**第6章　配列**

この章では JAVAの配列について説明します。

**配列**

変数には1つの値を格納することができますが、配列は、同じデータ型の複数の値をまとめて格納するために使用されます。

例えば10個の値を格納する場合、変数を使うと変数が10個必要ですが、配列では10個の値を1つの配列にまとめて格納することができます。

Javaの配列を利用するには**(1)扱うデータ型を指定して配列を宣言**し、(2)**必要な要素数を指定して配列を作成する(初期化)する**必要があります。

**配列を宣言する**には、変数の型を　**[ ]** で定義します。

1. String[] cars;
2. int number[];
3. int[] number;

上記の定義は配列の変数を宣言するだけであり、この状態では実体は存在しません。

**配列を初期化する**には、new演算子で初期化します。

1. int number[] = new int[3];
3. int number[] ;
4. number = new int [3];

このとき、number[0], number[1], number[2] にはすべて 0 が格納されています。

**new で配列を初期化した場合は**、その型クラスのデフォルト値で初期化されます。

1行目と3・4行目は、int型の値を3つ格納することのできる配列を宣言しており、どちらの書き方でも意味は同じです。

1行目は 変数宣言と初期化を同時に行っており、3・4行目は宣言と初期化を別に行っています。

new演算子で初期化する以外にも、値を指定して初期化することもできます。

値を指定する場合は、 **{ }** で囲み、カンマ区切りのリストにします。

1. String[] cars = {"Toyota", "Honda", "Subaru", "Nissan"};
3. String[] cars;
4. cars = {"Toyota", "Honda", "Subaru", "Nissan"};

**1.配列の要素にアクセスする**

インデックス番号を **[　]**内に参照して配列要素にアクセスします。

1. String[] cars = {"Toyota", "Honda", " Subaru ", "Nissan"};
2. System.out.println(cars[0]);

注：配列インデックスは0から始まります。[0]が最初の要素です。[1]は2番目の要素などです。

**実行結果**

Toyota

**2.配列の長さ**

配列の要素数を調べるには、次のlengthプロパティを使用します。

1. String[] cars = {"Toyota", "Honda", "Subaru", "Nissan"};
2. System.out.println(cars.length);

**実行結果**

3

上のインデックス番号について説明したように、配列インデックスは0から始まるため、配列cars は｛ [0], [1], [2], [3] }　と数えるので、出力結果は3となります。

**3.配列をループする**

配列を使用して処理を繰り返す場合、length プロパティを使用して繰り返しの実行回数を指定できます。

1. String[] cars = {"Toyota", "Honda", "Subaru", "Nissan"};
2. for (int i = 0; i< cars.length; i++){
3. System.out.println(cars[i]);
4. }

**実行結果**

Toyota

Honda

Subaru

Nissan

この例では2行目のfor文の[条件式](※1)に 「 i　が配列carsの要素数以下」と指定しているため i が3になるまで3行目の出力処理を繰り返します。

(※1)　5章　3.for文参照。

また、以下のように書いても同じ意味になります。

1. String[] cars = {"Toyota", "Honda", "Subaru", "Nissan"};
2. for (String i : cras){
3. System.out.println(i);
4. }

**4.多次元配列（「配列」の配列）**

多次元配列は、2つ以上の配列を含む配列です。

2次元配列を作成するには、各配列を **{ }** の内に追加します。

1. int[][] myNumbers = {{1, 2, 3, 4},{5, 6, 7} };
2. int x = myNumbers[1][2];
3. System.out.println(x);

**実行結果**

7

ここでは、myNumbersという配列は｛1, 2, 3, 4,} という配列と { 5, 6, 7 }という2つの配列が要素になった配列です。

2行目のmyNumbers[1][2]の部分は、[1]は配列myNumbersのインデックスナンバー[1]を表しており、具体的には{5, 6, 7}の配列のことを指しています。[2]は{5, 6, 7}の配列のインデックスナンバー[2]を表しているため7を指しています。

そのため int x には7が代入され、3行目の出力指示により 7 が出力されます。

例

[JSample6\_1.java]

1. package JSample;
2. public class JSample6\_1{
3. public static void main(Stirng[] atgs){
4. int[][] myNumbers = {{1, 2, 3, 4},{5, 6, 7} };
5. for (int i =0; i < myNumbers.length; ++i){
6. System.out.println("行ごとに繰り返し");
7. for (int j =0; j < myNumbers[i].length; ++j){
8. System.out.println("列の要素数は" + myNumbers[i].length);
9. }
10. }
11. }
12. }

**実行結果**

行ごとに繰り返し

列の要素数は4

1

列の要素数は4

2

列の要素数は4

3

列の要素数は4

4

行ごとに繰り返し

列の要素数は3

5

列の要素数は3

6

列の要素数は3

7

練習

問題1[JEx6\_1.java]

int[] cars = {1,2,3,4,5,6,7,8,9,10};

上記の配列から偶数のみを出力してください。

問題2[JEx6\_2.java]

配列を作成して、下記の九九表になるように出力してください。

1

2 4

3 6 9

4 8 12 16

5 10 15 20 25

6 12 18 24 30 36

7 14 21 28 35 42 49

8 16 24 32 40 48 56 64

9 18 27 36 45 54 63 72 81

問題3[JEx6\_3.java]

String[] cars = {"Toyota", "Honda", "Subaru", "Nissan"};

上記の配列を作って、下記のように出力してください。

Toyota

Honda

Subaru

Nissan