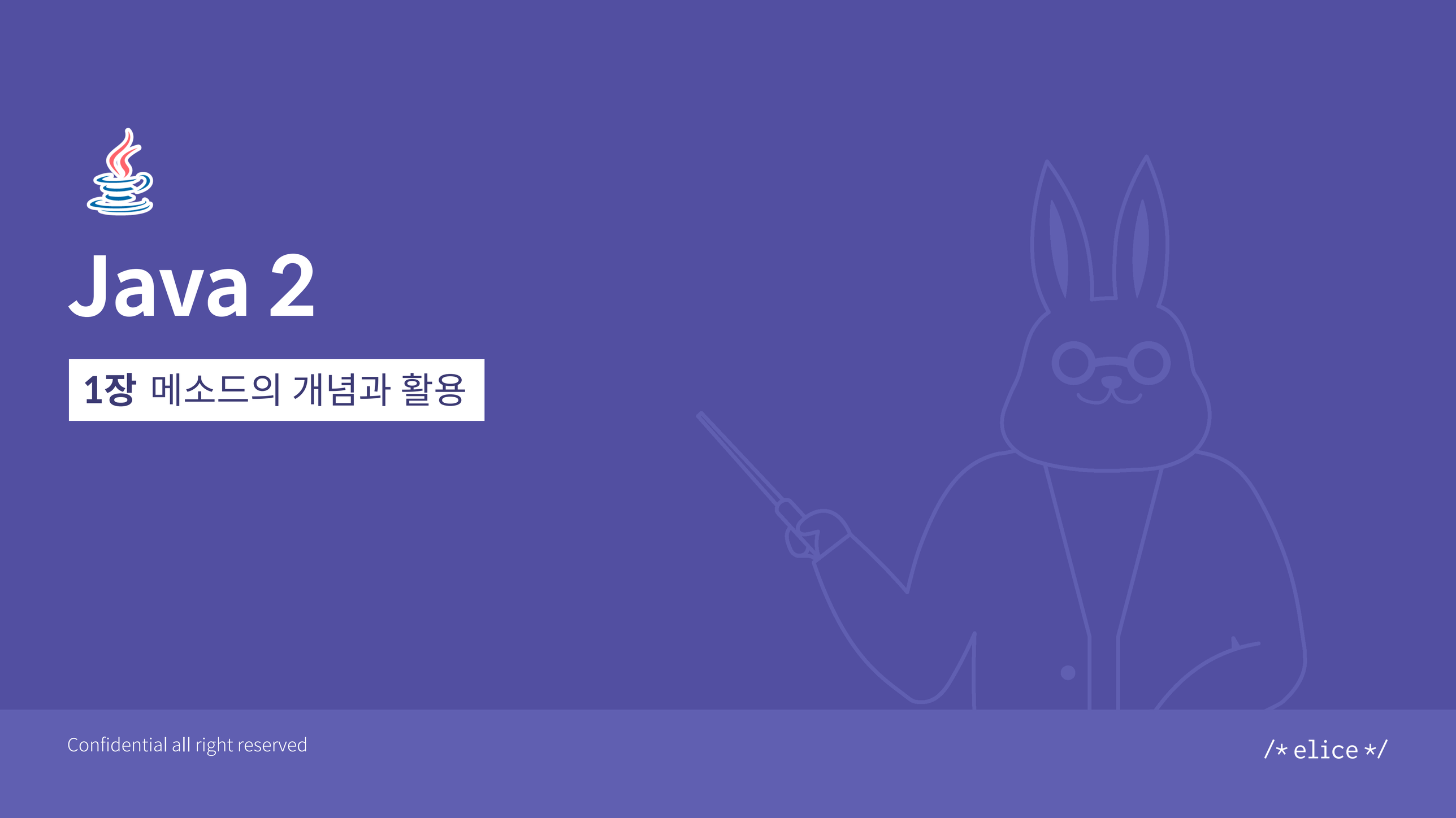


Java 2

1장 메소드의 개념과 활용

/* elice */

Contents

Goal

Goal 1

메소드에 대해 학습하고 활용할 수 있다.

Goal 2

클래스의 구조를 파악하고 객체 지향 프로그래밍의 개념을 알 수 있다.

Goal 3

생성자와 static 키워드를 학습하여 활용할 수 있다.

Goal 4

접근 제한자를 학습하여 캡슐화와 정보은닉의 개념을 습득할 수 있다.

Curriculum

01 메소드

메소드의 개념을 알아보고 직접 사용해봅니다.

02 객체지향 프로그래밍과 클래스

클래스와 인스턴스의 관계에 대해 알아보고 직접 클래스를 만들어봅니다.

03 객체를 만드는 생성자

생성자의 개념을 배우고 객체를 생성할 때 다양한 방법으로 초기화 해봅니다.

04 캡슐화와 정보 은닉

접근 제어자와 캡슐화의 필요성에 대해 알아봅니다.

실력 확인 테스트!

학습한 내용들을 활용해서 실력을 점검해봅니다.

Target

프로그래밍 입문자

기초 코딩을 자바로 배우고 싶은 분

자바 개발자 꿈나무

자바 개발의 첫 단계를 체계적으로 시작하고 싶은 분

객체지향 프로그래밍 입문자

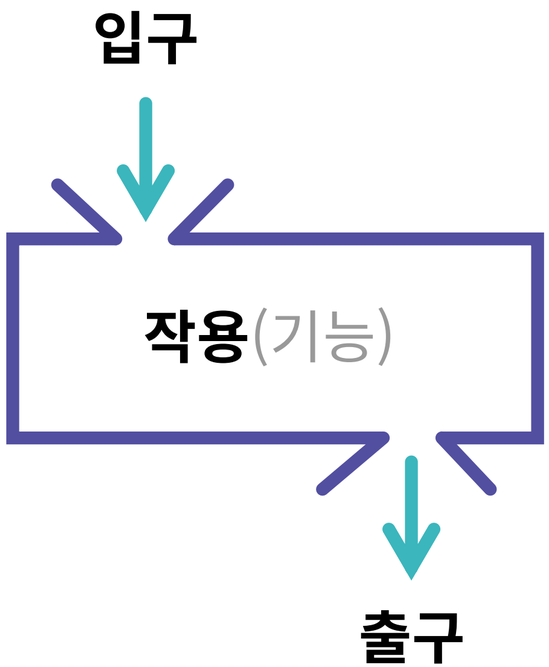
자바로 객체지향 프로그래밍을 배우고 싶은 분

01

메소드

메소드란?

특정 기능을 하는 함수(입력과 출력)



01 메소드

메소드의 개념

Example

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

    }
}
```

메소드의 입출력

입력 값 : **매개변수**(여러 개 가능)

출력 값 : **반환값**

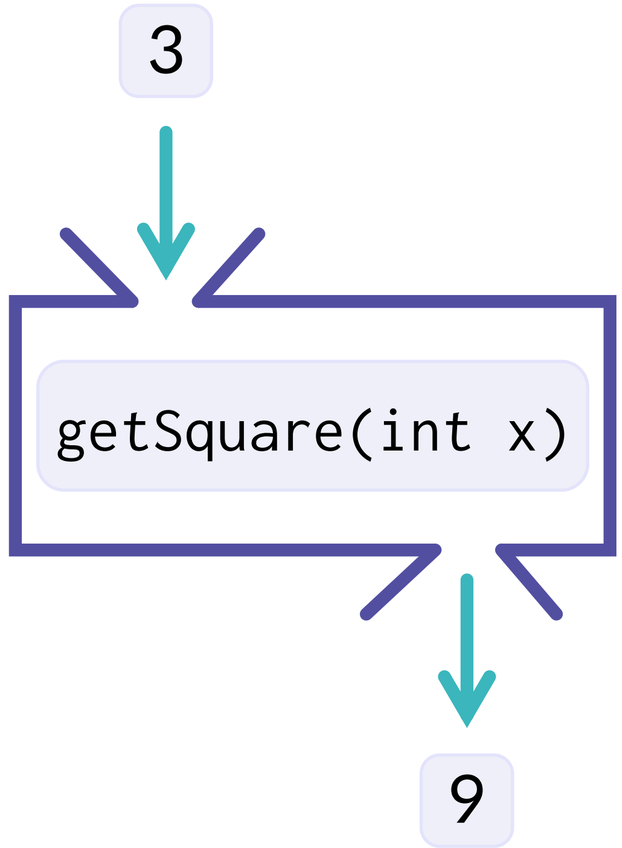
Example

```
public static int getSquare(int x){
                반환형    메소드명    매개변수

    return x * x;
           반환값

}
```

메소드 호출 예시

Example

```
public class Square {
    public static int getSquare(int x){
        return x * x;
    }
    public static void main(String[] args) {
        int value = 3;
        int result = getSquare(value);
        System.out.println(result); //9
    }
}
```



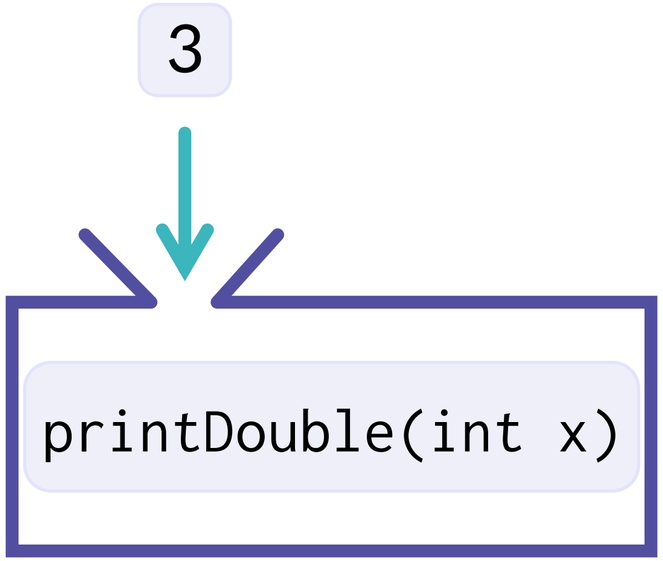
반환값이 없는 메소드

내부 기능만 동작하고 값을 반환하지 않는 메소드

반환값이 없는 메소드

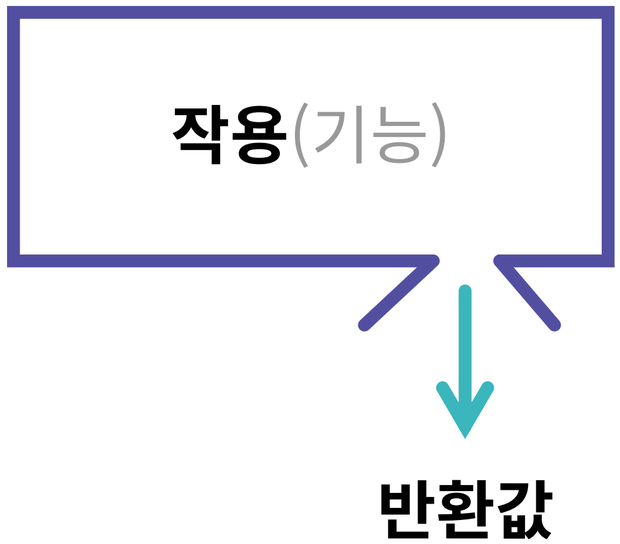
Example

```
public class Square {
    public static void printDouble(int x){
        System.out.println(x * 2);
        return;
    }
    public static void main(String[] args) {
        int value = 2;
        printDouble(value);       //4
        printDouble(3);           //6
    }
}
```



매개변수가 없는 메소드

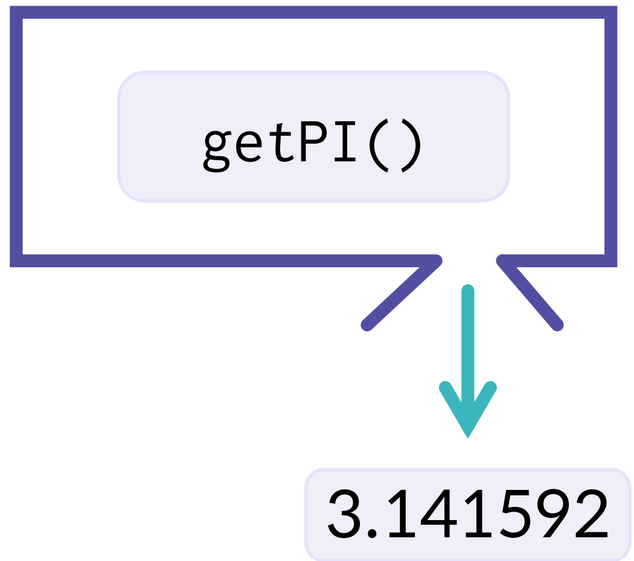
매개변수 없이 **결과값**만 반환하는 메소드



매개변수가 없는 메소드

Example

```
public class Square {
    public static double getPI(){
        return 3.141592;
    }
    public static void main(String[] args) {
        System.out.println(getPI()) //3.141592
    }
}
```



둘 다 없는 메소드

내부 동작만 수행하는 메소드

작용(기능)

01 메소드

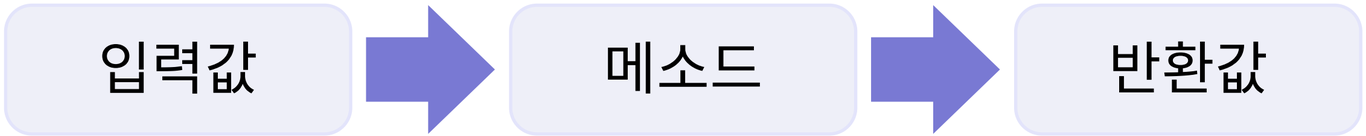
둘 다 없는 메소드

Example

```
public class Square {
    public static int sayHello(){
        System.out.println(“Hello, Elice!”);
    }
    public static void main(String[] args) {
        sayHello();
        sayHello();
        sayHello();
    }
}
```

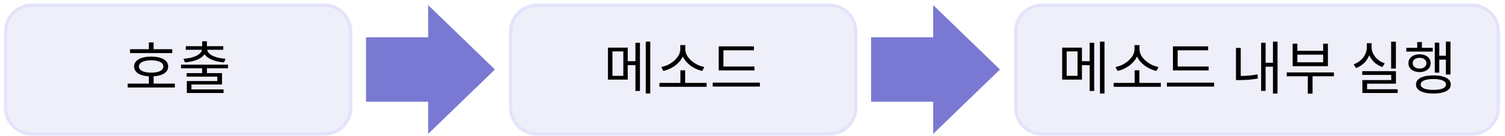
[실습1] 메소드 사용해보기

메소드를 사용하여 결과를 출력해봅시다!



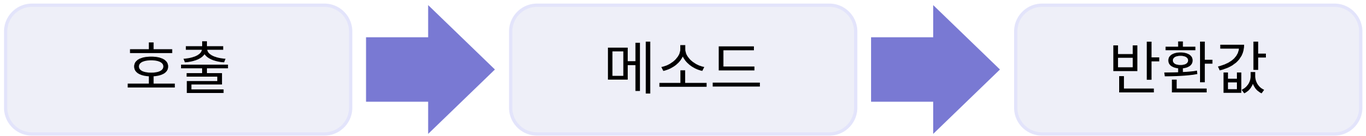
[실습2] 반환값이 없는 메소드

메소드를 사용하여 결과를 출력해봅시다!



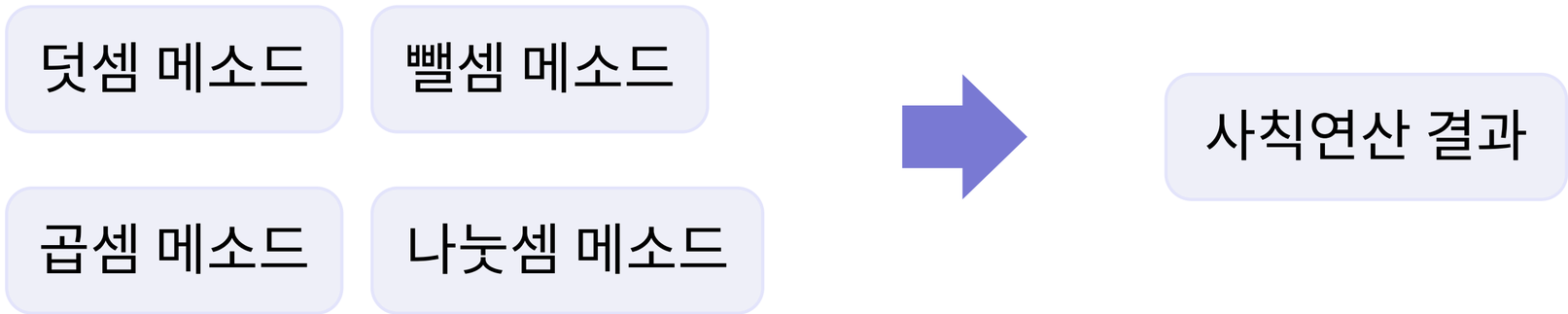
[실습3] 매개변수가 없는 메소드

메소드를 사용하여 결과를 출력해봅시다!



[실습4] 사칙연산

메소드를 이용해 사칙연산을 해봅시다!



02

기존 메소드 호출하기

02 기존 메소드 호출하기

이미 만들어진 메소드

메소드는 직접 구현할 수 있지만

이미 구현되어 있는 메소드를 호출하는 것이 더 **효율적**이다.

02 기존 메소드 호출하기

다양한 메소드의 종류

String 클래스의 메소드

Example

```
String str = “elice”;


System.out.println(str.length()); // 5
System.out.println(str.toUpperCase()); // “ELICE”
```

/* elice */

02 기존 메소드 호출하기

다양한 메소드의 종류

Math 클래스의 메소드

Example

```
System.out.println(Math.max(2, 3)); // 3
System.out.println(Math.round(9.99)); // 10
System.out.println(Math.sqrt(25)); // 5
```

02 기존 메소드 호출하기

다양한 메소드의 종류

Arrays 클래스의 메소드

Example

```
int[] arr = {31, 7, -3, 18};

Arrays.sort(arr); // {-3, 7, 18, 31}
```

02 기존 메소드 호출하기

String 클래스의 메소드

String 클래스에 포함된 주요 메소드

equals()	indexOf()	length()
substring()	startsWith()	endsWith()
replace()	toLowerCase()	toUpperCase()

02 기존 메소드 호출하기

String 클래스의 메소드

`equals()`	같은 문자열인지 비교
`indexOf()`	지정한 문자가 몇번째에 있는지 반환
`length()`	문자열의 길이를 반환
`substring()`	현재 문자열의 부분 문자열을 반환

02 기존 메소드 호출하기

String 클래스의 메소드

`startsWith()`	문자열이 특정 문자열로 시작되는지 확인
`endsWith()`	문자열이 특정 문자열로 끝나는지 확인
`replace()`	문자열에 있는 특정 문자열을 다른 문자열로 교체
`toLowerCase`	문자열을 모두 소문자로 변경
`toUpperCase`	문자열을 모두 대문자로 변경

02 기존 메소드 호출하기

더 많은 String 클래스의 메소드

char	charAt(int index)
IntStream	chars()
int	codePointAt(int index)
int	codePointBefore(int index)
int	codePointCount(int beginIndex, int endIndex)
IntStream	codePoints()
int	compareTo(String anotherString)
int	compareToIgnoreCase(String str)
String	concat(String str)
boolean	contains(CharSequence s)

int	indexOf(int ch)
int	indexOf(int ch, int fromIndex)
int	indexOf(String str)
int	indexOf(String str, int fromIndex)
String	intern()
boolean	isEmpty()
static String	join(CharSequence delimiter, CharSequence
static String	join(CharSequence delimiter, Iterable<? e CharSequence> elements)
int	lastIndexOf(int ch)
int	lastIndexOf(int ch, int fromIndex)

[실습5] String 클래스의 메소드(1)

아래 **메소드를 활용**하여 값을 출력해봅시다!

`charAt()` `concat()` `equals()`

[실습6] String 클래스의 메소드(2)

아래 **메소드를 활용**하여 값을 출력해봅시다!

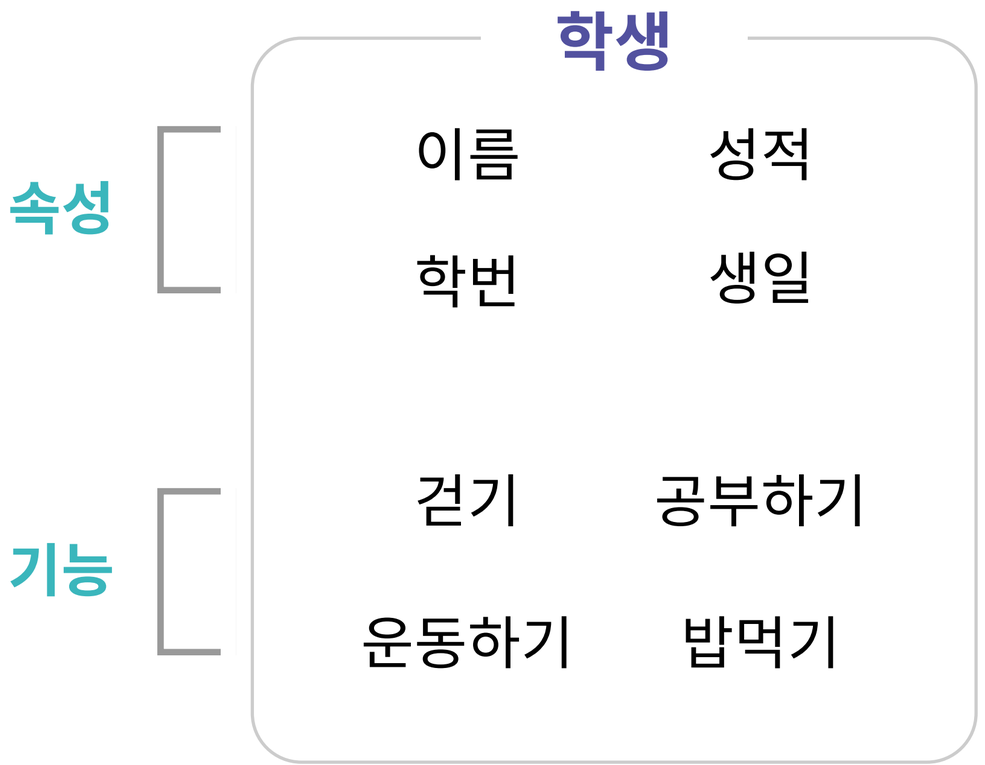
`length()` `substring()` `toUpperCase()` `toLowerCase()`

03

클래스 소개

클래스

속성과 **기능**을 묶어 놓은 집합체

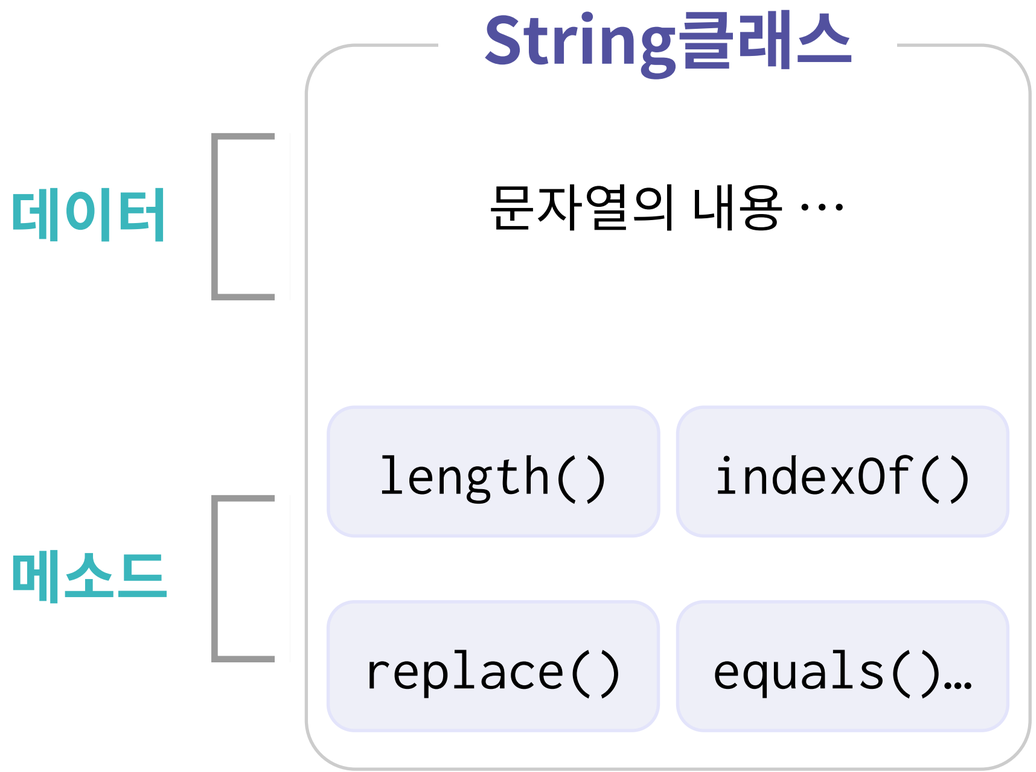


클래스

속성(데이터)과 **기능**(메소드)을 묶어 놓은 집합체

클래스

속성(데이터)과 **기능**(메소드)을 묶어 놓은 집합체



클래스

자바 프로그래밍의 중심에는 **클래스**가 있다.

프로그램 작성한다 =

필요한 클래스의 객체를 생성하고

그 객체의 **속성**과 **기능**을 호출한다.

Credit

/* elice */

코스 매니저
강윤수

콘텐츠 제작자
강윤수

강사
유동환 선생님

디자인
박주연

Contact

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io