Java プログラミング入門

— メソッド —

早稲田大学

例題

キーボードから身長（cm）と体重（kg）を入力すると標準体重を表示し，標準体重との差（体重 − 標準体重）に応じて以下のようなメッセージを表示させなさい：

体重 − 標準体重	メッセージ
-10 未満	`Underweight`
-10 以上 10 以下	`Normal`
10 超過	`Overweight`

ただし，標準体重の計算式は次のとおりとする：

$$\text{標準体重}\,(\mathrm{kg}) = \text{身長}\,(\mathrm{m})^2 \times 22$$

メソッド

これまでのプログラム

```
public class ClassName {
  public static void main(String[] args) {
        プログラム本体
  }
}
```

- すべてのプログラムを “プログラム本体” の部分に記述
- 上記の “public static void main(String[] args)” もメソッドの一つ（Main メソッド）
- 一般にプログラミングでは，プログラムをあるまとまりを持った部分に細分化し，効率化を図る
- Java では，プログラムの細分化を “メソッド” で実現する
- 細分化されたプログラムの単位を Java ではメソッドと呼ぶ

メソッドの書き方

```
public class ClassName {
   public static void main(String[] args) {
           :
     c = method1(a, b);
           :
     method2(x);
           :
   }

   public static int method1(int a, int b) {
     ....
   }

   public static void method2(double x) {
     ....
   }
}
```

メソッドの書き方

```
public class ClassName {
   public static void main(String[] args) {
            :
     c = method1(a, b);
            :
     method2(x);
            :
   }

   public static int method1(int a, int b) {
     ....
   }

   public static void method2(double x) {
     ....
   }
}
```

メソッドの書き方

```
public class ClassName {
   public static void main(String[] args) {
          :
     c = method1(a, b);
          :
     method2(x);
          :
   }

   public static int method1(int a, int b) {
     ....
   }

   public static void method2(double x) {
     ....
   }
}
```

メソッドの書き方

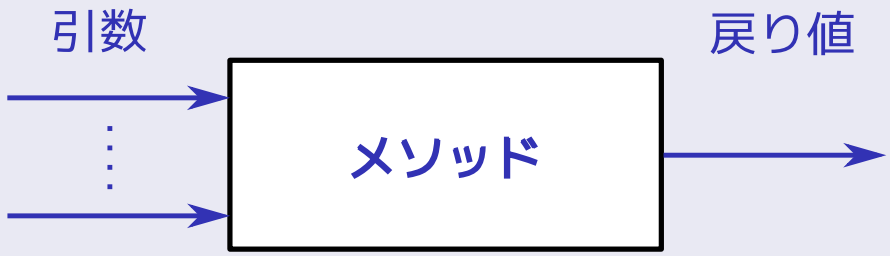
```
public class ClassName {
   public static void main(String[] args) {
            :
     c = method1(a, b);
            :
     method2(x);
            :
   }

   public static int method1(int a, int b) {
     ....
   }

   public static void method2(double x) {
     ....
   }
}
```

メソッドの概要

- メソッドは，プログラムの細分化，効率化を目的として自由に作成できる
- メソッドはクラスの中で定義する
- 例では，main メソッドの他に method1 と method2 という 2 つのメソッドを定義している
- これら 2 つのメソッドを main メソッドから呼び出している

引数と戻り値

- メソッドは引数と戻り値を持つ
- メソッドに対してデータを与える部分を引数という
- メソッドでの処理の結果として呼び出し元に返すデータを戻り値という



メソッドのイメージ

メソッドの作成（1）

```
修飾子 戻り値の型 メソッド名（引数）{
    メソッド本体
}
```

修飾子 必要に応じて記述．今は “public static” とする

戻り値の型 メソッドでの処理の結果として返すデータの型

メソッド名 メソッドの名前．内容が分かり易いようにつける

引数 メソッドにデータを与える部分の記述．データの型名と変数名を記述する．引数は複数個指定できる

メソッドの作成（2）

```
return 戻り値;
```

- “戻り値” で記述された部分が，メソッドの戻り値となり、そのメソッドの呼び出し元に返すことができる

戻り値の例

```
public static double add(double x,double y){
  double z;
  z = x + y;
  return z;
}
```

メソッドの作成（3）

「身長のデータを受け取り標準体重を計算するメソッド」

を作成する.

- メソッド名は “calcStdWeight” とする
- 身長のデータを受け取るので，double 型の引数 height を導入する
- 標準体重を戻り値とする。戻り値は double 型となる

標準体重の計算法

$$\text{標準体重}\,(\mathrm{kg}) = \text{身長}\,(\mathrm{m})^2 \times 22$$

メソッドの作成（4）

メソッド名：calcStdWeight

```
public static double calcStdWeight(double height) {
  double weight;
  height = height / 100;
  weight = height * height * 22;
  return weight;
}
```

メソッドの書き方いろいろ

`public static int sampleMethod1(int a, int b)`
— 2 つの int 型の引数を受け取り，
int 型の戻り値を返すメソッド

`public static double sampleMethod2(int a, double x)`
— int 型と double 型の引数を 1 つずつ受け取り，
double 型の戻り値を返すメソッド

`public static boolean sampleMethod3(int a, int b, int c)`
— 3 つの int 型の引数を受け取り，
boolean 型（true または false）の戻り値を返すメソッド

`public static void sampleMethod4(double x)`
— double 型の引数を 1 つ受け取り，戻り値のないメソッド

`public static double sampleMethod5()`
— 引数を受け取らず，double 型の戻り値を返すメソッド

メソッドの呼び出し（1）

calcStdWeight メソッドを呼び出す

```
sw = calcStdWeight(170.0);
```

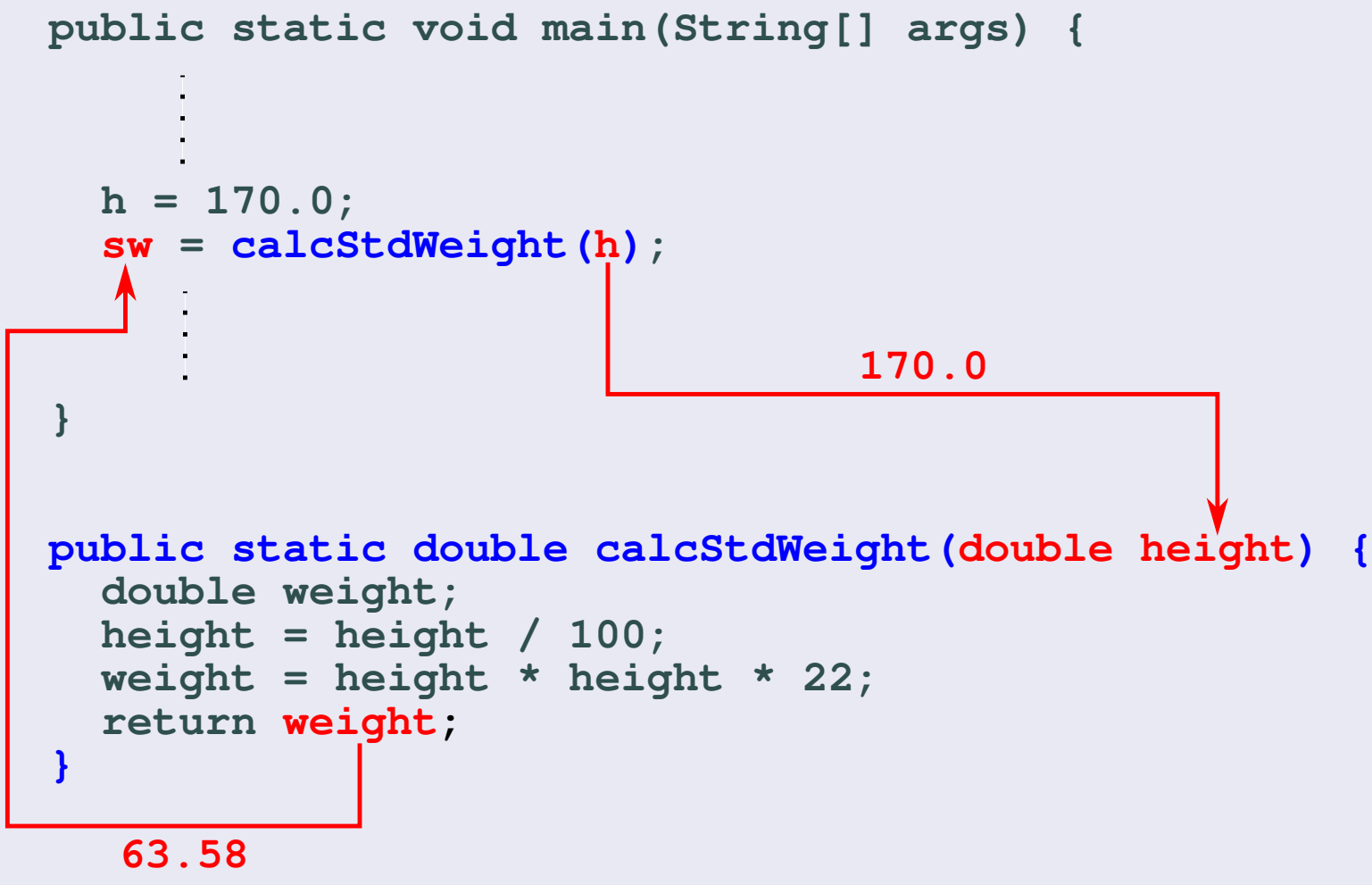
- 身長 170cm で計算する
- calcStdWeight(170.0) で 170cm のときの標準体重が計算される
- 結果を変数 sw に代入する

```
h = 170.0;
sw = calcStdWeight(h);
```

- 引数に変数を渡すこともできる

メソッドの呼び出し（2）

メソッド呼び出しの流れ



引数や戻り値を持たないメソッド（1）

“例題” において....

メッセージを表示する部分について，

「体重と標準体重のデータを受け取り，メッセージを表示するメソッド」

を作成する．

- メソッド名は “printMessage” とする
- 体重，標準体重のデータを受け取るので，double 型の引数 2 つを導入
- 結果としてメッセージを表示．戻り値は，なし

引数や戻り値を持たないメソッド（2）

メソッド名：printMessage

```
public static void printMessage(double weight, double stdWeight) {
  double diff = weight - stdWeight;
  if (diff < -10) {
    System.out.println("Underweight.");
  } else if (diff < 10) {
    System.out.println("Normal.");
  } else {
    System.out.println("Overweight.");
  }
}
```

引数や戻り値を持たないメソッド（3）

戻り値がないメソッドの呼び出し

```
h = 170.0;
w = 70.0
sw = calcStdWeight(h);    — 戻り値を変数 sw に代入
printMessage(w, sw);      — 代入操作はいらない
```

- 戻り値を持たない場合，代入操作は必要ない

ローカル変数

```
public static void main(String[] args) {
  double x, y;
  ....
  y = method1(x);
  z = method1(y);                    — エラー×
  ....
}

public static double method1(double t)
  double x, y, z;
  ...
}
```

- z は method1 の中だけで使える変数
- 宣言されたメソッド外で使用するとエラー
- 別のメソッド内で同じ変数名が宣言された場合，それらは別の変数として扱われる

引数として配列を取るメソッド

- int 型の配列を受け取り，配列内の最大値を返すメソッド

メソッド名： maxOfArray

```
public static int maxOfArray(int[] a) {
  int i, max;

  max = a[0];
  for (i=1; i<a.length; i++) {
    if (a[i] > max) max = a[i];
  }
  return max;
}
```

- このメソッドの呼び出し
- int 型配列 arr の各要素にデータが格納されているとすると

```
max = maxOfArray(arr);
```

戻り値として配列を返すメソッド

- すべての要素が 1.0 である配列を生成するメソッド

メソッド名：ones

```
public static double[] ones(int n) {
  int i;
  double[] x = new double[n];

  for (i=0; i<x.length; i++) {
    x[i] = 1.0;
  }
  return x;
}
```

- 配列の生成にはその長さの指定が必要なので，これを引数 n として受け取る
- このメソッドの呼び出し

```
double[] x;
x = ones(5);
```

キーボードからの入力を行うメソッド（1）

- キーボードから int 型のデータ入力を行うメソッド

メソッド名：inputInt

```
public static int inputInt() {
 Scanner sc = new Scanner(System.in);
 int i = 0;

 i= sc.nextInt();
 return i;
}
```

- このメソッドの呼び出し

```
a = inputInt();
```

キーボードからの入力を行うメソッド（2）

- キーボードから double 型のデータ入力を行うメソッド

メソッド名：inputDouble

```
public static double inputDouble() {
  Scanner sc = new Scanner(System.in);
  double d = 0.0;

  d = sc.nextDouble();
  return d;
}
```

- このメソッドの呼び出し

```
x = inputDouble();
```

キーボードからの入力を行うメソッド（3）

SampleInput.java

```
import java.util.Scanner;          ←― この一行を必ず入れる

public class SampleInput {
  public static void main(String[] args) {
    System.out.print("Input an int : ");
    a = inputInt();
    System.out.print("Input a double : ");
    x = inputDouble();
  }
  public static int inputInt() {
    ....
  }
  public static double inputDouble() {
    ....
  }
}
```

例題のプログラム例（大枠）

Weight3.java

```
import java.util.Scanner;

public class Weight3 {
  public static void main(String[] args) {
    ....
  }

  public static double calcStdWeight(double height) {
    ....
  }

  public static void printMessage(double weight, double stdWeight) {
    ....
  }

  public static double inputDouble() {
    ....
  }
}
```

例題のプログラム例（Main メソッド）

```
public static void main(String[] args) {
    double h, sw, w;
    System.out.print("Input height (cm) : ")
    h = inputDouble();
    System.out.print("Input weight (kg) : ")
    w = inputDouble();
    sw = calcStdWeight(h);
    System.out.println("Standard weight : " + sw + "kg");
    printMessage(w, sw);
}
```

calcStdWeight メソッド
— 身長（double 型）を受け取り，標準体重（double 型）を返す

printMessage メソッド
— 体重（double 型）と標準体重（double 型）を受け取り，メッセージを表示

inputDouble メソッド
— キーボードから double 型の値を入力させ，その値を返す

calcStdWeight メソッド

```
public static double calcStdWeight(double height) {
  double stdWeight;

  height = height / 100;
  stdWeight = height * height * 22;
  return stdWeight;
}
```

printMessage メソッド

```
public static void printMessage(double weight, double stdWeight) {
  double diff;

  diff = weight - stdWeight;
  if (diff < -10) {
    System.out.println("Underweight.");
  } else if (diff < 10) {
    System.out.println("Normal.");
  } else {
    System.out.println("Overweight.");
  }
}
```

inputDouble メソッド

```
public static double inputDouble() {
  Scanner sc = new Scanner(System.in);
  double d = 0.0;

  d = sc.nextDouble();
  return d;
}
```