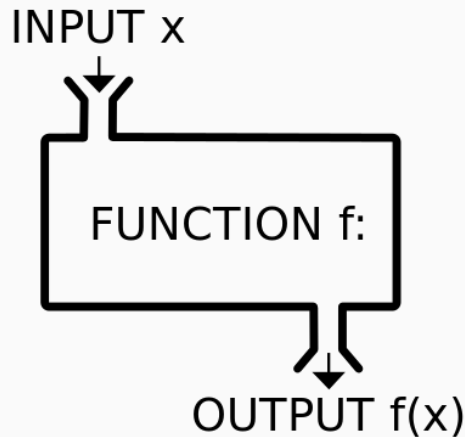


제5장 메소드 (method)

메소드

코드가 길어지거나, 같은 코드를 반복해야 할 때,
코드를 부품화하여 분리해 보기 좋은 코드로 만들 수 있다.
코드의 부품화 하는 방법 중 한가지가 메소드이다.

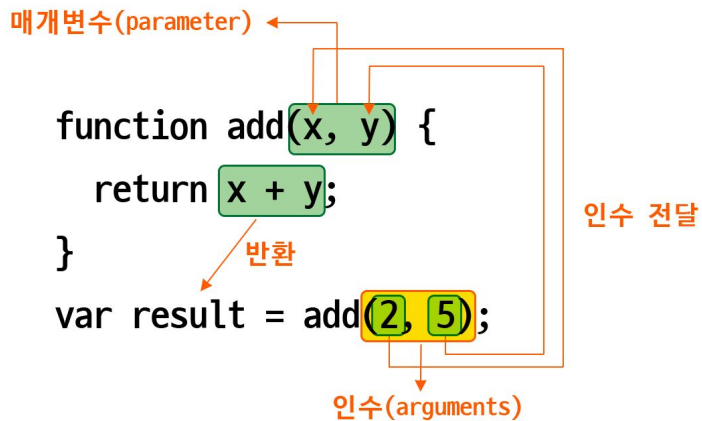


```
1  public class Main {  
2      public static void hello() {  
3          System.out.println("안녕하세요");  
4      }  
5  }
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("메소드를 호출합니다");  
4         hello();  
5         System.out.println("메소드 호출이 종료되었습니다");  
6     }  
7  
8     public static void hello() {  
9         System.out.println("안녕하세요");  
10    }  
11 }
```

```
1 public class Main {  
2     public static void methodA() {  
3         System.out.println("methodA");  
4         methodB();  
5     }  
6  
7     private static void methodB() {  
8         System.out.println("methodB");  
9     }  
10  
11     public static void main(String[] args) {  
12         methodA();  
13     }  
14 }
```

인수 (argument) 와 인자 (parameter)



```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("메소드를 호출합니다");  
4         hello("준석");  
5         hello("제임스 고슬링");  
6         hello("순다 피차이");  
7         System.out.println("메소드 호출을 종료합니다");  
8     }  
9  
10    private static void hello(String name) {  
11        System.out.println(name + "씨 안녕하세요");  
12    }  
13 }
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         add(100, 20);  
4         add(200, 50);  
5     }  
6  
7     private static void add(int x, int y) {  
8         int ans = x + y;  
9         System.out.println(x + "+" + y + "=" + ans);  
10    }  
11 }
```



```
1 public class Main {  
2     public static void main(String[] args) {  
3         int x = 100;  
4         int y = 10;  
5         add();  
6     }  
7  
8     private static void add() {  
9         int ans = x + y;  
10        System.out.println(x + " + " + y + " = " + ans);  
11    }  
12 }
```

반환 값 (return)

메소드 호출시 결과 값을 돌려주는 것을 **값을 돌려준다**.

이 값을 **반환 값** (return) 이라고 한다.

```
1 public class Main {  
2     public static void main(String[] args) {  
3         int ans = add(100, 10);  
4         System.out.println("100 + 10 = " + ans);  
5     }  
6  
7     private static int add(int x, int y) {  
8         int ans = x + y;  
9         return ans;  
10    }  
11 }
```

```
1 public class Main {  
2  
3     private static int add(int x, int y) {  
4         int ans = x + y;  
5         return ans;  
6     }  
7  
8     public static void main(String[] args) {  
9         System.out.println(add(add(10, 20), add(30, 40)));  
10    }  
11 }
```

```
public static int sample() {  
    return 1;  
    int x = 10;  
}
```

오버로드 (overload)

같은 이름의 메소드를 여러개 정의 하는 것

```
1 public class Main {  
2     // 첫번째 add 메소드  
3     public static int add(int x, int y) {  
4         return x + y;  
5     }  
6     // 두번째 add 메소드  
7     public static double add(double x, double y) {  
8         return x + y;  
9     }  
10    // 세번째 add 메소드  
11    public static String add(String x, String y) {  
12        return x + y;  
13    }  
14  
15    public static void main(String[] args) {  
16        System.out.println(add(10, 20));  
17        System.out.println(add(3.5, 2.7));  
18        System.out.println(add("Hello", "World"));  
19    }  
20 }
```

```
1 public class Main {  
2     // 첫번째 add 메소드  
3     public static int add(int x, int y) {  
4         return x + y;  
5     }  
6     // 두번째 add 메소드  
7     public static int add(int x, int y, int z) {  
8         return x + y + z;  
9     }  
10  
11     public static void main(String[] args) {  
12         System.out.println("10 + 20 = " + add(10, 20));  
13         System.out.println("10 + 20 + 30 = " + add(10, 20, 30));  
14     }  
15 }
```



```
1 public class Main {  
2     // int형 배열을 받아 모든 요소를 출력  
3     public static void printArray(int[] array) {  
4         for (int element : array) {  
5             System.out.println(element);  
6         }  
7     }  
8  
9     public static void main(String[] args) {  
10        int[] array = { 1, 2, 3 };  
11        printArray(array);  
12    }  
13 }
```

```
1 public class Main {  
2     // int형 배열을 받아 배열내의 요소 전부 1을 더하는 메소드  
3     public static void incArray(int[] array) {  
4         for (int i = 0; i < array.length; i++) {  
5             array[i]++;  
6         }  
7     } // 계산 결과를 리턴하지 않음  
8  
9     public static void main(String[] args) {  
10        int[] array = { 1, 2, 3 };  
11        incArray(array);  
12  
13        for (int i : array) {  
14            System.out.println(i);  
15        }  
16    }  
17 }
```

```
1 public class Main {  
2     public static int[] makeArray(int size) {  
3         int[] newArray = new int[size];  
4         for (int i = 0; i < newArray.length; i++) {  
5             newArray[i] = i;  
6         }  
7         return newArray;  
8     }  
9  
10    public static void main(String[] args) {  
11        int[] array = makeArray(3);  
12        for (int i : array) {  
13            System.out.println(i);  
14        }  
15    }  
16 }
```

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

다음 사양을 참고하여 메소드 “introduceOneself” 를 정의하시오

메소드명	introduceOneself
리턴 값	없음
인수 항목	없음
처리내용	이름(문자열), 나이(정수), 키(부동소수점), 성별(문자열)를 대입하는 변수를 선언하고 값을 대입한다. 변수를 이용하여 자기소개를 표시한다. 표시할 데이터의 내용이나 방법은 자유롭게 정한다.

다음 사양을 참고하여 메소드 “email” 를 정의하시오

메소드명	email
리턴 값	없음
인수(parameter) 항목	메일의 제목(String title) 메일을 받는 주소 (String address) 메일 본문 (String text)
처리내용	아래의 형식으로 표시를 한다. (빨간색 문자의 부분은 인수를 사용한다) 메일을 받는 주소 에 아래의 메일을 송신한다. 제목 : 메일의 제목 본문 : 메일 본문

다음 사양을 참고하여 연습문제 5-2 의 코드에 메소드 “email” 를
오버로드하여 main메소드에서 호출하시오

메소드명	email
리턴 값	없음
인수(parameter) 항목	<p>메일을 받는 주소 (String address)</p> <p>메일 본문 (String text)</p>
처리내용	<p>아래의 형식으로 표시를 한다. (빨간색 문자의 부분은 인수를 사용한다)</p> <p>메일을 받는 주소 에 아래의 메일을 송신한다.</p> <p>제목 : 제목 없음</p> <p>본문 : 메일 본문</p>

다음 사양을 참고하여 메소드 “`calcTriangleArea`” 와 “`calcCircleArea`” 를 작성하시오.

적당한 값을 인수로 넘겨 올바른 면적이 표시되는지 확인하시오.

메소드명	<code>calcTriangleArea</code>
리턴 값	삼각형의 면적 (<code>double</code>)
인수 항목	삼각형의 밑변의 길이. 단위는 <code>cm (double bottom)</code> 삼각형의 높이. 단위는 <code>cm (double height)</code>
처리내용	인수를 사용하여 면적을 구하고, 그것을 반환 함

메소드명	<code>calcCircleArea</code>
리턴 값	원의 면적 (<code>double</code>)
인수 항목	원의 반지름. 단위는 <code>cm (double radius)</code>
처리내용	인수를 사용하여 면적을 구하고, 그것을 반환 함