2-2.配列

提出する 評定を受ける

はじめに

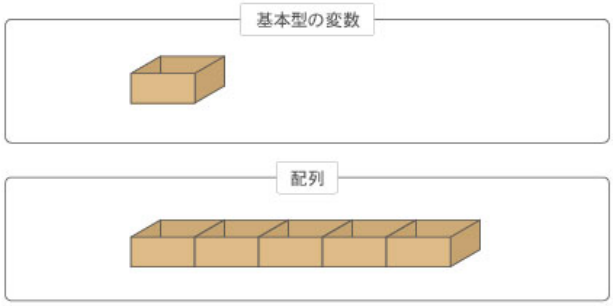
プログラミング言語には、「いくつかの関係のあるデータをグループにして、まとめて1つの変数として扱う」仕組みが用意されており、
以下のような、 **その変数が保持する特定の値や値全体に対して行える機能** を持ち合わせています。

- 取り出し
- 書き換え
- 削除

これらの機能をプログラミング上の呼び方として、「配列操作」と呼びます。

バラバラではなく、ひとまとめにすることで非常に扱いやすくなります。

配列のイメージはこんな感じです。



それではさっそく作り方を見ていきましょう！

Step1: 配列の作り方

配列を作るためには、次の2段階の作業が必要です。

- Phase1: 配列の宣言
- Phase2: 要素代入

Phase1　　配列の宣言
要素の型[] 配列変数名

```
int[] arr;
```

↑ここでint型の要素を代入できる配列変数arrを用意する

Phase2　　要素の代入
配列変数名 = new 要素の型[要素数]

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

↑ここでint型の要素を5個作成してarrに代入し、配列arrの完成。

?

※Phase2の補足
プログラミングで使用する「=(イコール)」は、等しいという意味ではなく、
代入 という意味になります。上記の例でいうと右辺を左辺に代入です。

Phase1とPhase2を合わせて以下のように記述してあげます。

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

↑コレが一連の流れになります！！

ポイント

しかし実は、上記のPhase1とPhase2は同時に行うことが可能です！！

```
int[] arr = new int[5];
```
↑**同時に行うと、コードが1行でまとまってスッキリするので極力こちらを使用するようにしましょう。**

Step2: 初期化

変数の値を取り出すには、必ず値を代入して初期化しなければなりません。
初期化をしていない変数を利用すると、エラーが発生します。

【初期化の方法】

```
int[] arr = new int[5];    //配列arrの宣言

arr[0] = 3; // 1番目（先頭）の要素に 3 を代入
arr[1] = 1; // 2番目の要素に 1 を代入
arr[2] = 6; // 3番目の要素に 6 を代入
arr[3] = 0; // 4番目の要素に 0 を代入
arr[4] = 4; // 5番目の要素に 4 を代入
```

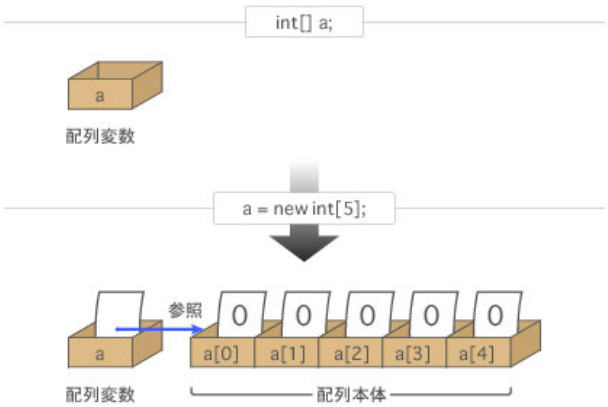
解説

上記の形で初期化することができますが、一行ずつ記述するのが面倒だと思うかもしれません。
しかし、あらかじめ要素の数が決まっている場合は、配列の宣言と初期化を同時に行うことができます！

【宣言と初期化を同時に行う】

```
// 6、10、3、1、8 を要素の値（初期値）とする、要素数 5 の // a という名前のint型配列を宣言。 int[] a = {6, 10, 3, 1, 8};
```
これでだいぶスッキリしましたね！！

今宣言した配列のイメージはこんな感じです。



・ length

Javaの配列には、その配列の要素数を調べることができる length というフィールド（情報）があります。
lengthを使用することにより、プログラマが配列の要素数を別の変数に保持するなどの
管理をする必要がなくなるため、配列がとても利用しやすくなります。

・ length **値の参照方法**

配列の変数に対して「配列変数.length」のように「 . （ドット）」で繋いで記述します。

```
int[] arr = {2, 0, 95};

// 配列arrのlengthフィールドを表示してみる。 →要素数は「3」が表示される

System.out.println("配列arrの要素数…" + arr.length);
```

実行結果

```
配列arrの要素数…3
```

また、配列の各要素を参照したり更新したりするには、配列名の直後に [ ] をつけ、
添え字(インデックス)と呼ばれる数字を指定して要素を特定します。
添え字は **0から順番に数えます。**
配列の先頭は、 1 ではなく 0 であることに注意しましょう。

【添字の指定】

```
// 添え字（インデックス）が 2 の要素の値を表示 System.out.println("arr[2] … " + arr[2]);
```

実行結果

```
arr[2] … 95
```

【添字の指定: 全要素】

```
// 配列の全要素を表示 for (int i = 0; i < arr.length; i++) { System.out.println("arr[" + i + "] … " + arr[i]); }
```

実行結果

```
arr[0] … 2 arr[1] … 0 arr[2] … 95
```

補足

・サンプルコードに出てくる for (int i = 0; i < arr.length; i++){ } について
俗にループ文と呼ばれる書き方のひとつです。
以下簡単に紹介します。

```
for ( /* ① */ ; /* ② */ ; /* ③ */ ){ /* ④ */ }
```

①：変数宣言＆初期値代入
②：ループが終了する条件
③：更新(i++ ならループが2回以降繰り返される度に①の変数に1追加される)
④：ループさせたい処理

・**配列値操作時に使用する用語について**

値操作に関しては難しくはありませんが、「文字や言葉として説明する際」には注意する必要があります。

- **n** 番目の要素値
- 添え字（インデックス） が **n** の要素値

【サンプル】

```
String[] strArray = new String[3]; strArray[1] = "ABC";
```

上記コードを説明（コメント）する場合は、、、

```
// 要素数が3のStringクラスの配列 strArray を定義
String[] strArray = new String[3];
// 2番目の要素値（or 添え字（インデックス）が 1 の要素値）を 文字列ABC で初期化
strArray[1] = "ABC";
```

といったように表現します。

例

ここで、具体例を用いて考えてみましょう。
10人が受けたテストの合計点を求めるプログラムを書きます。

```
// int[]型の配列scoreの宣言と初期化
int score[] = { 80, 70, 90, 60, 65, 35, 80, 50, 45, 85 };

// 合計スコアの変数を宣言し、0で初期化する
int sum = 0;

// for文：ループするのに必要な書き方
for (int i = 0; i < score.length; i++) {
    // 合計スコアに配列scoreの要素を先頭～最後まで順に代入
    sum = sum + score[i];
}

// 文字列と変数sumを「+」で連結して出力
System.out.println("合計点は" + sum + "点です");
```

【実行結果】

```
合計点は660点です
```

提出課題

提出ファイル

・Task2_2.java
2-2/Task2_2.java に課題がありますので、指示に従って記述して下さい。

評定概要

学生から秘匿	No
参加者	80
提出	55
要評定	0