**배열**

**배열 개요**

배열은 동일한 타입의 데이터를 메모리에 연속적으로 나열한 데이터입니다.

배열의 3가지 요소

배열의 이름, 타입, 요소의 개수

배열의 선언 방법

|  |  |
| --- | --- |
| 항목 | 설명 |
| new 연산자 | 타입[] 배열이름 = new 타입[크기] ;  int[] exam = new int[20] ;  String[] soshi = new String[9] ; |
| 초기화 기법 | 해당 배열의 내용을 직접 초기화시키는 방법입니다.  타입[] 배열이름 = {값1, 값2, 값3, … };  String[] soshi = {"태연", "효연", "티파니"} ; |

|  |
| --- |
| 배열의 특징 |
| 배열의 요소 번호는 0부터 시작합니다.  요소를 접근하고자 할 때 대괄호를 사용합니다.  요소 1개는 변수 1개와 동급 레벨입니다.  요소의 개수는 배열이름.length라는 메소드를 사용하면 됩니다.  자바에서 대괄호가 나오면 무조건 배열입니다. |

|  |  |
| --- | --- |
| 항목 | 설명 |
| 장점 | 변수 1개로 표현이 가능합니다.  반복문(for 구문)이 가능합니다. |
| 단점 | 크기가 정해지고 나면 더 이상 수정이 불가능합니다. |

배열의 초기 값

일반 변수와는 다르게 배열은 타입마다 디폴트 초기 값이 존재합니다.

|  |  |  |  |
| --- | --- | --- | --- |
| 타입 | 설명 | 타입 | 설명 |
| byte, short, int, long | 0 | float, double | 0.0 |
| char | 공백 문자 | boolean | false |
| 참조 자료형 | null |  |  |

배열 관련 OX 문제

|  |
| --- |
| 배열의 특징 |
| OX 문제 :  배열 요소 1개는 일반 변수와 동등하게 취급하면 됩니다.  배열의 요소 번호는 1부터 시작합니다.  배열을 생성하는 방법은 크게 ( 2 ) 가지가 있습니다.  하나는 ( new 연산자 )을 이용하는 방법이고, 다른 하나는 ( 초기화 기법 )입니다.  요소 갯수를 구하는 방법은 ( 배열이름.length )라는 메소드를 사용하면 됩니다. |

다음의 result는 얼마인가?

int x = 3 ;

int[] arr = new int[2] ;

double result = Math.pow( arr.length, x ) ;

다차원 배열

대괄호의 개수가 2개 이상인 배열을 말합니다.

예를 들어서, 대괄호가 2개이면 이차원 배열이라고 합니다.

일반적인 개발 과정에서는 이차원 배열 정도까지 많이 사용합니다.

|  |  |
| --- | --- |
| 항목 | 설명 |
| new 연산자 | int[][] arr = new int[2][3] ; //2행 3열의 배열 선언  arr[0][0] = 1 ;  int x = arr[0][1] + arr[1][0] ; // x = 2 + 4 = 6 |
| 초기화 기법 | int[][] brr = {{1, 2, 3}, {4, 5}} ;  가장 외부의 중괄호는 초기화 기법을 위한 중괄호입니다.  내부에 들어 있는 중괄호는 하나의 행을 의미하는 중괄호입니다. |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0열 | 1열 | 2열 |
| 0행 | 1 | 2 | 3 |
| 1행 | 4 | 5 | 6 |

|  |
| --- |
| 이차원 배열의 특징 |
| 중첩 for 구문을 사용할 수 있습니다.  대괄호가 2개 사용되면 값을 의미하고, 1개이면 메모리 주소를 의미합니다.  ▶ arr[0][0] = 1 ; (값 의미) , arr[1] = 2 ; (번지수 의미)  행의 개수는 배열이름.length로 구할 수 있습니다.  열의 개수는 배열이름[i].length로 구할 수 있습니다.(단, i는 행의 번호입니다.) |

**메소드**

**메소드 개요**

입력된 데이터를 가공/데이터 연산을 수행한 다음 결과물을 되돌려 주는 역할을 하는 소스 코드를 말합니다.

자바에서는 함수라는 이름 대신 메소드라는 용어를 사용합니다.(ex. excel sum 등)

메소드는 재사용성(반복적이고 패턴이 동일한 코드에 대한 중복 지양)을 위하여 사용합니다.

"매개 변수"란 메소드의 외부에서 메소드 내부로 넣어 주는 데이터를 의미합니다.

"매개 변수"는 인자, 인수, 파라미터 등의 용어로 사용이 됩니다.

메소드의 3가지 요소

메소드 이름, 매개 변수, 반환 타입

메소드 작성법

2개 이상의 매개 변수는 콤마를 이용하여 열거합니다.

매개 변수는 필수 사항이 아니므로 명시하지 않을 경우 소괄호만 명시하면 됩니다.

return이라는 구문은 메소드를 종료함을 의미하므로, return 구문 이후에 추가 코딩을 하면 안됩니다.(반환\_타입을 작성했으면 return 반환할 값을 추가 작성한다.)

메소드가 값을 반환하지 않을 경우에는 키워드 void를 명시하면 되고, return 구문을 사용하지 않습니다.

void는 no response, empty, vacant 등의 의미로 사용됩니다.

|  |
| --- |
| 메소드 작성 방법 |
| [public] [static] 반환\_타입 메소드\_이름([매개변수리스트]){  //할 일을 명시  [return 반환할\_값 ;]  } |

|  |
| --- |
| 메소드 작성 가이드 라인 |
| 구현한 메소드를 사용하는 동작을 "호출"이라는 용어를 사용합니다.  실 매개 변수와 형식 매개 변수는 개수가 동일해야 하고, 타입도 동일해야 합니다.  실 매개 변수와 형식 매개 변수의 이름은 서로 달라도 상관이 없습니다.  메소드의 위치는 반드시 클래스 내부에서만 가능합니다.  메소드는 main 메소드와 동급 레벨에서 작성해야만 합니다.(main 메소드 내부도 오류)  변수는 소괄호가 없고, 메소드는 반드시 소괄호가 있습니다. |