デジタル表現論・第 6 回

劉　雪峰（リュウ　シュウフォン）

2016 年 5 月 16 日

本日の目標

Java プログラミングの基礎

- 配列（復習・関数の値を配列に格納する）
- 文字列
- ファイルの書き込み

配列 (復習)

配列は同じ型のデータの集まりである。

配列の宣言

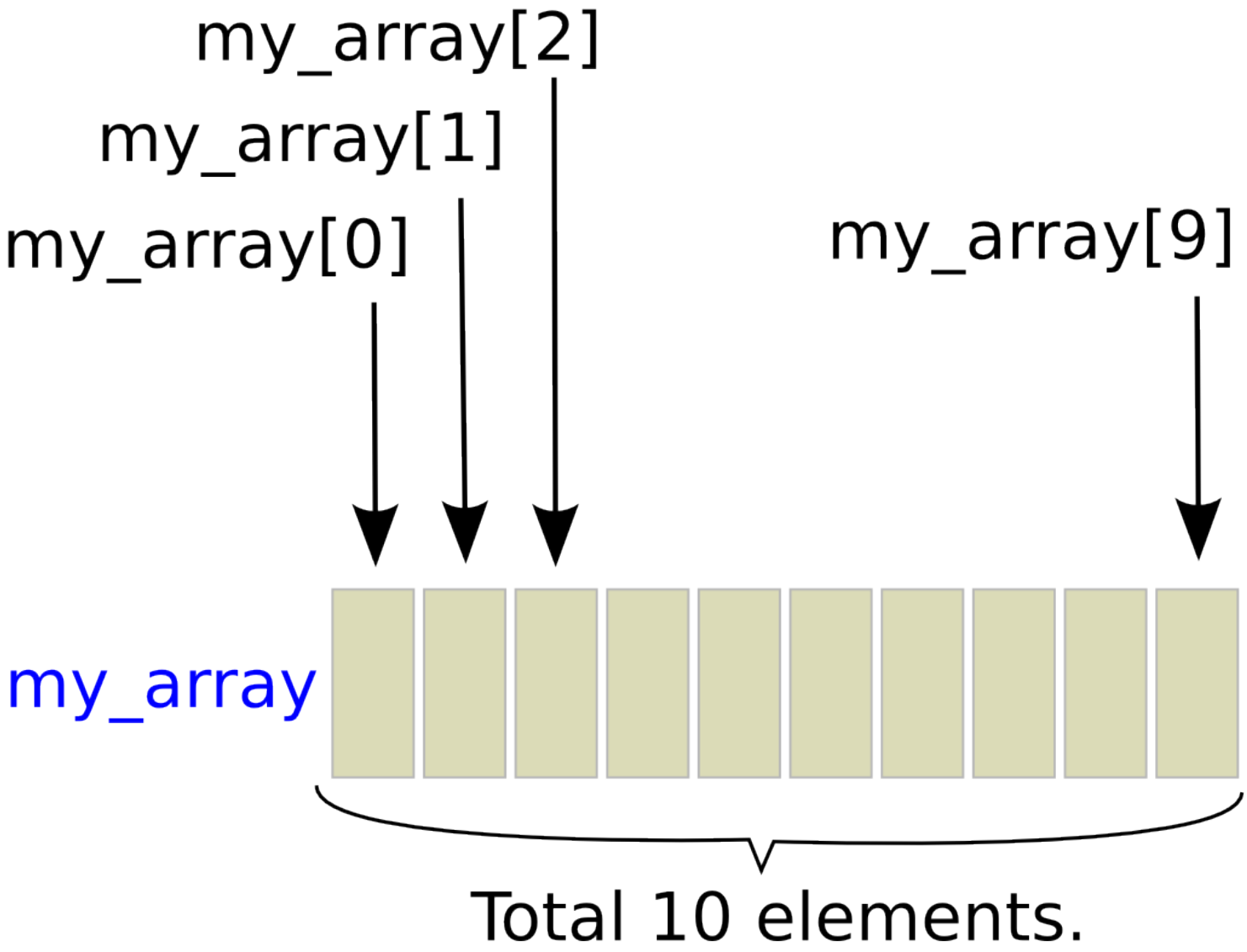
```
int[] my_array = new int[10];
// my_arrayの値（アドレス）の確認
System.out.println(my_array);
```

(*) 変数 my_array の値はメモリにおける配列のアドレスとなる。

上記のコードは以下のものと同じである。

```
int[] my_array;
my_array = new int[10];
System.out.println(my_array);
```

配列のイメージ


my_array[2]
my_array[1]
my_array[0]
my_array[9]
my_array
Total 10 elements.

配列

配列の初期化（方法１）

```
int[] my_array = {1,2,3,4,5};
```

配列の初期化（方法２）

```
int[] my_array = new int[5];

my_array[0]=1;
my_array[1]=2;
my_array[2]=3;
my_array[3]=4;
my_array[4]=5;
```

配列のコピー

以下のコードでは、array_A と array_B は同じ配列に対応している。

```
int [] array_A = {4,5,6,7,8};
int [] array_B;
array_B = array_A;
```

整数のコピーと比較してください。

```
int a = 1, b;
b = a;
b = -1;
```

イメージ：

配列のコピー

以下のコードの出力結果を推測してください。（このコードはホームページからダウンロードできます。）

```
public class Demo6_1 {
  public static void main ( String [] args) {

   int [] array_A = {1,2};  int a=1;
   int [] array_B;   int b;

   array_B = array_A; //array_Aとarray_Bは同じ配列に対応している。
   b = a;

   System.out.println("Before:first value of A:" + array_A[0]);
   System.out.println("Before:first value of B:" + array_B[0]);
   System.out.println("Before:value of a,b:" + a + "," + b);

   array_B[0] = -1; //Bを使って配列の要素の値を更新する。
   b = -1;

   System.out.println("After:first value of A:" + array_A[0]);
   System.out.println("After:first value of B:" + array_B[0]);
   System.out.println("After:value of a,b:" + a + "," + b);
  }
}
```

演習 1 ：配列に関数の値を入れる。

関数 $f(x) = 1 + x^3 + sin(\pi x)$ について、以下の x_i での f の値を浮動小数型配列 FunctionValues に格納する。

$$x_0 = 0, x_1 = 0.1, ..., x_i = i * 0.1, x_{10} = 1.$$

ヒント

数学関数の計算はクラス Math を使うこと:

`Math.sin(x), Math.PI, Math.pow(x,3)` （返却値は `double`）

演習 1 の雛ファイル

```
//ファイル名 : Exe6_1.java
public class Exe6_1 {
  public static double f ( double x) {
    double value;
    //ここに f の計算式のコードを書く。
    return value;
  }
  public static void main ( String [] args) {
    double [] FunctionValues = new double[11];
    for(int i=0; i<11; i++){
      //ここにコードを書く。
    }
    for(int i=0; i<11; i++){ //配列を確認する。
     System.out.println( "["+i+"]:" + FunctionValues[i]);
    }
  } // main
} // Exe 6_1
```

文字列

文字列とは

文字列は文字 (char) の集まりである。クラス String を利用することで、文字列を処理できる。

文字列の宣言（方法１）

```
String greeting = "Hello world!";
```

文字列の宣言（方法２）

```
char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.' };
String greeting = new String(helloArray);
```

クラス String には多くの文字処理のメソッドが用意されている。例えば、特定の文字の探す、置き換えなどの処理。

(*) クラス String の仕様説明 https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

String のメソッドの例：charAt(i)

文字列の i 番目の文字は charAt(i) となる。

```
String hello = "Glad to meet you";
for(int i=0; i< hello.length(); i++){
    System.out.println( hello.charAt(i));
}
```

演習２：文字列の逆順番

入力した文字列を逆順番で表示する。例えば、"ABC" を入力すると、"CBA" を出力する。以下の雛ファイルを参照ください。

```
//ファイル名： Exe6_2.java
import java.util.Scanner;
public class Exe6_2 {
  public static void print_reverse ( String x) {
    //コードを書いてください;
  }
  public static void main ( String [] args) {
   Scanner input = new Scanner(System.in);
   String input_string, output_string;

   System.out.println("Please input a string.");
   input_string = input.next();

   System.out.println("Reverse string is:");
   print_reverse(input_string);
  } // main
} // Exe6_2
```

演習 3：文字列の逆順番 [オプション]（難しい！）

入力した文字列の逆順番を新しい文字列に格納する。

```
import java.util.Scanner;
public class Exe6_3 {
  public static String string_reverse ( String input_string) {
    String new_string = "";
    //コードを書いてください;
    return new_string;
  }
  public static void main ( String [] args) {
   Scanner input = new Scanner(System.in);
   String input_string, output_string;

   System.out.println("Please input a string.");
   input_string = input.next();

   output_string = string_reverse(input_string);
   System.out.println("Reverse string is:");
   System.out.println(output_string);

  } // main
} // Exe6_3
```

ファイルの書き込み

PrintWriter の使用

文字列をファイルに書き込むために、PrintWriter を使用する。

1) IO クラスを利用するために、コードの先頭に IO クラスを import が必要である。
2) 書き込みのエラーを処理するために、main メソードの宣言に「throws IOException」を追加して、以下のようになる。

```
public static void main ( String [] args)  throws
IOException {
 //mainの中身
}
```

3) close() メソッドは書き込みの終了後に呼び出す。

ファイルの書き込み

例

```
import java.io.*;
public class Demo6_2 {
  public static void main( String [] args) throws IOException{

    //コードとは同じフォルダーの下にファイルを書き込む。
    File file = new File("data.txt");

    PrintWriter out = new PrintWriter(file);
    out.println("Data in file:");
    out.println(32);
    out.close();
    System.out.println("Data is saved in data.txt");

  } // main
} // Demo 6_2
```

演習 4：ファイルの書き込み [オプション]

演習 1 中の f の数値を格納している配列 FunctionValues の値をファイル function_values.txt に保存する。

以下は function_values.txt の中身の例である。

```
1.0
2.3
1.2
...
0.23
```