Ch5
메서드

객체지향프로그래밍(기본) 2019

경상대학교 항공우주및소프트웨어공학전공

5.1 메서드의 구조

메서드는 함수라고도 하며 어떠한 동작을 실행하기 위해 만들어졌다.
메서드 내부에는 조건문, 반복문과 실행문등 다양한 문장으로 이루어 질 수 있다.
자바에서 메서드는 단독으로 사용될 수 없으며 반드시 객체나 클래스의 일부로 호출이 가능하나 static 키워드를 이용하며 객체의 생성 없이 메서드를 사용 할 수 있다.

반환 값이 있는 메서드

```
1  public class Example {
2    public static void main(String[] args) {
3        int sum;
4
5        sum = add(10,20);
6        System.out.println(sum);
7      }
8
9      static int add(int x, int y)
10     {
11       return x+y;
12     }
13
14 }
```

호출

반환되는 데이터의 타입

결과

```
30
```

반환 값이 없는 메서드

```
1  public class Example {
2    public static void main(String[] args) {
3      myPrint("hello World!!");
4    }
5
6    static void myPrint(String str)
7    {
8      System.out.println(str);
9    }
10 }
```

호출

결과

```
hello World!!
```

5.2 메서드와 매개변수

메서드에서 사용하는 것을 매개변수 호출하는 곳에서 사용하는 것을 인자라고 한다. 매개 변수가 없는 메서드도 존재한다.

```
1  public class Example {
2    public static void main(String[] args) {
3        int number;
4
5        number = getTen();
6        System.out.println(number);
7      }
8
9      static int getTen()
10     {
11       return 10;
12     }
13 }
```

반환 (return)

호출 (getTen())

결과

```
10
```

인자와 매개변수간의 타입 일치

메서드를 호출할 때 매개변수의 형태와 동일한 타입의 인자를 사용해야만 한다.

```
1  public class Example {
2    public static void main(String[] args) {
3        int number;
4
5        number = getTen();
6        myPrint(number);
7      }
8
9      static void myPrint(int num)
10     {
11       System.out.println(num);
12     }
13
14     static int getTen()
15     {
16       return 10;
17     }
18 }
```

데이터 타입 일치

결과

```
10
```

5.3 메서드와 변수의 범위

변수는 라이프 사이클과 범위가 있다. 라이프 사이클은 변수가 메모리에 저장되어 있는 동안의 시간을 뜻하며 범위는 변수가 사용 가능한 범위를 의미한다.

자바는 동일한 메서드 내에서 같은 이름을 가진 변수의 선언을 허용하지 않지만 다른 메서드 간의 동일한 이름을 가진 변수의 존재는 허용한다.

이러한 상황에서 변수의 이름이 같다고 하더라도 두 개의 변수는 같은 이름을 가진 변수 일뿐 동일한 변수는 아니다.

다른 메서드 내의 동일 이름 변수

```java
public class Example {
  public static void main(String[] args) {
      int number;

      number = 100;
      anotherNumber();

      System.out.println("maun 메서드 출력");
      System.out.println(number);
    }

    static void anotherNumber()
    {
      int number;

      number = 200;

      System.out.println("anotherNumber 메서드 출력");
      System.out.println(number);
    }
}
```

동일한 이름을 가진 변수 선언

결과

```
anotherNumber 메서드 출력
200
main 메서드 출력
100
```

인자와 매개변수 간의 관계

```
1  public class Example {
2    public static void main(String[] args) {
3        int number = 100;
4
5        addTen(number);
6
7        System.out.println("maun 메서드 출력");
8        System.out.println(number);
9      }
10
11     static void addTen(int number)
12     {
13       number += 10;
14
15       System.out.println("addTen 메서드 출력");
16       System.out.println(number);
17     }
18 }
```

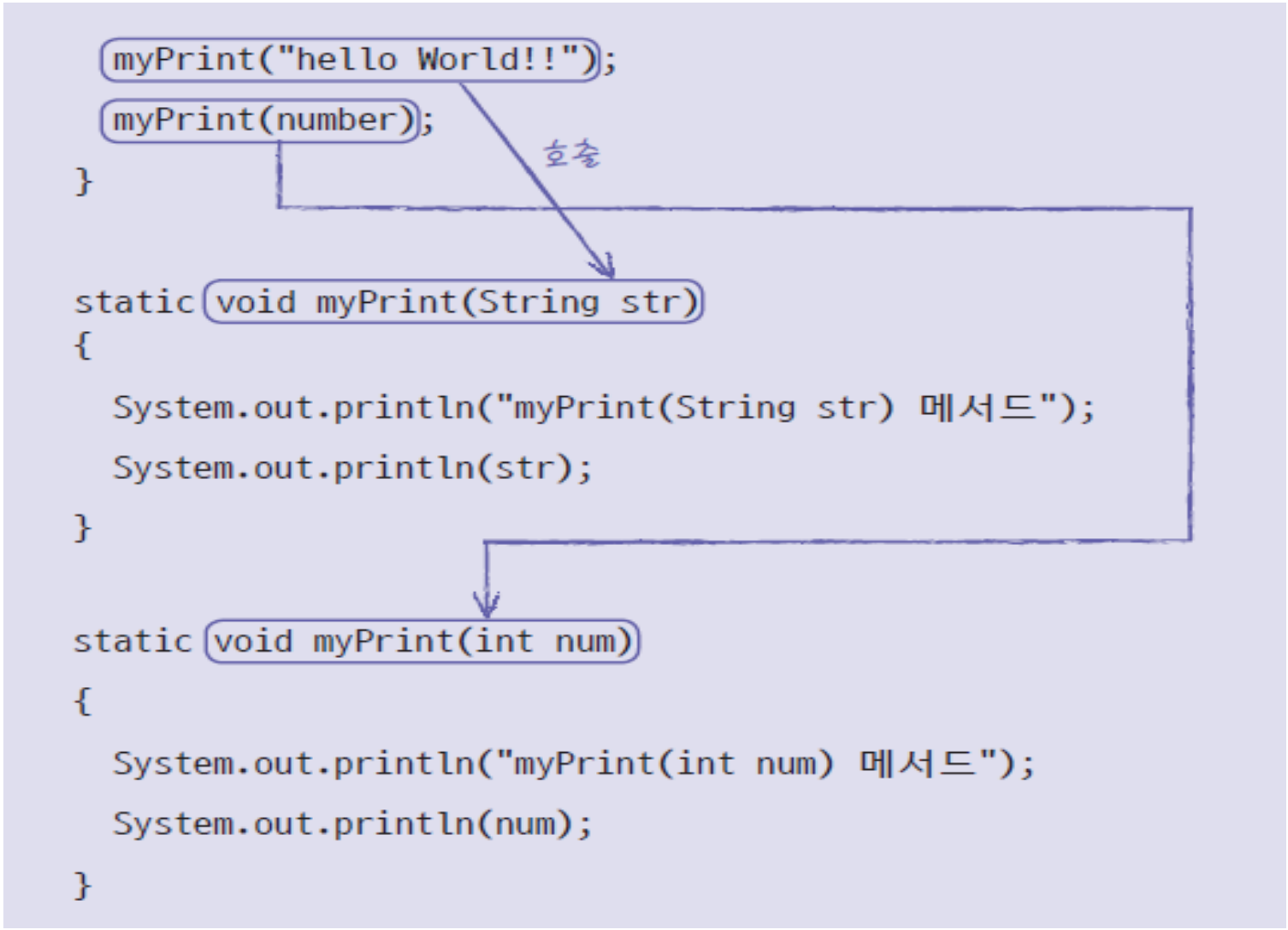
매개변수는 인자의 값만을 복사할 뿐 인자 자체가 전달되는 것은 아니다.

메서드에서 매개변수의 값이 변경 되더라도 인자 자체의 값은 변경되지 않는다.

결과

```
addTen 메서드 출력
110
main 메서드 출력
100
```

5.3 메서드 오버로드



프로그램 내에 일정한 조건을 만족시킬 경우 동일한 이름을 가진 메서드를 허용한다.

매개변수의 개수나 매개변수의 데이터 타입이 다를 경우 동일한 이름을 가지는 메서드를 허용하는데 이것이 메서드 오버로드다.

5.4 순환 메서드(재귀함수)

순환 메서드는 재귀함수라고도 불리며 메서드의 반환 값에 메서드 자신을 호출하는 구조로 선언된 함수를 말한다.

순환 메서드를 위해 메서드는 메서드 자신반환 하는 부분과 반복적인 호출이 멈추고 일정한 값을 출력하는 부분으로 구성된다.

순환 메서드를 사용하면 큰 문제를 작은 문제로 나누어 생각 할 수 있기 때문에 복잡한 알고리즘의 구현에 많이 사용된다

팩토리얼

팩토리얼이란 1부터 정해진 숫자까지의 곱한 값을 말하며 1부터 n까지 차례대로 곱한 값을 n!로 표현하며 0!은 1이다.

$$factorial(n) = \begin{cases} 1 & \text{if } n = 1 \text{ or } 0 \\ n \times factorial(n-1) & otherwise \end{cases}$$

```
static int factorial(int num)
{
  if(num == 1 || num == 0 )  return 1;   ← 값을 반환
  return num * factorial(num-1);   ← factorial 메소드의 호출
}
```

하노이 탑

하노이 탑은 수학적인 문제를 만들기 위해 생각해낸 가상의 이야기로 인도의 한 사원에 세 개의 다이아몬드 기둥이 있고 그 중 하나에 64개의 금 원판이 있어 이 원판을 다른 기둥으로 옮기면 세상이 멸망한다고 한다.

제일 밑에 가장 큰 원판이 있고 크기가 커지는 순서로 아래에 쌓여있다. 원판을 움직일 때 다음과 같은 두 가지 규칙을 따른다.

1) 한 번에 하나의 원판만 움직일 수 있다.
2) 크기가 큰 원판은 작은 원판 밑에 있어야 한다.

n개의 금판을 다른 기둥으로 옮길 때 2^n-1번의 이동을 해야 된다.

하노이 탑

$$
\begin{aligned}
hanoi(5) &= hanoi(4) + 1 + hanoi(4) \\
&= 1 + 2hanoi(4) \\
&= 1 + 2(1 + 2\,hanoi(3)) \\
&= 1+2+4+8+16 = 2^5 - 1
\end{aligned}
$$

```
static void hanoi(int number,  char from,  char via,  char to)
{
  if(number == 1)
    System.out.println(number+"번 원판을 "+ from +"에서 "+ to +"로 옮김");
  else
  {
    hanoi(number -1, from, to, via);
    System.out.println(number+"번 원판을 "+ from +"에서 "+ to +"로 옮김");
    hanoi(number -1, via, from, to);
  }
```

수학 메서드

자바에서 객체를 생성하지 않고 메서드를 사용하기 위해서 메서드에 static 키워드를 사용한 정적 메서드를 생성한다.

자바에 정의된 수학을 위한 메서드를 위해 Math 클래스를 사용한다.

삼각함수

```
(Math.sin(Math.PI/6));
(Math.cos(Math.PI/6));
(Math.tan(Math.PI/6));
```

최댓값 최솟값 출력

```
(Math.max(100,200));
(Math.min(100,200));
```

랜덤함수

```
(Math.random() * 3);
```