메소드(method)

안 화 수

메소드

- 메소드(method)

1. 메소드는 여러 코드를 모아놓은 집합체이다.
2. 메소드는 자바에서 클래스를 구성하는 멤버 중 하나이다.
   클래스 = 필드 + 생성자 + 메소드
3. 클래스에서 메소드를 사용하면 중복되는 코드의 사용을 줄일 수 있다.
4. 클래스에서 메소드를 사용하면 코드를 재사용 할 수 있다.
5. 프로그램에서 문제가 발생하거나 기능의 변경이 필요할 때 손쉽게 유지보수를 할 수 있다.

메소드

- 메소드(method) 선언

선언부 { 접근제어자 리턴타입 메소드명(매개변수 선언) {

구현부 {
```
        // 실행할 코드 작성
        return 반환값;
    }
```

```
public    static    int    check(int a){

        return    값;
}
```

메소드

❖ 접근 제한자

해당 메소드에 접근할 수 있는 범위를 명시한다.

- public : 외부 클래스에서 자유롭게 사용할 수 있다.
- default : 같은 패키지에 소속된 클래스에서만 사용할 수 있다. - 생략된 형태
- protected : 같은 패키지 또는 자식 클래스에서 사용할 수 있다. - 상속에서 주로 사용
- private : 외부에서 사용할 수 없다. (클래스 내부에서만 사용가능)

메소드

메소드 예제1 (1/3)

```
public class MethodEx01 {

    // 사용자 정의 메소드
    // 사용자 정의 메소드는 프로그래머가 직접 호출해야 실행된다.
    static void check() {    // static : 정적 메소드
        System.out.println("메소드 호출 성공");
        return;              // return 생략가능
    }
    // 값 전달에 의한 메소드 호출방식(Call by Value방식)
    static void check(int a) {    //매개변수(parameter) : int a=30
        System.out.println("전달된 값:"+a);
    }
    static void check(int a, double d) {
        double result = a + d;
        System.out.println("전달된 값의 합:"+ result);
    }
    static void check(char c) {                 // char c = 'A'
        System.out.println("전달된 값:"+ c);
    }
    static void check(boolean b) {              // boolean b = true;
        System.out.println("전달된 값:"+b);
    }
```

메소드

메소드 예제1 (2/3)

```
// 주소값 전달에 의한 메소드 호출방식(Call by Reference방식)
static void check(String s) {          //String s="자바;
    System.out.println("전달된 값:"+s); //String s=new String("파이썬");
}

// return문 : 메소드를 호출한 곳에 값을 돌려주는 역할
// return문은 메소드 가장 마지막 줄에 사용해야 한다.
static int check01() {
    System.out.println("리턴구문");
    return 50;
}

static double check02(int a, double d) {
    double result = a + d;
    return result;
}
```

메소드

메소드 예제1 (3/3)

```
// main() 메소드는 자바 가상머신(java.exe)으로만 호출된다.
// main() 메소드는 프로그램머가 직접 호출할 수 없다.
public static void main(String[] args) {
    MethodEx01.check();          // check메소드 호출
    check();
    check(30);
    check(10, 20.5);
    check('A');
    check(true);
    check("자바");
    check(new String("파이썬"));

    check01();       // return문으로 돌려받은 값이 출력되지 않는다.
    int result = check01();
    System.out.println("돌려 받은 값1:" + result);
    System.out.println("돌려 받은 값2:" + check01());

    double result2 = check02(50, 3.14);
    System.out.println("돌려 받은 값3:"+ result2);
    System.out.println("돌려 받은 값4:"+ check02(50, 3.14));
}
}
```

메소드

- 메소드 예제2

```
public class MethodEx02 {

    // 1 ~ n까지 합을 구하는 메소드
    static void sum(int n) {     // int n = 3
        int hap=0;
        for(int i=1; i<=n; i++) {
            hap += i;            // hap = hap + i
        }
        System.out.println("1~"+n+"="+hap);
    }

    public static void main(String[] args) {
        sum(3);
        sum(5);
        sum(10);
        sum(30);
        sum(100);
        sum(1000);
        sum(10000);
    }
}
```

메소드

메소드 예제3 (1/2)

```
public class MethodEx03 {

    // 키보드로 입력한 2개의 정수 중에서 최대값과 최소값을 구하는 프로그램을 작성하세요?
    // 단, 메소드를 이용해서 작성하세요?

    // 최대값
    static int max(int a, int b) {        // 정적 메소드
        if(a > b)
            return a;
        else
            return b;
    }

    // 최소값
    static int min(int a, int b) {        // 정적 메소드
        if(a < b)
            return a;
        else
            return b;
    }
```

메소드

- 메소드 예제3 (2/2)

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

        int n1, n2, max, min;
        System.out.println("2개의 정수를 입력 하세요?");
        Scanner sc = new Scanner(System.in);
        n1 = sc.nextInt();                  // n1 = 10
        n2 = sc.nextInt();                  // n2 = 20

        max = MethodEx03.max(n1, n2);    // max() 메소드 호출
        min = MethodEx03.min(n1, n2);    // min() 메소드 호출
        System.out.println("max:"+ max);
        System.out.println("min:"+ min);

    }
}
```