

4장. 배열(Array), 함수(Method)



Array & Method



배열(Array)

● 배열은 왜 써야 할까?

- 정수 20개를 이용한 프로그램을 할 때 20개의 정수 타입의 변수를 선언

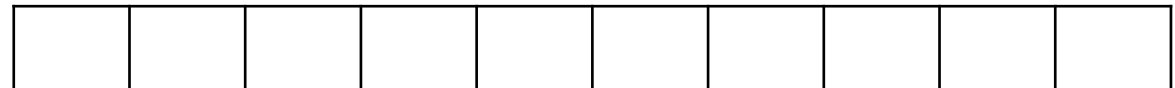
int num1, int num2, int num3... num20;



비효율적이고 관리하기 어렵다.

- 배열을 선언하면 선언한 자료형과 배열 길이에 따라 메모리가 할당된다.

int num[20];



num[0] num[1] num[2]

num[19]

num - 배열 이름
[] - 인덱스 연산자

0 ~ n-1 개



배열 사용하기

● 배열 선언 및 자료 저장

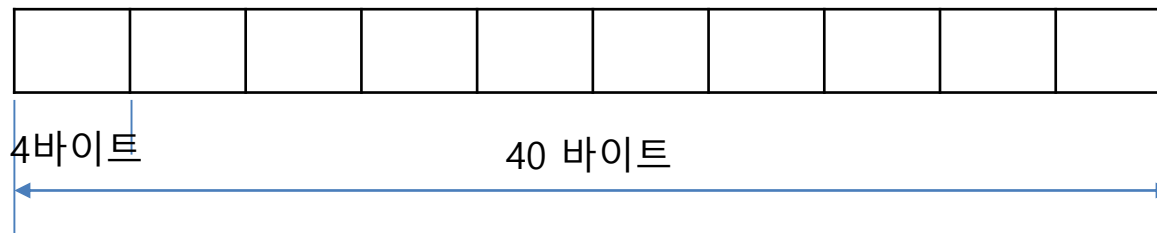
① 자료형[] 배열 이름 = **new** 자료형[개수]

② 자료형 배열 이름[] = **new** 자료형[개수]

```
int[ ] numbers = new int[10]
```

- 배열 길이(length) - 10개

numbers



- 배열의 길이 - `numbers.length`



배열(Array)

■ 문자열 배열 관리

```
package arrays;

public class ArrayStringTest {

    public static void main(String[] args) {
        // 변수
        String car1 = "소나타";
        String car2 = "EV3";
        String car3 = "BMW";

        System.out.println(car1);
        System.out.println(car2);
        System.out.println(car3);

        //1. 문자열 배열의 선언과 초기화
        String[] cars = {"소나타", "EV3", "BMW"};

        //특정 요소 검색
        System.out.println(cars[0]);

        //배열의 길이
        System.out.println("배열의 길이: " + cars.length);

        //전체 조회(검색)
        for(int i=0; i<cars.length; i++) {
            System.out.println(cars[i]);
        }
    }
}
```



배열(Array)

▪ 문자열 배열 관리

```
//2. 문자열 배열 선언
String[] carList = new String[3];

//전체 출력
for(int i=0; i<carList.length; i++) {
    System.out.println(carList[i]);
}

//요소 저장
carList[0] = "소나타";
carList[1] = "EV3";
carList[2] = "BMW";

//특정 요소 검색
System.out.println(carList[2]);

//특정 요소 수정
carList[1] = "Ionic6";

//전체 출력
for(int i=0; i<carList.length; i++) {
    System.out.println(carList[i]);
}
}
```



배열(Array)

■ 정수형 배열 관리

```
public class ArrayTest2 {  
    public static void main(String[] args) {  
        //정수형 배열의 크기가 4(초기값: 0 0 0 0)  
        int[] number = new int[4];  
  
        //배열에 저장  
        number[0] = 10;  
        number[1] = 20;  
        number[2] = 30;  
        number[3] = 40;  
  
        System.out.println("배열의 개수 : " + number.length);  
  
        //특정 위치에 접근(인덱싱)  
        System.out.println(number[1]);  
  
        //전체 값 조회(출력)  
        for(int i=0; i<number.length; i++) {  
            System.out.println(number[i]);  
        }  
    }  
}
```



배열(Array)

■ 디버깅(Debugging)

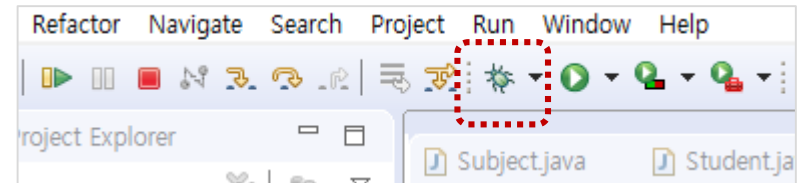
① 브레이크 포인트 설정

```
6 // 배열의 연산
7 int[] arr = new int[4];
8
9 //저장
10 arr[0] = 3;
11 arr[1] = 6;
12 arr[2] = 9;
13 arr[3] = 12;
14
15 System.out.printf("배열의 개수 : %d\n", arr.length);
16
```

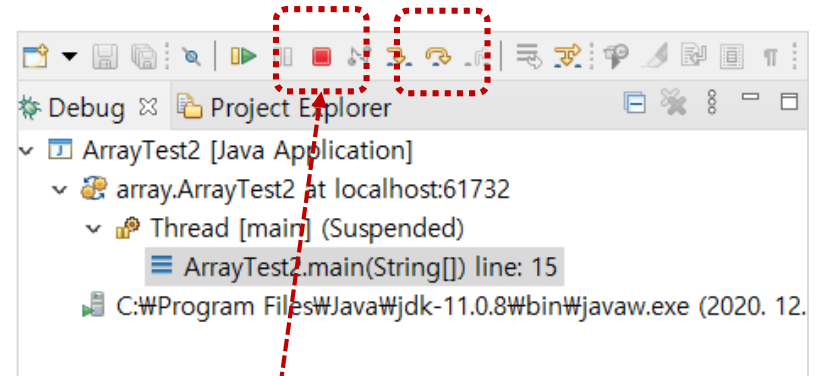
④ 결과 확인

| (x) Variables Breakpoints Expressions | |
|---------------------------------------|-------------------|
| Name | Value |
| no method return value | |
| args | String[0] (id=19) |
| arr | (id=20) |
| [0] | 3 |
| [1] | 6 |
| [2] | 9 |
| [3] | 12 |

② 디버그 실행



③ Step Over(단계)



⑤ 디버깅 종료



배열(Array)

■ 실수형 배열의 연산

```
double[] data = new double[5];
double total = 0.0;    //총합
double times = 0.0;    //곱한값

//저장
data[0] = 10.0;
data[1] = 20.0;
data[2] = 30.0;

//연산 및 조회
for(int i=0; i<data.length; i++) {
    total += data[i];
    times *= data[i];
    System.out.println(data[i]);
}
System.out.println();
System.out.println("총합 : " + total);
System.out.println("곱 : " + times);
```

10.0

20.0

30.0

0.0

0.0

총합 : 60.0

곱 : 0.0



배열(Array)

■ 배열의 유효한 요소값 출력하기

```
double[] data = new double[5];
int size = 0;

data[0] = 10.0;
size++;

data[1] = 20.0;
size++;

data[2] = 30.0;
size++;

//출력
for(int i=0; i<size; i++) {
    System.out.println(data[i]);
}
```

```
10.0
20.0
30.0
```



배열(Array)

■ 문자형 배열 – 알파벳 저장하고 출력하기

```
public class ArrayCharTest {  
    public static void main(String[] args) {  
        // 알파벳 대문자 저장하고 출력  
        char ch = 'A';  
        char[] alphabets = new char[26];  
  
        //ch = (char) (ch + 1);  
        //ch++;  
        //System.out.println(ch); //'B'  
  
        //배열에 알파벳 저장  
        for(int i=0; i<alphabets.length; i++) {  
            alphabets[i] = ch;  
            ch++;  
        }  
  
        //출력  
        for(int i=0; i<alphabets.length; i++) {  
            System.out.println(alphabets[i] + ", " + (int)alphabets[i]);  
        }  
    }  
}
```

| | |
|-----|---|
| 65, | A |
| 66, | B |
| 67, | C |
| 68, | D |
| 69, | E |
| 70, | F |
| 71, | G |
| 72, | H |
| 73, | I |
| 74, | J |
| 75, | K |
| 76, | L |
| 77, | M |
| 78, | N |
| 79, | O |
| 80, | P |
| 81, | Q |
| 82, | R |
| 83, | S |
| 84, | T |
| 85, | U |
| 86, | V |



향상된 for문과 배열

■ 향상된 for문

```
for(자료형 변수 : 배열이름){  
    반복실행(변수)  
}
```

```
//향상 for  
for(String car : carList) {  
    System.out.println(car);  
}
```

```
//향상 for  
for(int num : number) {  
    System.out.print(num + " ");  
}
```



배열 복사하기

■ 배열 복사하기

1. 기존 배열과 자료형 및 배열 크기가 똑같은 배열을 새로 만들때.
2. 배열의 모든 요소에 자료가 꼭 차서 더 큰 배열을 만들때

```
public class ArrayCopy {  
  
    public static void main(String[] args) {  
        // 배열 복사  
        int[] arr1 = {10, 20, 30, 40};  
        int[] arr2 = new int[4];  
  
        //복사하여 저장  
        for(int i=0; i<arr2.length; i++) {  
            arr2[i] = arr1[i];  
        }  
  
        for(int i=0; i<arr2.length; i++) {  
            System.out.print(arr2[i] + " ");  
        }  
        System.out.println();  
    }  
}
```



배열 복사하기

■ 배열 복사하기

거꾸로 복사하기

```
//역순으로 복사
char[] array1 = {'N', 'E', 'T'};
char[] array2 = new char[3];

//복사
for(int i=0; i<array1.length; i++) {
    array2[i] = array1[2-i];
}

//출력
for(int i=0; i<array2.length; i++) {
    System.out.print(array2[i] + " ");
}
}
```

NET
TEN



최대값 찾기

■ 최대값과 최대값 위치 찾기

```
public class FindMaxValue{  
  
    public static void main(String[] args) {  
        //최대값 찾기  
        int[] array = new int[]{1, 5, 3, 8, 2};  
        int maxVal = array[0]; //0번 인덱스값을 최대값으로 설정함  
  
        for(int i=1; i<array.length; i++) {  
            if(array[i] > maxVal)  
                maxVal = array[i];  
        }  
        System.out.println("최대값 : " + maxVal);  
  
        //최대값의 위치 찾기  
        int maxIdx = 0; //인덱스 0을 최대값 위치로 설정함  
  
        for(int i=1; i<array.length; i++) {  
            if(array[i] > array[maxIdx])  
                maxIdx = i;  
        }  
        System.out.println("최대값의 위치 : " + maxIdx);  
    }  
}
```



배열을 이용한 성적 처리 프로그램

■ 성적 처리 프로그램

- 메뉴 화면 만들기 : 1.학생수 | 2.점수입력 | 3.점수리스트 | 4.분석 | 5.종료
- 학생수 : 배열로 생성
- 점수 입력 : 학생 배열에 점수 입력
- 점수 리스트: 점수 목록 조회
- 분석 : 평균 점수와 최고 점수를 계산
- 종료 : 프로그램 종료
- 메뉴에 없는 번호 선택 : 메뉴를 잘못 누름, 다시 입력



배열을 이용한 성적 처리 프로그램

■ 성적 처리 프로그램

```
=====
1. 학생수 | 2. 점수입력 | 3. 점수 리스트 | 4. 분석 | 5. 종료
=====
선택> 1
학생수> 3
=====
1. 학생수 | 2. 점수입력 | 3. 점수 리스트 | 4. 분석 | 5. 종료
=====
선택> 2
scores[0]>75
scores[1]>85
scores[2]>95
=====
1. 학생수 | 2. 점수입력 | 3. 점수 리스트 | 4. 분석 | 5. 종료
=====
선택> 3
scores[0]=75
scores[1]=85
scores[2]=95
```

```
=====
1. 학생수 | 2. 점수입력 | 3. 점수 리스트 | 4. 분석 | 5. 종료
=====
선택> 4
평균 점수: 85.0
최고 점수: 95
=====
1. 학생수 | 2. 점수입력 | 3. 점수 리스트 | 4. 분석 | 5. 종료
=====
선택> 5
프로그램을 종료합니다.
```



배열을 이용한 성적 처리 프로그램

■ 성적 처리 프로그램

```
public class CalcScore {
    public static void main(String[] args) {
        boolean run = true;    //스위칭 변수
        int studentNum = 0;    //학생수
        int[] scores = null;    //점수 배열
        Scanner scan = new Scanner(System.in);

        while(run) {
            System.out.println("=====");
            System.out.println("1.학생수 | 2.점수입력 | 3.점수 리스트 | 4.분석 | 5.종료");
            System.out.println("=====");
            System.out.print("선택> ");

            int menu = scan.nextInt(); //menu 입력
            if(menu == 1) {
                System.out.print("학생수> ");
                studentNum = scan.nextInt();
                scores = new int[studentNum];
            }else if(menu == 2) {
                for(int i=0; i<scores.length; i++) {
                    System.out.print("scores[" + i + "]>");
                    scores[i] = scan.nextInt();
                }
            }else if(menu == 3) {
                for(int i=0; i<scores.length; i++) {
                    System.out.println("scores[" + i + "]=" + scores[i]);
                }
            }
        }
    }
}
```



배열을 이용한 성적 처리 프로그램

■ 성적 처리 프로그램

```
    }else if(menu == 4) {  
        //평균, 최고 점수  
        int sum = 0;  
        double avg = 0.0;  
        int max = scores[0]; //최고점수 설정  
  
        for(int i=0; i<scores.length; i++) {  
            sum += scores[i]; //합계  
            if(scores[i] > max) //점수와 최고점수 비교  
                max = scores[i];  
        }  
        //평균 계산  
        avg = (double) sum / scores.length;  
        System.out.println("평균 점수: " + avg);  
        System.out.println("최고 점수: " + max);  
    }else if(menu == 5) {  
        System.out.println("프로그램을 종료합니다.");  
        run = false;  
    }else {  
        System.out.println("메뉴를 잘못 눌렀습니다. 다시 선택하세요.");  
    }  
} //while 종료  
scan.close();  
}
```



정렬(Sort)

■ 배열 요소의 정렬

정렬(sort)은 자료를 크기 순서로 맞춰 일렬로 나열하는 것이다.

오름 차순 또는 내림 차순으로 정렬한다.

버블 정렬, 선택정렬 등의 방법이 있다.

● 버블정렬

서로 인접한 두 원소를 검사하여 정렬하는 알고리즘으로 크기를 비교하여 서로 교환 한다.



정렬(Sort)

■ 배열 요소의 정렬

```
public class SortTest {  
  
    public static void main(String[] args) {  
  
        int[] arr = {3, 6, 9, 2, 5, 4};  
        int i, j, temp;  
  
        //오름차순 정렬로 저장하기 - 버블 정렬 방식  
        for(i = 0; i < arr.length; i++) {  
            for(j = 0; j < arr.length-1; j++) { //열 종료값-1  
                if(arr[j] > arr[j+1]) { //앞 수가 뒷 수보다 크면  
                    temp = arr[j]; //자리 바꿈  
                    arr[j] = arr[j+1];  
                    arr[j+1] = temp;  
                }  
            }  
        }  
    }  
}
```



정렬(Sort)

■ 배열 요소의 정렬

```
/*  
    1행 -- 3, 6, 2, 5, 4, 9  
    2행 -- 3, 2, 5, 4, 6, 9  
    3행 -- 2, 3, 5, 4, 6, 9  
    4행 -- 2, 3, 4, 5, 6, 9  
*/  
  
//변경된 데이터 출력  
for(int a : arr)  
    System.out.print(a + " ");  
}  
}
```



다차원 배열

■ 2차원 이상의 배열

1. 지도, 게임 등 평면이나 공간을 구현할 때 많이 사용됨.
2. 이차원 배열의 선언과 구조

```
int[ ][ ] arr = new int [2][3]
```

| | | |
|-----------|-----------|-----------|
| arr[0][0] | arr[0][1] | arr[0][2] |
| | | |
| | | |
| arr[1][0] | arr[1][1] | arr[1][2] |

3. 선언과 초기화

```
int[ ][ ] arr = {{1, 2, 3},{4, 5, 6}}
```

| | | |
|-----------|-----------|-----------|
| arr[0][0] | arr[0][1] | arr[0][2] |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| arr[1][0] | arr[1][1] | arr[1][2] |



다차원 배열

■ 2차원 배열 생성 및 초기화

```
public class TwoDimention1 {  
  
    public static void main(String[] args) {  
        //정수형 2차원 배열  
        int[][] a = new int[2][3];  
        int i, j;  
  
        System.out.println(a.length);           //행의 크기  
        System.out.println(a[0].length);        //열의 크기  
        System.out.println(a[1].length);  
  
        //배열의 초기값 출력  
        for(i=0; i<a.length; i++) {  
            for(j=0; j<a[i].length; j++) {  
                System.out.print(a[i][j] + " ");  
            }  
            System.out.println();  
        }  
        System.out.println("=====");  
    }  
}
```



다차원 배열

■ 2차원 배열 생성 및 초기화

```
//저장1
a[0][0] = 1;
a[0][1] = 2;
a[0][2] = 3;
a[1][0] = 4;
a[1][1] = 5;
a[1][2] = 6;

//저장2
/*int[][] a = {
    {1, 2, 3},
    {4, 5, 6}
};*/
for(i=0; i<a.length; i++) {
    for(j=0; j<a[i].length; j++) {
        System.out.print(a[i][j] + " ");
    }
    System.out.println();
}
}
```



다차원 배열

■ 실습 예제 - 아파트 세대 구현하기

| | | |
|------|------|------|
| 5, 1 | 5, 2 | 5, 3 |
| 4, 1 | 4, 2 | 4, 3 |
| 3, 1 | | |
| 2, 1 | 2, 2 | 2, 3 |
| 1, 1 | 1, 2 | 1, 3 |



다차원 배열

■ 실습 예제 - 아파트 세대 구현하기

```
public class ApartmentTest {  
  
    public static void main(String[] args) {  
        // 아파트 구조  
        int[][] household = new int[5][3];  
        household[2] = new int[1];  
  
        //배열의 행의 크기  
        System.out.println(household.length);  
  
        //배열의 열의 크기  
        System.out.println(household[0].length);  
  
        //세대수 출력  
        System.out.printf("5층 %d세대\n", household[4].length); //3세대  
        System.out.printf("4층 %d세대\n", household[3].length);  
        System.out.printf("3층 %d세대\n", household[2].length); //1세대  
        System.out.printf("2층 %d세대\n", household[1].length);  
        System.out.printf("1층 %d세대\n", household[0].length);  
    }  
}
```



다차원 배열

■ 알파벳 대문자 저장하기

```
public class Array2dAlphabets {  
  
    public static void main(String[] args) {  
        //알파벳 대문자 26개를 저장할 이차원 배열 생성  
        char[][] alphabets = new char[13][2];  
        char ch = 'A';  
        //System.out.println(ch);  
        //ch = (char) (ch + 1); //1증가  
        //ch++;  
        //System.out.println(ch);  
  
        for(int i=0; i<alphabets.length; i++) {  
            for(int j=0; j<alphabets[i].length; j++) {  
                alphabets[i][j] = ch;  
                ch++;  
            }  
        }  
    }  
}
```

| | |
|---|---|
| A | B |
| C | D |
| E | F |
| G | H |
| I | J |
| K | L |
| M | N |
| O | P |
| Q | R |
| S | T |
| U | V |
| W | X |
| Y | Z |



다차원 배열

```
for(int i=0; i<alphabets.length; i++) {  
    for(int j=0; j<alphabets[i].length; j++) {  
        System.out.print(alphabets[i][j] + " ");  
    }  
    System.out.println();  
}  
}
```



다차원 배열

■ 2차원 배열의 연산

```
int[][] numbers = { {1, 2, 3, 4}, {5, 6, 7} };
int i, j;
int sum = 0;
int count = 0;
double avg = 0.0;

//인덱싱
int x = numbers[1][2];
System.out.println("numbers[1][2] = " + x);

//전체 조회
for(i=0; i<numbers.length; i++) {
    for(j=0; j<numbers[i].length; j++) {
        System.out.print(numbers[i][j] + " ");
    }
}
```



다차원 배열

■ 2차원 배열의 연산

```
//합계 및 평균
for(i=0; i<numbers.length; i++) {
    for(j=0; j<numbers[i].length; j++) {
        sum += numbers[i][j];
        count++;
    }
}
avg = (double)sum / count;
System.out.println("합계: " + sum);
System.out.println("개수: " + count);
System.out.println("평균: " + avg);
```

```
numbers[1][2] = 7
1 2 3 4 5 6 7
합계: 28
개수: 7
평균: 4.0
```



다차원 배열

■ 성적 관리 프로그램

```
public class CalcScore {  
  
    public static void main(String[] args) {  
        //학생 5명의 국어, 수학 점수 배열  
        int[][] score = {  
            {91, 70},  
            {80, 50},  
            {76, 60},  
            {90, 49},  
            {80, 80}  
        };  
  
        int[] total = {0, 0};           //과목별 총점  
        double[] avg = {0.0, 0.0};     //과목별 평균  
        int i, j;  
  
        //데이터 출력(조회)  
        System.out.println("국어 수학");  
        for(i=0; i<score.length; i++) {  
            for(j=0; j<score[i].length; j++) {  
                System.out.print(score[i][j] + " ");  
            }  
            System.out.println();  
        }  
    }  
}
```



다차원 배열

■ 성적 관리 프로그램

```
//과목별 총점 계산
for(i=0; i<score.length; i++) {
    total[0] += score[i][0];
    total[1] += score[i][1];
}
//과목별 평균 계산
avg[0] = (double)total[0] / score.length;
avg[1] = (double)total[1] / score.length;

System.out.println("국어 점수 합계 : " + total[0]);
System.out.println("국어 점수 평균 : " + avg[0]);
System.out.println("수학 점수 합계 : " + total[1]);
System.out.println("수학 점수 평균 : " + avg[1]);
}
```



메서드(멤버 함수)

■ 메서드란?

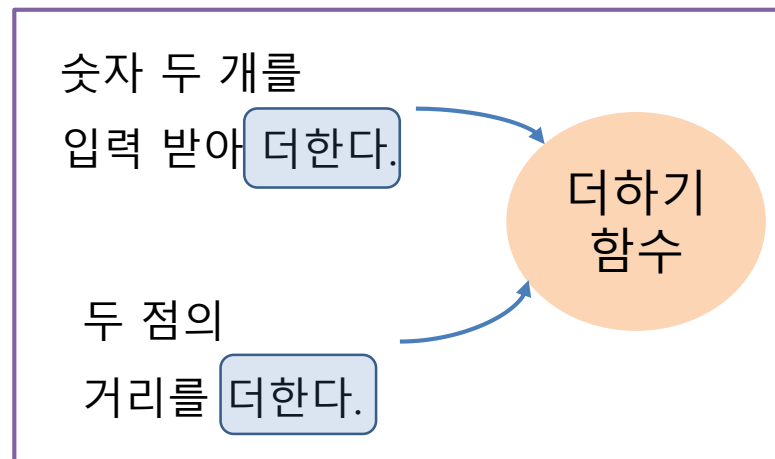
- 객체의 기능을 제공하기 위해 클래스 내부에 구현되는 함수

■ 함수란?

- 하나의 기능을 수행하는 일련의 코드
- 중복되는 기능은 함수로 구현하여 함수를 호출하여 사용함

■ 함수의 장점

- 기능을 나누어 코드를 효율적으로 구현
예) 사칙연산 계산기 – 덧셈, 뺄셈, 곱셈, 나눗셈
※ `main()` 함수 안에서 한꺼번에 구현하면 복잡함
`add()`, `subtract()`, `times()`, `divide()`



메서드(멤버 함수)

■ 메서드(함수) 정의하기

- 함수의 이름, 매개변수, 반환값을 선언하고 코드를 구현함
 - 반환형이 없는 경우 void로 쓴다.
 - 매개변수가 없을 수도 있다.

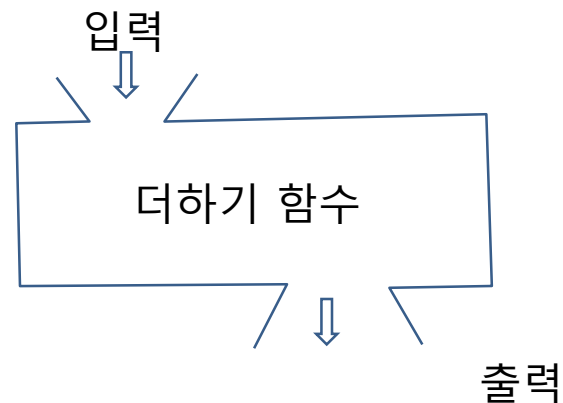
■ 함수의 유형

1. 반환값이 없는 경우

```
void 함수이름(){  
    ...  
}
```

2. 매개변수가 있는 경우

```
void 함수이름(매개변수){  
    ...  
}
```



메서드(멤버 함수)

1. 반환값이 없는 메서드(함수) - 예제

메서드(함수)를 사용하는 것을 '함수를 호출한다'라고 한다.

main()이 있는 파일에서는 static을 사용하여 new 객체를 생성하지 않고도 실행할 수 있다. **(static을 사용해야 하는 이유)**

```
public static void main(String[] args) {  
    sayHello();  
    sayHello2("민수");  
    sayHello2("은서");  
}  
  
public static void sayHello() {  
    System.out.println("hello~");  
}  
  
public static void sayHello2(String name) {  
    System.out.println("hello~ " + name);  
}
```

← 메서드 호출

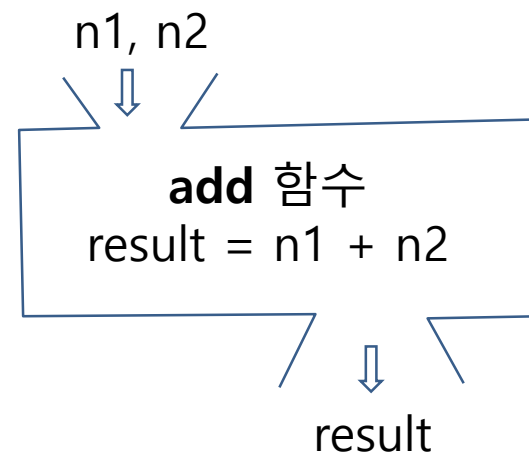
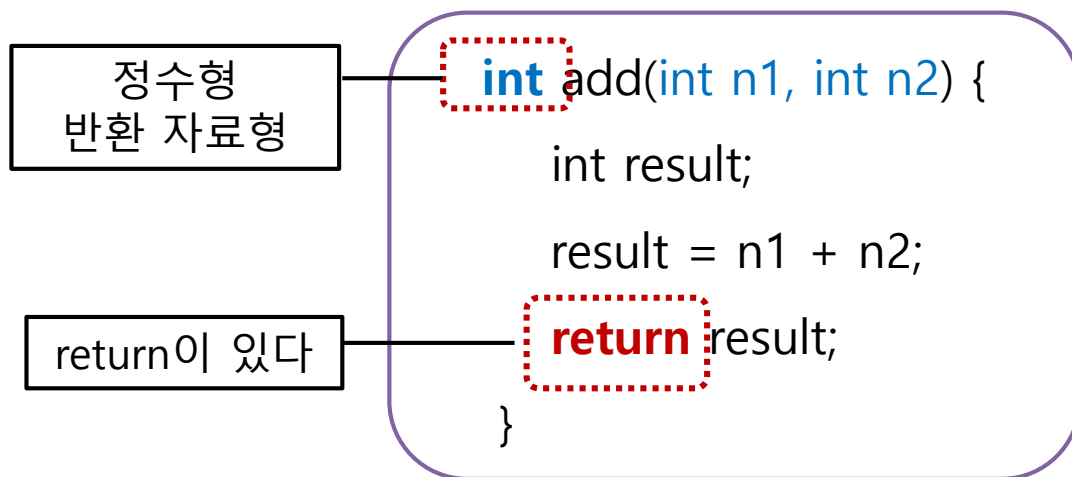
반환 자료형이 없다

← 메서드 이름



메서드(함수)의 유형

- 반환값이 있는 메서드(함수) – 예제 1
반환값이 있는 경우 'return' 예약어를 사용해야 한다.



메소드(함수)의 사용

- 반환값이 있는
메서드(함수)

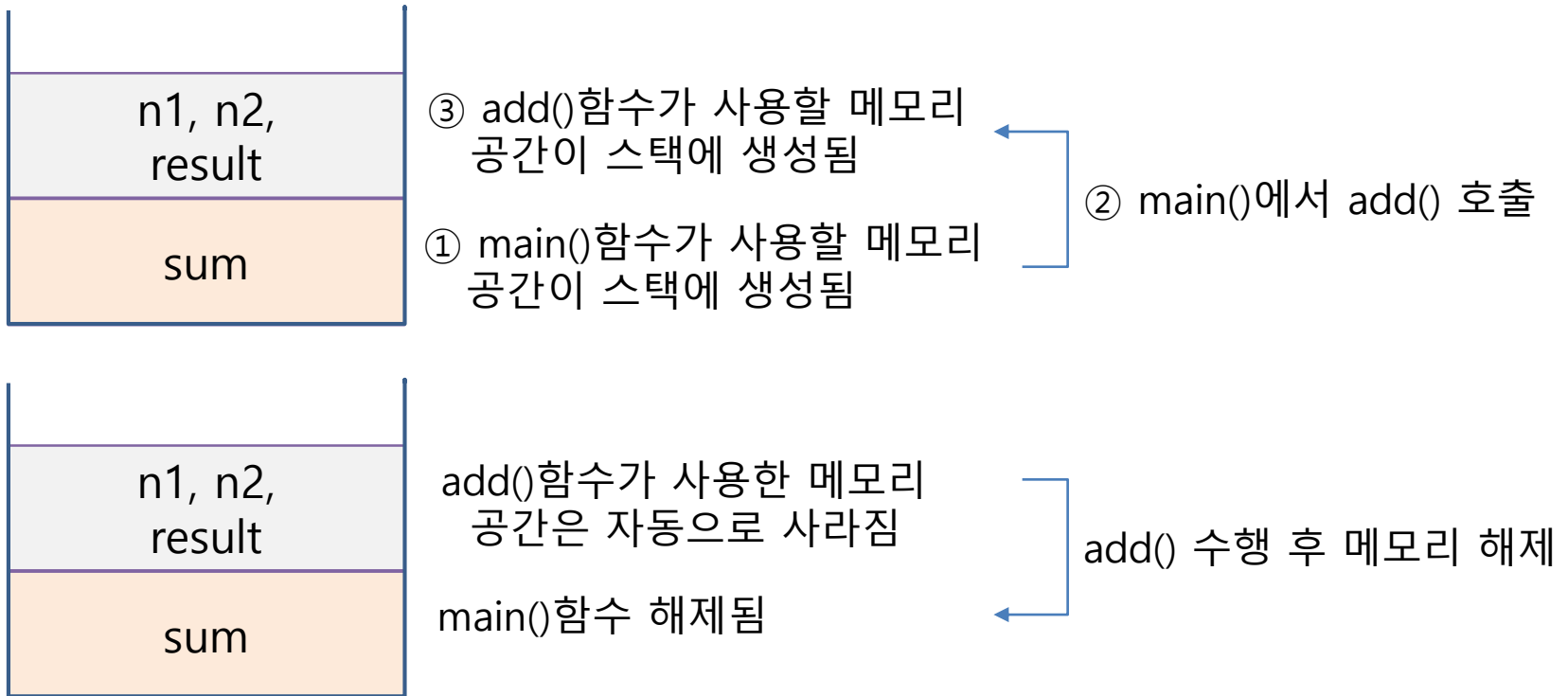
```
public class ReturnMethod {  
  
    //두 수를 더하는 메서드 정의  
    public static int add(int n1, int n2) {  
        int result = n1 + n2;  
        return result;  
    }  
  
    //문자를 반환하는 메서드  
    public static String message() {  
        return "안녕하세요~";  
    }  
  
    public static void main(String[] args) {  
        //static 사용 - new 객체를 생성하지 않아도 실행 할 수 있음  
        //즉, 클래스를 반드시 만들지 않더라도 실행 가능.  
  
        int sum = add(10, 11); //add() 메서드 호출  
        System.out.println("두 수의 합 : " + sum);  
  
        String msg = message();  
        System.out.println(msg);  
    }  
}
```



함수 호출과 스택 메모리

■ 함수 호출과 스택 메모리

- 함수가 호출될 때 사용하는 메모리 – 스택(stack), 지역변수가 위치함
- 함수의 수행이 끝나면 함수에 할당했던 메모리 공간이 해제됨.



변수의 유효 범위

- 변수의 유효 범위

- 지역 변수의 유효 범위

지역 변수는 함수나 메서드 내부에 선언하기 때문에 함수 밖에서는 사용할 수 없다. 지역변수가 생성되는 메모리를 **스택(stack)**이라 한다.

- 멤버 변수의 유효 범위

멤버 변수는 인스턴스 변수라고도 한다. 클래스의 어느 메서드에서나 사용할 수 있다. 클래스가 생성될때 **힙(heap) 메모리**에 생성된다.

- **static** 변수의 유효 범위

사용자가 프로그램을 실행하면 메모리에 프로그램이 상주한다. new로 생성되지 않고 처음부터 **데이터 영역 메모리**에 생성된다. 이 영역에는 상수, 문자열, static 변수가 생성된다.



변수의 범위(스코프)

■ 지역변수의 범위

함수나 제어문에서 사용되며 호출 후에 메모리 공간에서 소멸(해제)한다.

```
public class OneUpTest {  
  
    public static void main(String[] args) {  
        //oneUp() 메서드 호출  
        System.out.println(oneUp()); //2  
        System.out.println(oneUp()); //2  
  
        //x값 호출  
        //System.out.println(x); //x는 이미 소멸한 변수임  
    }  
  
    public static int oneUp() { //x - 지역 변수  
        int x = 1;  
        x++;  
        return x;  
    }  
}
```



변수의 범위(스코프)

■ 정적변수의 범위

- 정적 변수는 **static** 키워드가 붙은 변수이며, 값이 공유된다.
- 프로그램이 실행되면 메모리에 적재되고, 프로그램이 종료되면 메모리 공간에서 소멸한다.

```
public class OneUpTest2 {  
  
    static int x = 1; //정적 변수  
  
    public static void main(String[] args) {  
        //oneUp() 메서드 호출  
        System.out.println(oneUp()); //2  
        System.out.println(oneUp()); //3  
  
        //x값 호출  
        System.out.println(x);  
    }  
  
    public static int oneUp() { //x - 매개 변수, 지역 변수  
        x = x + 1;  
        return x;  
    }  
}
```



Math 클래스의 내장 매서드

● static(정적) 매서드

Module java.base
Package java.lang

Class Math

java.lang.Object
java.lang.Math

```
public final class Math  
extends Object
```

The class Math contains methods for performing

static 메서드이므로 클래스이름으로
직접 접근(new 객체 생성하지 않음)
사용 예) Math.abs(-4)

Fields

| Modifier and Type | Field | Description |
|-------------------|-------|------------------------------------|
| static double | E | The double value that is closer th |
| static double | PI | The double value that is closer th |

Method Summary

| All Methods | | Static Methods | Concrete Methods |
|-------------------|----------------------------|----------------------|------------------|
| Modifier and Type | Method | Description | |
| static double | <code>abs(double a)</code> | Returns the absolute | |
| static float | <code>abs(float a)</code> | Returns the absolute | |
| static int | <code>abs(int a)</code> | Returns the absolute | |
| static long | <code>abs(long a)</code> | Returns the absolute | |



메서드로 배열 전달

■ 메서드의 매개변수로 배열을 전달

```
public class ArrayTest {  
    public static int add(int[] score) { // 배열이 매개변수  
        int sum = 0;  
        for(int i=0; i<score.length; i++) {  
            sum += score[i];  
        }  
        return sum;  
    }  
  
    public static void main(String[] args) {  
        int[] numbers = {1, 2, 3, 4};  
        int result = add(numbers);  
        double avg = (double)result / numbers.length;  
  
        System.out.println("합계 : " + result);  
        System.out.println("평균 : " + avg);  
    }  
}
```



Math 클래스

● Math 클래스의 주요 메서드 - 실습

```
int v1 = Math.abs(-10);    //절대값  
System.out.println("v1 = " + v1);
```

```
long v2 = Math.round(5.6); //반올림  
System.out.println("v2 = " + v2);
```

```
double v3 = Math.floor(5.9);  
System.out.println("v3 = " + v3); //버림
```

```
int max = Math.max(10, 20);  
System.out.println("max = " + max); //최대값
```

```
double rand = Math.random();  
System.out.println("rand = " + rand); //난수 값(0.0 <= rand <1.0)
```

```
int dice = (int)(Math.random()*6) + 1;  
System.out.println("주사위 눈 : " + dice);
```

```
v1 = 10  
v2 = 6  
v3 = 5.0  
max = 20  
rand = 0.5067546025032655  
주사위 눈 : 3
```



Math 클래스

● Math.random() 사용하기

```
System.out.println("=== 주사위 10번 던지기 ===");
for(int i=1; i < 11; i++) {
    int dice = (int) ((Math.random()*6)+1);
    System.out.print(dice + " ");
}
System.out.println();

System.out.println("=== 문자열 랜덤하게 뽑아내기 ===");
String[] word = {"나", "너", "우리", "세계", "우주"};
//System.out.println(word[0]);
//System.out.println(word[2]);
//word[rnd] - 인덱스 변수 필요
int rnd = (int)(Math.random()*word.length);
System.out.println(word[rnd]);
```

```
=== 주사위 10번 던지기 ===
6 4 2 6 4 2 5 1 4 4
=== 문자열 랜덤하게 뽑아내기 ===
우주
```



Math 클래스

● Lotto 복권 프로그램

```
public class LottoTest {  
  
    public static void main(String[] args) {  
        // 로또 복권 추천  
        int[] lotto = new int[6];  
  
        //번호 생성  
        lotto[0] = (int)(Math.random()*45) + 1;  
        //System.out.println(lotto[0]);  
  
        for(int i=0; i<lotto.length; i++) {  
            lotto[i] = (int)(Math.random()*45) + 1;  
            //중복 번호 제거  
            for(int j = 0; j < i; j++) {  
                if(lotto[i] == lotto[j])  
                    i--;  
            }  
        }  
    }  
}
```



Math 클래스

● Lotto 복권 프로그램

```
        /*
        41 31 31 13 7  -> 31이 중복인 경우
        i=0          41
        i=1 j=0      31
        i=2 j=1      31 중복 -> i-- (다시 생성)
        i=2 j=2
        ...
        */
    }

    //번호 출력
    for(int i=0; i<lotto.length; i++) {
        System.out.print(lotto[i] + " ");
    }
}
```



● 영어 타자 게임

영어타자 게임, 준비되면 엔터

문제1

moon

moon

통과!

문제2

tree

tree

통과!

문제3

moon

오타! 다시 도전!

문제9

cow

cow

통과!

문제10

mountain

mountain

통과!

게임 소요 시간35초입니다.

- 게임이 시작되면 영어 단어가 화면에 표시된다.
- 사용자는 최대한 빠르고 정확하게 입력해야 한다.
- 바르게 입력했으면 다음 문제로 넘어가고 "통과"를 출력한다.
- 오타가 있으면 같은 단어가 한 번 더 나온다.
- 타자 게임 시간을 측정한다.



Math 클래스

```
public static void main(String[] args) {
    String[] words = {"river", "mountain", "sky", "earth", "moon",
        "tree", "flower", "cow", "pig", "horse"};
    int n = 1; //문제 번호
    long start, end;
    Scanner scan = new Scanner(System.in);

    System.out.println("영어타자 게임, 준비되면 엔터");
    scan.nextLine();
    start = System.currentTimeMillis(); //게임시작 시간측정
    while(n < 11) {
        int rand = (int)(Math.random()*words.length);
        System.out.println("문제" + n);
        String question = words[rand];
        System.out.println(question); //화면에 문제 표시

        String answer = scan.nextLine();
        if(answer.equals(question)) { //대답이 질문과 같으면
            System.out.println("통과!");
            n++; //통과하면 문제번호 1 증가
        }else {
            System.out.println("오타! 다시 도전!");
        }
    }
    end = System.currentTimeMillis(); //게임종료 시간측정
    System.out.println("게임 소요 시간은 " + (end-start)/1000 + "초입니다.");
    scan.close();
}
```



날짜와 시간 클래스

❖ 날짜와 시간

- java.util.Date - JDK 1.0(1995년)
 - 날짜와 시간을 다룰 목적으로 만들어진 클래스(JDK 1.0)
- java.util.Calendar - JDK 1.1(1997년)
 - Date 클래스를 개선한 새로운 클래스- 여전히 단점이 존재
 - 지금도 많이 사용함
- java.time 패키지 - LocalDate, LocalTime, - JDK 1.8(2014년)



날짜/ 시간 객체

● Date 클래스

```
public class DateTest {  
    public static void main(String[] args) {  
        Date today = new Date(); //오늘 날짜 객체 생성  
        System.out.println(today);  
  
        SimpleDateFormat date = new SimpleDateFormat("yyyy/MM/dd"); //날짜 포맷  
        System.out.println(date.format(today));  
  
        SimpleDateFormat time = new SimpleDateFormat("hh:mm:ss a"); //시간 포맷  
        System.out.println(time.format(today));  
    }  
}
```



Calendar 클래스

- Calendar 클래스

- 추상클래스이며 getInstance()를 통해 구현된 객체를 얻어야 함.
- get()으로 날짜와 시간 필드 가져오기
- set()으로 날짜 설정

Calendar cal = Calendar.getInstance()

int year = cal.get(Calendar.YEAR) // 2022년

theDay.set(2022, 5, 9) //2022. 5. 9

| 필드명(상수) | 설명 | 필드명 | 설명 |
|-------------|-----------|-------------|----------|
| YEAR | 년 | HOUR | 시 |
| MONTH | 월(0부터 시작) | MINUTE | 분 |
| DATE | 일 | SECOND | 초 |
| DAY_OF_WEEK | 요일 | MILLISECOND | 1/1000 초 |



Calendar 클래스

● Calendar 클래스

```
public class CalendarTest {
    public static void main(String[] args) {
        //Calendar 객체 생성
        Calendar cal = Calendar.getInstance();

        System.out.println(cal); //cal 객체 출력
        System.out.println(cal.getTime()); //현재 날짜와 시간

        //현재 시간 - 1970, 1. 1일 자정 이후부터 현재까지 밀리초(ms)로 측정
        System.out.println(cal.getTimeInMillis());
        System.out.println(cal.getTimeInMillis()/(24*60*60*1000)); //일
        System.out.println(cal.getTimeInMillis()/(24*60*60*1000)/365); //년

        //System 클래스로 현재 시간 측정
        long ctime = System.currentTimeMillis();
        System.out.println("현재 시간(밀리초): " + ctime);
        System.out.println("=====");

        //날짜 - 년, 월, 일
        int year = cal.get(Calendar.YEAR);
        System.out.println(year);

        int month = cal.get(Calendar.MONTH) + 1;
        System.out.println(month);
    }
}
```

```
java.util.GregorianCalendar[time:
Fri Apr 04 15:49:02 KST 2025
1743749342878
20182
55
현재 시간(밀리초): 1743749342930
=====
2025
4
4
3
49
2
6
금
```



Calendar 클래스

● Calendar 클래스

```
int date = cal.get(Calendar.DATE);
System.out.println(date);

//시간 - 시, 분, 초
int hour = cal.get(Calendar.HOUR);
System.out.println(hour);

int minute = cal.get(Calendar.MINUTE);
System.out.println(minute);

int second = cal.get(Calendar.SECOND);
System.out.println(second);

//요일 - {1:일, 2:월, 3:화, 4:수, 5:목, 6:금, 7:토}
int day = cal.get(Calendar.DAY_OF_WEEK);
System.out.println(day);

String[] days = new String[] {"일", "월", "화", "수", "목", "금", "토"};
System.out.println(days[day-1]);
System.out.println("=====");
```



Calendar 클래스

● Calendar 클래스

```
//시간 차이 계산
Calendar theDay = Calendar.getInstance();
Calendar today = Calendar.getInstance();

theDay.set(2025, 3, 29); //특정 날짜 설정
today.set(2025, 4, 5);

System.out.println(theDay.getTime()); //theDay 날짜 가져오기

//시간 차이 계산
long passedTime = today.getTimeInMillis() - theDay.getTimeInMillis();
System.out.println(passedTime);

//초를 일로 환산
passedTime = passedTime / (24*60*60*1000);
System.out.println(passedTime + "일이 지났습니다.");
}
```

```
Tue Apr 29 15:49:02 KST 2025
518400000
6일이 지났습니다.
```



날짜/ 시간 객체

● LocalDate, LocalTime, LocalDateTime 클래스

```
public class LocalDateTimeTest {  
  
    public static void main(String[] args) {  
        //현재 날짜와 시간 객체 생성  
        LocalDateTime now = LocalDateTime.now();  
        System.out.println(now);  
  
        //날짜와 시간 포맷 설정  
        DateTimeFormatter dateTime = DateTimeFormatter.ofPattern("yyyy/MM/dd HH:mm:ss a");  
        System.out.println(now.format(dateTime));  
  
        //특정 날짜 지정하기  
        LocalDate theDay = LocalDate.of(2023, 8, 1);  
        LocalDate today = LocalDate.now();  
  
        System.out.println(today);  
        System.out.println(theDay);  
  
        //지나온 날짜 계산하기  
        long passedTime = ChronoUnit.DAYS.between(theDay, today);  
        System.out.println(passedTime + "일 지났습니다.");  
    }  
}
```



상수 사용하기

■ 상수 정의 및 사용

```
public class UsingDefine {  
    //상수 정의  
    public static final int MAX_NUM = 100;  
    public static final int MIN_NUM = 1;  
}
```

```
public class ConstantEx {  
  
    public static void main(String[] args) {  
        //상수 사용하기  
        //new를 사용하지 않고 클래스로 직접 접근함  
        int maxV = UsingDefine.MAX_NUM;  
        int minV = UsingDefine.MIN_NUM;  
  
        System.out.println(maxV);  
        System.out.println(minV);  
    }  
}
```



열거 타입

■ 열거 타입

한정된 값인 열거 상수 중에서 하나의 상수를 저장하는 타입이다.

```
public enum Season {  
    봄,  
    여름,  
    가을,  
    겨울  
}
```

```
Season season = null;
```

```
season = Season.여름
```



열거 타입

■ 열거 타입

```
public class SeasonTest {  
  
    public static void main(String[] args) {  
        Season season = null;  
        season = Season.여름;  
  
        switch(season) {  
            case 봄:  
                season = Season.봄;  
                break;  
            case 여름:  
                season = Season.여름;  
                break;  
            case 가을:  
                season = Season.가을;  
                break;  
            case 겨울:  
                season = Season.겨울;  
                break;  
        }  
        System.out.println("현재 계절은 " + season + "입니다.");  
  
        if(season == Season.여름) {  
            System.out.println("무더위와 장마가 옵니다.");  
        }else {  
            System.out.println("무더위와 장마가 별로 없습니다.");  
        }  
    }  
}
```



열거 타입

■ 열거 타입

```
enum Level{ //열거형 상수
    LOW,
    MEDIUM,
    HIGH
}

public class EnumLevel {
    public static void main(String[] args) {
        Level level = Level.HIGH; //상수이므로 new를 사용하지 않음

        switch(level) {
            case LOW:
                System.out.println("Low level");
                break;
            case MEDIUM:
                System.out.println("Medium level");
                break;
            case HIGH:
                System.out.println("High level");
                break;
            default:
                System.out.println("레벨이 없습니다.");
                break;
        }
    }
}
```



Enum 예제

```
public class EnumWeek {  
  
    public static void main(String[] args) {  
  
        Week today = null;  
  
        Calendar cal = Calendar.getInstance();  
        int week = cal.get(Calendar.DAY_OF_WEEK);  
        //System.out.println(week);  
  
        switch(week) {  
        case 1:  
            today = Week.SUNDAY; break;  
        case 2:  
            today = Week.MONDAY; break;  
        case 3:  
            today = Week.TUESDAY; break;  
        case 4:  
            today = Week.WEDNESDAY; break;  
        case 5:  
            today = Week.THURSDAY; break;  
        case 6:  
            today = Week.FRIDAY; break;  
        case 7:  
            today = Week.SATURDAY; break;  
        }  
        //switch 닫기
```

```
        System.out.println("오늘 요일 : " + today);  
  
        if(today == Week.SUNDAY) {  
            System.out.println("일요일에는 영화를 보러 갑니다.");  
        }else {  
            System.out.println("열심히 프로그램 코딩합니다.");  
        }  
    }  
} //main 닫기
```



실습 문제 1 – 배열

■ 실습 예제

배열 길이가 5인 정수 배열을 선언하고, 1~10중 짝수만을 배열에 저장한 후 그 합과 평균을 계산하세요.

☞ 실행 결과

```
총합: 30  
평균: 6.0
```



실습 문제 2 – 함수

- 실습 예제

절대값을 계산하는 함수 `myAbs()`를 정의하고 아래와 같이 실행하세요.

[파일 이름: `MyAbs.java`]

☞ 실행 결과

```
myAbs(-4) = 4  
myAbs(4) = 4
```

