[] ary1={1,2,3}, ary2;

        // int 型変数へ代入
        n2=n1;
        // 配列変数へ代入
        ary2=ary1;

        // 変数と配列要素を出力
        System.out.println("n1="+n1+", n2="+n2);
        System.out.print("ary1={");
        for(i=0;i<3;i++)
            System.out.print(ary1[i]+" ");
        System.out.println("}");
        System.out.print("ary2={");
        for(i=0;i<3;i++)
            System.out.print(ary2[i]+" ");
        System.out.println("}");

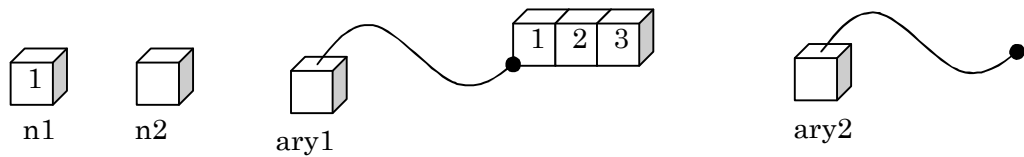
        // 一方の int 型変数の値を変更
        System.out.println("n2 := 2;");
        n2=2;
        // 一方の配列要素の値を変更
        System.out.println("ary2[1] := 4;");
        ary2[1]=4;

        // 変数と配列要素を出力
        System.out.println("n1="+n1+", n2="+n2);
        System.out.print("ary1={");
        for(i=0;i<3;i++)
            System.out.print(ary1[i]+" ");
        System.out.println("}");
        System.out.print("ary2={");
        for(i=0;i<3;i++)
            System.out.print(ary2[i]+" ");
        System.out.println("}");
    }
}
```

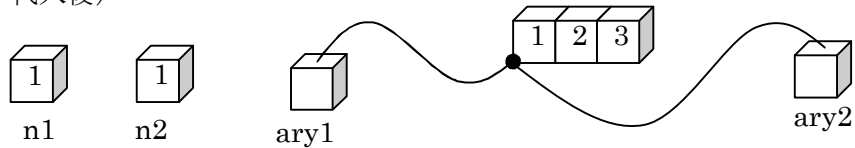
実行画面

```
>java Sample10_4  
n1=1, n2=1  
ary1={1 2 3 }  
ary2={1 2 3 }  
n2 := 2;  
ary2[1] := 4;  
n1=1, n2=2  
ary1={1 4 3 }  
ary2={1 4 3 }  
-- Press any key to exit (Input "c" to continue) --
```

参考：例題 Sample10_4 の変数の振る舞いは次のように図的に理解できる
(変数の宣言時)



(変数へ代入後)



(値の変更後)

