JAVA 입문 : 이론과 실습

제 2장 어휘구조와 자료형

목차

어휘구조와 자료형

- 어휘
  - 프로그램을 이루고 있는 기본 소자
    - 토큰(token)이라 부름
    - 문법적으로 의미 있는 최소의 단위
  - 컴파일 과정의 첫번째 단계(어휘분석)에서 처리

- 자료형
  - 자료 객체가 갖는 형
  - 구조, 개념, 값, 연산자를 정의

토 큰

- 문법적으로 의미 있는 최소의 단위
  - if (i<100) sum+ =i;

    토큰 : if, (, i, <, 100,), sum, + =, i, ; (10개)

지정어 (keyword)

- 자바 지정어 (50개)

abstract	continue	for	new	switch
assert	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient*
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const*	float	native	super	while

명칭 (identifier) [1/2]

- **자료의 항목**을 식별하기 위하여 붙이는 이름
  (변수, 상수, 배열, 클래스, 메소드, 레이블)
- 명칭의 형태
  - 대소문자를 구분, 문자나 밑줄(_) 문자, 혹은 $로 시작

바른 명칭 : sum, sum1, _sum, $sum, money_sum, MoneySum
틀린 명칭 : 1sum, sum!, #sum, Money Sum, package

  - 합성 명칭인 경우 :
    - lowerCamelCase : billingAddress
    - UpperCamelCase : BillingAddress

명칭 (identifier) [2/2]

- 자바 문자 집합
  - 유니코드(Unicode)
    - 문자 표현 : 16 Bit
    - 세계 모든 언어 표현

```
static final double π = 3.14159265358979323846;
```

  - 대소문자 구별

```
int X;
int x;
int MyVar, myvar, myVar, Myvar;
```

리터럴(literal) [1/4]

- 자신의 표기법이 곧 자신의 값이 되는 상수
  - 정수형 상수, 실수형 상수, 문자형 상수, 스트링 상수
  - true, false, null
- 정수형 상수
  - 10진수, 8진수, 16진수

```
10진수 : 15, 255, 65535
8진수 : 017, 0377, 0177777
16진수 : 0xf, 0xff, 0xffff
```

  - default : 32 bit
  - 접미어 -L, -l : 64 bit
    - 16, 26l, 45L

리터럴(literal) [2/4]

- 실수형 상수
  - 지수(exponent) 의 유무에 따라
    - 고정소수점 수 : 1.414, 3.1415924, 0.00001
    - 부동소수점 수 : 0.1414e01, 0.1414E1, 5E-5f
  - 정밀도(precision) 에 따라
    - float 형 : 접미어 -f, -F
      - 3.14f, 0.526f
    - double 형 : default
      - 3.14, 0.526

리터럴(literal) [3/4]

- 논리형 상수
  - binary value
    - true, false
- 문자 리터럴
  - 단일 인용부호(single quote) 사이에 표현
    - 'a' '\n'
  - escape sequence : 특수한 문자를 표현

\n	newline(\u000A)
\t	tab(\u0009)
\b	backspace(\u0008)
\r	return(\u000D)
\f	form feed(\u000C)
\ \	backslash itself(\u005C)
\ '	single quote(\u0027)
\ “	double quote(\u0022)
\ ddd	8진수 문자이며, 여기서 d는 0-7 중의 하나이다.

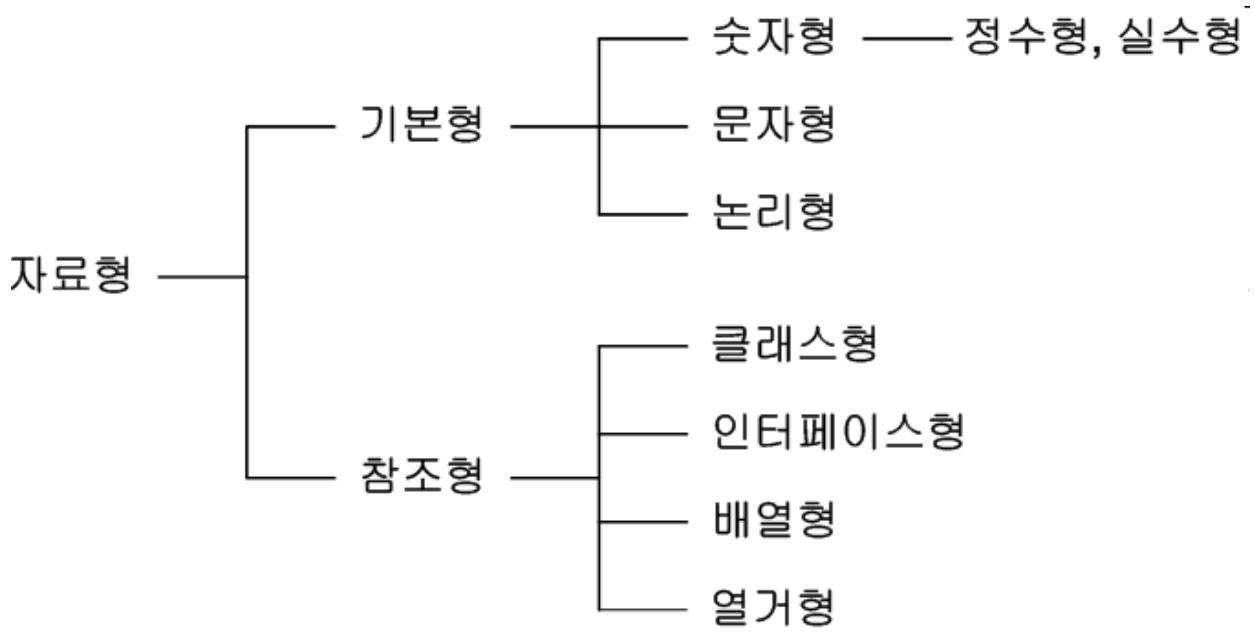
리터럴(literal) [4/4]

- 스트링 리터럴
  - 이중 인용부호(double quote) 사이에 표현
    - "I am a string"
    - " \ ""
  - java.lang.String 클래스의 객체로 취급
- 객체 참조 리터럴
  - null
    - 아무 객체도 가리키지 않는 상태
    - 초기화에 사용
    - 부적당하거나 객체를 생성할 수 없는 경우 사용

주석(comment)

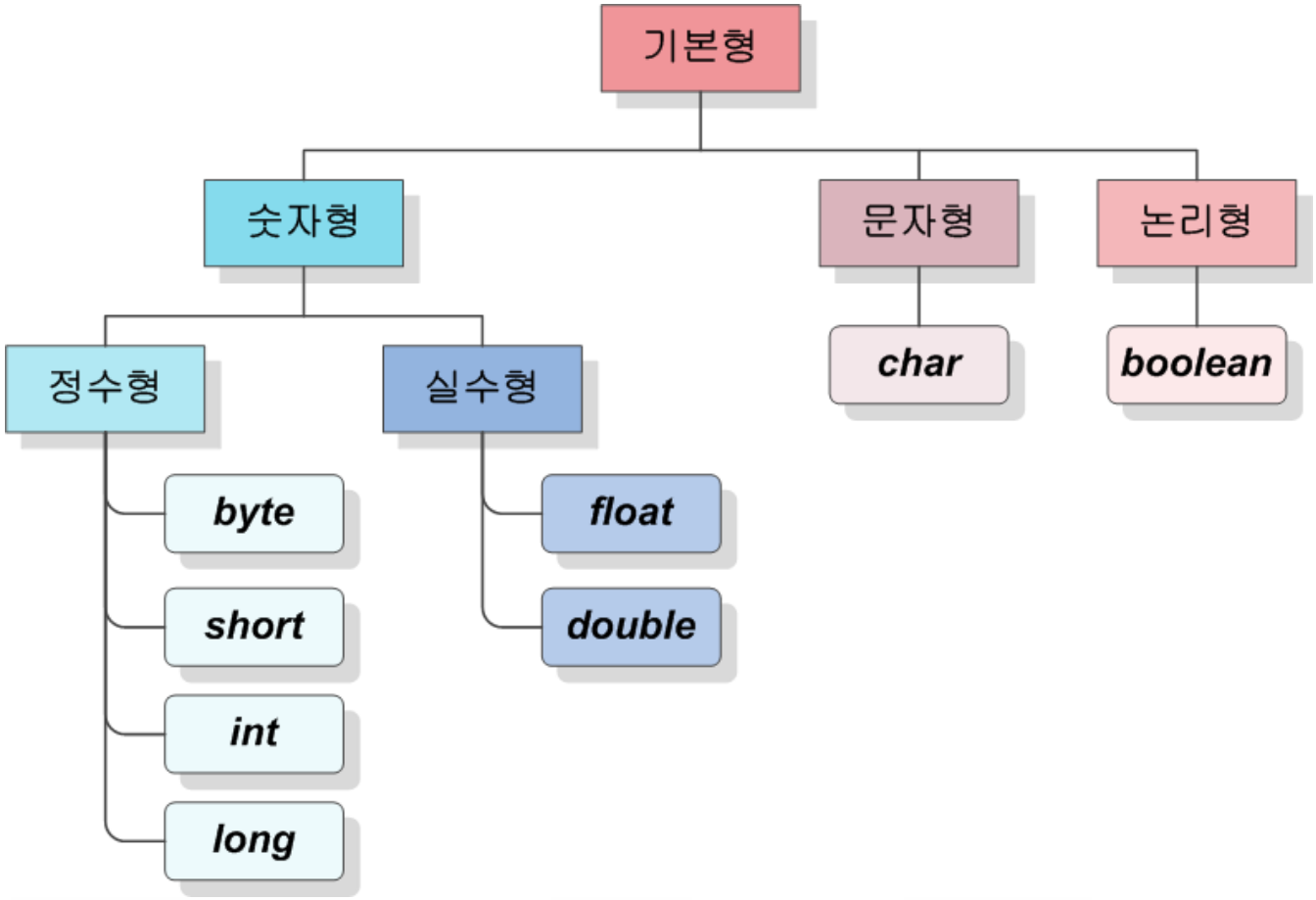
- 프로그램 설명을 위한 문장
  - 프로그램 유지보수에 중요
  - 프로그램의 실행에는 무관
- 주석 종류
  - **Line Comment** : // comment
    - //부터 새로운 줄 전까지 주석으로 간주
  - **Text Comment** : /* comment */
    - /*와 다음 */ 사이의 모든 문자들은 주석으로 간주
  - **Documentation Comment** : /** comment */
    - /**와 다음 */ 사이의 모든 문자들은 주석으로 간주
    - javadoc 도구를 이용하여 API 문서 작성에 사용

자료형(data type)



- 자료형은 구조 및 개념, 값의 범위, 연산 등을 정의.

기본형(primitive type) [1/4]



기본형(primitive type) [2/4]

- 정수형(integer type)

유형	크기	최소값	최대값
byte	8bit	-128	127
short	16bit	-32768	32767
int	32bit	-2147483648	2147483647
long	64bit	-9223372036854775808	9223372036854775807

  - C/C++와는 달리 **unsigned는 지원하지 않음**

- 실수형(real type)
  - float(32비트), double(64비트)
  - 실수의 표현 방법과 실수 연산은 **IEEE 754** 표준 따름

기본형(primitive type) [3/4]

- 문자형(character type)
  - Unicode 사용
- 논리형(boolean type)
  - true와 false 중 하나의 값만을 가지는 자료형
  - **숫자 값 가질 수 없음**
  - 다른 자료형으로 변환되지 않음

기본형(primitive type) [4/4]

- 자료형에 대한 초기값
  - 초기값(initial value)이 명시되지 않으면, 해당하는 field type에 따라 default 초기값을 할당

자료형	기본 표준값	초기값
byte	zero	(byte) 0
short	zero	(short) 0
int	zero	0
long	zero	0L
float	positive zero	0.0f
double	positive zero	0.0d
char	null 문자	'₩u0000'
boolean	false	
reference	null	

  - local variable, static initializer : default 초기값을 할당 않음

참조형(reference type)

- 객체를 가리키는 형
- 배열
  - 같은 형의 여러 값을 저장하는데 이용하는 자료형
  - C/C++와 달리 객체로 취급
- 클래스형
  - 클래스 이름
  - 객체를 가리키는 참조형
- 인터페이스형
  - 인터페이스 이름
  - 인터페이스를 구현한 객체에 대한 참조
- 열거형
  - 여러 개의 숫자 상수만을 가진 특별한 형태의 클래스형

배열형 [1/6]

- 순서가 있고 같은 형의 원소들의 모임
- 배열 사용 단계
  - 배열 변수 선언
    - 배열 변수 : 배열을 가리키는 변수(참조변수)
    - 배열에 속한 원소의 형과 차원 등을 명시

```
int[]       vector;
short       matrix[][];
Object[]    myArray;
```

  - 배열 객체 생성
    - new 연산자를 통해 동적으로 생성

배열형 [2/6]

- 배열 객체 생성
  - new 연산자를 통해 동적으로 생성

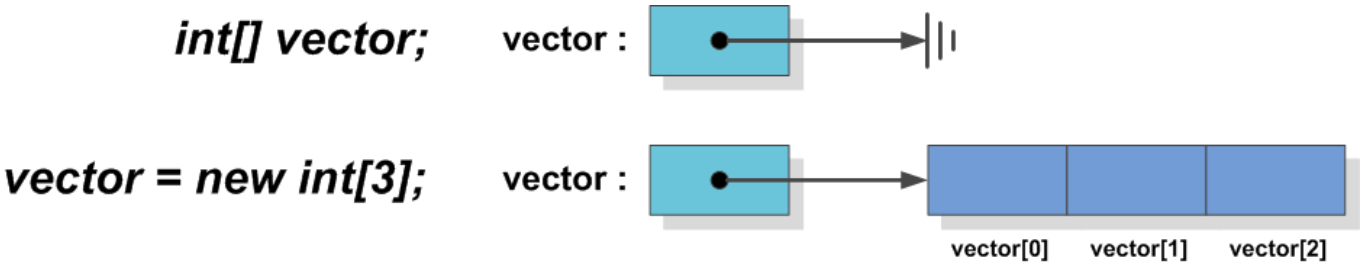
```
vector1 = new int[5];
vector2 = {1, 2, 3, 4, 5, 6, 7, 8, 9};
matrix = new short[10][100];
myArray = new Point[3];
```

- 배열 변수 선언과 동시에 배열 할당

```
int[] day = new int[31];
```

배열형 [3/6]

♣ 참고: 배열 변수 선언과 생성에 따른 메모리 구조



배열형 [4/6]

- 배열에 값 저장
  - 배열의 인덱스는 0부터 시작
  - length를 통한 배열의 길이
  - 범위 초과 : **IndexOutOfBoundException**이 발생

```
int a[] = new int[50];
for(int i=0; i < a.length; i++)
        a[i] = i;
```

  - [예제 2.8 - ArrayType.java]

배열형 [5/6]

- 배열의 배열(array of array)
  - 배열의 원소가 다시 배열이 되는 배열
  - 다차원 배열

```
int[][] matrix = new int[3][3];
```

  - 첫번째 차원의 크기는 반드시 명시

```
int[][] matrix = new int[3][];
for (int i=0; i < matrix.length; i++)
        matrix[i] = new int[3];
```

  - [예제 2.9 - ArrayOfArray.java]

배열형 [6/6]

[예제 2.8- ArrayOfArray.java]

```
public class ArrayOfArray {
    public static void main(String[] args) {
        int[][] matrix = new int[3][];                          // declaration
        int i, j;

        for (i = 0; i < matrix.length; i++)                     // creation
            matrix[i] = new int[i+3];
            for (i = 0; i < matrix.length; i++)                 // using
                for (j = 0; j < matrix[i].length; j++)
                    matrix[i][j] = i*matrix[i].length + j;
                    for (i = 0; i < matrix.length; i++) {       // printing
                        for (j = 0; j <matrix[i].length; j++)
                            System.out.print(" " + matrix[i][j]);
                            System.out.println();
                        }
    }
}
```

실행 결과 :

```
 0  1   2
 4  5   6   7
 10 11 12 13 14
```

열거형 [1/2]

- 열거형
  - 서로 관련 있는 상수들의 모음을 심볼릭한 명칭의 집합으로 정의
  - 기호 상수(symbolic constant)
    - 명시된 명칭들
  - 순서값은 0부터 시작
  - 정수형으로 교환하여 사용 가능
  - 장점
    - 프로그램 가독성 증가 – 기호 상수 표현
    - 디버깅 용이 – 명칭의 이름 출력
- 열거형 기본 메소드
  - values(): 열거된 모든 원소를 순서대로 반환하는 메소드
  - ordinal(): 원소의 열거된 순서를 정수 값으로 반환하는 메소드
  - valueOf(): 매개변수로 주어진 스트링과 열거형에서 일치하는 이름을 갖는 원소를 반환하는 메소드

열거형 [2/2]

[예제 2.10 - EnumTypeExample.java]

```
enum Color { Red, Green, Blue }
public class EnumTypeExample {
    public static void main(String[] args) {
        for (Color col : Color.values()) {
            System.out.println(col);
        }
        Color c = Color.Red;
        System.out.println(c + "'s value is " + c.ordinal());
        c = Color.valueOf("Blue");
        System.out.println(c + "'s value is " + c.ordinal());
        c = Color.valueOf("Yellow");
        System.out.println(c + "'s value is " + c.ordinal());
    }
}
```

실행 결과 :

```
Red
Green
Blue
Red's value is 0
Blue's value is 2
Exception in thread "main" java.lang.IllegalArgumentException: No enum const
class Color.Yellow
        at java.lang.Enum.valueOf(Unknown Source)
        at Color.valueOf(EnumTypeExample.java:1)
        at EnumTypeExample.main(EnumTypeExample.java:21)
```