＜コンストラクタ＞

Java はインスタンス化の際「コンストラクタ」というメソッド（窓口）を呼び出します。
どのようにやっていくかは以下の演習で確認します。

（演習①）

Java プロジェクト「Person2」を作成しましょう。
以下の Person クラスと Test クラスをプログラミングしてみましょう。

```
public class Person{
   public String name = null;
   public int age = 0;
}
```

```
public class Test{
   public static void main(String[] args){
      Person taro= new Person();
   }
}
```

（演習②）

Person クラスにコンストラクタ①②を追加してみましょう。

```
public class Person{
   public String name = null;
   public int age = 0;

   public Person( ){}

   public Person(String name, int age){
      this.name = name;
      this.age = age;
   }
}
```

コンストラクタ①

コンストラクタ②

演習①は前章のインスタンス化ですが、前章では「Person taro= new Person();」の「( )」については説明しませんでした。実は Test クラスのこの( )が Person クラスのコンストラクタと連動することで、インスタンス化の際のさまざまな初期値（例えば age は 20 にするなど）をあらかじめプログラムすることができます。①の型をデフォルトコンストラクタといい、他の型のコンストラクタを宣言しない場合は書く必要はありません。

（演習③）

Test クラスに以下のプログラムを追加してみましょう。

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

Person taro = new Person();
taro.name = “taro”;
taro.age = 18;
System.out.println(taro.name);
System.out.println(taro.age);

Person jiro = new Person(“jiro”, 20);
System.out.println(jiro.name);
System.out.println(jiro.age);
   }
}
```

コンストラクタ①を使ったインスタンス化です。
※前章のインスタンス化と同じ

下は、コンストラクタ②を使ったインスタンス化です。
コンストラクタ①を使った時よりも 2 行分節約して書くことができます。

```
public class Person{
   public String name = null;
   public int age = 0;

   public Person(){}

   public Person(String name, int age){
      this.name = name;
      this.age = age;
   }
}
```

this.とはこのクラスのインスタンスのフィールド（変数）であることを表す。

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

Person jiro = new Person(“jiro”, 20);
System.out.println (jiro.name);
System.out.println (jiro.age);
   }
}
```

(“jiro”,20)に合う型を Person クラスで作成する。今回だと(String name,int age)を作成。

Test クラスの new Person(“jiro”,20)が Person クラスの赤い枠の型と合うので jiro 20 と表示することが出来ます。また、この型以外の型を作成することが可能です。例えば年齢だけは書きたくない、名前だけは教えたくないなどの条件に対して、それに合った型を定義することができます。つまり、引数の型や数、順序が違う複数のコンストラクタの作成が可能です。このように、複数のコンストラクタを定義することを多重定義（オーバーロード）といいます。実際に次の演習で確認してみましょう。

（演習④）

Person クラスにコンストラクタ③～⑤を追加してみましょう。

```
public Person(){}
```
コンストラクタ①

```
public Person(String name, int age){
    this.name = name;
    this.age = age;
}
```
コンストラクタ②

```
public Person(String name){
    this.name = name;
    this.age = 0;
}
```
コンストラクタ③
（あたらしく追加）

```
public Person(int age){
    this.name = “名前なし”;
    this.age = age;
}
```
コンストラクタ④
（あたらしく追加）

```
public Person(int age, String name){
    this.name = name;
    this.age = age;
}
```
コンストラクタ⑤
（あたらしく追加）

（演習⑤）

Test クラスを使って以下の条件でプログラムを実行してみましょう。

コンストラクタ③を使ってインスタンス化・・・ saburo と 0 を表示しましょう。
コンストラクタ④を使ってインスタンス化・・・名前なしと 25 を表示しましょう。
コンストラクタ⑤を使ってインスタンス化・・・ hanako と 17 を表示しましょう。
