[2-3] 다음의 문장에서 리터럴, 변수, 상수, 키워드를 적으시오.

```
int i = 100;
long l = 100L;
final float PI = 3.14f;
```

- 리터럴 : 100, 100L, 3.14f
- 변수 : i, l
- 키워드 : int, long, final, float
- 상수 : PI

[2-5] 다음 문장들의 출력결과를 적으세요. 오류가 있는 문장의 경우, 괄호 안에 '오류'라고 적으시오.

```
System.out.println( "1" + "2" )   → (12      )
System.out.println(true + "" )    → (true    )
System.out.println( 'A' + 'B')    → (131     )
System.out.println('1' + 2)       → (51      )
System.out.println('1' + '2')     → (99      )
System.out.println('J' + "ava" )  → (Java    )
System.out.println(true + null)   → (오류    )
```

[해설] 문자열과 덧셈연산을 하면 그 결과는 항상 문자열이 된다.

```
문자열 + any type → 문자열 + 문자열 → 문자열
any type + 문자열 → 문자열 + 문자열 → 문자열
```

```
"" + 7  →  "" + "7"  →  "7"     // 빈 문자열을 더해서 숫자를 문자열로 변환한다.
7 + ""  →  "7" + ""  →  "7"

7 + " " →  "7" + " "  →  "7 "
" " + 7 →  " " + "7"  →  " 7"

7 + "7"  →  "7" + "7"  →  "77"

7 + 7 + ""  →  14 + ""  →  "14" + ""  →  "14"
"" + 7 + 7  →  "7" + 7  →  "7" + "7"  →  "77"

true + "" → "true" + "" → "true"
null + "" → "null" + "" → "null"
```

하지만 문자와 문자의 덧셈연산의 결과는 int형 정수값이 된다. 왜냐하면 int형보다 작은 타입(byte, char, short)은 int형으로 변환된 후에 덧셈연산이 진행되기 때문이다.
'A'+'B'의 경우, char + char → int + int → int의 과정을 통해 최종결과는 int형 정수값이 된다.

```
'A'+'B' → 65 + 66 → 131      'A'와 'B'의 문자코드의 값은 각각 65와 66이다.
'1'+2   → 49 + 2  → 51       '1'의 문자코드의 값은 49이다.
'1'+'2' → 49 + 50 → 99
```

[2-6] 다음 중 키워드가 아닌 것은?(모두 고르시오)

a. if
b. True
c. NULL
d. Class
e. System

[정답] b, c, d, e
[해설] Java에서는 대소문자를 구별하기 때문에 true는 키워드이지만 True는 키워드가 아니다. 다음은 Java에서 사용하는 키워드이다.

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

[2-7] 다음 중 변수의 이름으로 사용할 수 있는 것은? (모두 고르시오)

a. $ystem
b. channel#5 - 허용하지 않는 특수문자'#'를 사용할 수 없다.
c. 7eleven - 숫자로 시작하면 안된다.
d. If
e. 자바
f. new - 예약어라서 사용할 수 없다.
g. $MAX_NUM
h. hello@com - 허용하지 않는 특수문자'@'를 사용할 수 없다.

[정답] a, d, e, g
[해설] 변수의 이름(identifier)은 다음과 같은 규칙이 있다.

1. 대소문자가 구분되며 길이에 제한이 없다.
   - True와 true는 서로 다른 것으로 간주된다.
2. 예약어를 사용해서는 안 된다.
   - true는 예약어라서 사용할 수 없지만, True는 가능하다.
3. 숫자로 시작해서는 안 된다.
   - top10은 허용하지만, 7up는 허용되지 않는다.
4. 특수문자는 '_'와 '$'만을 허용한다.
   - $harp은 허용되지만, S#arp은 허용되지 않는다.

[2-8] 참조형 변수(reference type)와 같은 크기의 기본형(primitive type)은? (모두 고르시오)

a. int(4 byte)
b. long(8 byte)
c. short(2 byte)
d. float(4 byte)
e. double(8 byte)

[정답] a, d
[해설] 모든 참조형 변수는 4 byte이므로, 크기가 4 byte인 기본형 타입을 고르면 된다.

[2-9] 다음 중 형변환을 생략할 수 있는 것은? (모두 고르시오)

```
byte b = 10;
char ch = 'A';
int i = 100;
long l = 1000L;

a. b = (byte)i;      // int(4byte) → byte(1byte)이므로 반드시 형변환 필요
b. ch = (char)b; // byte(1byte) → char(2byte)이지만 범위가 달라서 형변환 필요
c. short s = (short)ch; // char,short은 2byte이지만 범위가 달라서 형변환 필요
d. float f = (float)l;  // float(4byte)의 범위가 long(8byte)보다 커서 생략가능
e. i = (int)ch; // char(2 byte) → int(4byte)이므로 생략가능
```

[정답] d, e

[2-10] char타입의 변수에 저장될 수 있는 정수 값의 범위는? (10진수로 적으시오)

[정답] 0~65535
[해설] char는 2 byte(2*8=16bit)이므로 '2의 16제곱' 개의 값을 표현할 수 있다.
2의 16제곱은 65536개이며, 0을 포함해야하므로 0~65535(모두 65536개)가 char범위가 된다.

[2-11] 다음중 변수를 잘못 초기화 한 것은? (모두 고르시오)

```
a. byte b = 256; // byte의 범위(-128~127)를 넘는 값으로 초기화 할 수 없음.
b. char c = '';  // char는 반드시 한 개의 문자를 지정해야함
c. char answer = 'no'; // char에 두 개의 문자를 저장할 수 없음.
d. float f = 3.14 // 3.14는 3.14d의 생략된 형태. 접미사f를 붙이거나 형변환필요
e. double d = 1.4e3f; // double(8byte)에 float값(4byte)을 넣는 것이므로 OK
```

[정답] a,b,c,d
[해설] 접미사가 있는 자료형은 long, float, double 모두 세 개의 자료형이며, 접미사는 대소문자를 구별하지 않는다. double은 접미사를 생략할 수 있으므로 float리터럴에는 반드시 접미사를 붙여야한다.

[2-12] 다음 중 main메서드의 선언부로 알맞은 것은? (모두 고르시오)

```
a. public static void main(String[] args)
b. public static void main(String args[])
c. public static void main(String[] arv) // 매개변수 args의 이름은 달라도 됨
d. public void static main(String[] args) // void는 반드시 main앞에 와야 한다.
e. static public void main(String[] args) // public과 static은 위치가 바뀌어도 됨
```

[정답] a,b,c,e
[해설] 배열을 의미하는 기호인 '[]'는 타입 뒤에 붙여도 되고 변수명 뒤에 붙여도 되기 때문에 'String[] args'와 'String args[]'는 같은 뜻이다. 자세한 내용은 '5장 배열(Array)' 에서 자세히 설명할 것이다.

```
[2-13] 다음 중 타입과 기본값이 잘못 연결된 것은? (모두 고르시오)
    a. boolean - false
    b. char - '₩u0000'
    c. float - 0.0 // float는 0.0f가 기본값. 0.0은 0.0d에서 접미사 d가 생략된 것
    d. int - 0
    e. long - 0   // long은 0L이 기본값.
    f. String - "" // String은 참조형 타입. 모든 참조형 타입의 기본값은 null

[정답] c,e,f
[해설] 리터럴의 접미사는 대소문자를 구분하지 않으므로, long의 경우 'L'또는 'l'(소문
자)을 사용할 수 있다. 'l'은 숫자 '1'과 혼동하기 쉬우므로 대문자를 사용하는 것이 좋
다.
```

● 문자열 & 문자의 덧셈연산 규칙
- 문자열과 어떤 타입이든 덧셈 연산을 하면 그 결과는 항상 문자열이 된다 (그 어떤 타입을 문자열로 바꾼 다음 더해준다)
- 문자와의 덧셈 연산의 결과는 int형 정수값이 된다. int형보다 작은 타입(byte, char, short)는 int형으로 변환된 후에 덧셈연산이 진행되기 때문이다(문자일 경우 해당 문자코드의 값으로 변환 후 덧셈연산 수행)

● 모든 참조형 변수의 크기는 4byte다.

● byte와 short는 음수와 양수를 범위 안에 모두 가지고 있는 반면 char은 양수만 범위 안에 가지고 있기 때문에, 범위가 달라서 형변환을 생략할 수 없다.

● String은 "" 빈칸으로 저장할 수 있는 반면 (빈 문자열: empty string), char은 반드시 한 개의 문자를 지정해야 한다.

● 접미사가 있는 자료형은 long, float, double 모두 세 개의 자료형이며, double만 접미사를 생략할 수 있으므로 long과 float에는 반드시 접미사를 붙여 주어야 한다. 접미사는 대소문자를 가리지 않고 모두 사용 가능하다.

● 자료형의 기본 값
- 참조형 타입: 모든 참조형 타입의 기본 값은 null이다.
ㄴ String: null
- boolean: false
- char: '₩u0000' - '₩u0000'은 유니코드의 첫 번째 문자로써 아무런 문자도 지정되지 않은 빈 문자이다.
- int: 0
- long: 0L/0l
- float: 0.0f/0.0F
- double: 0.0(0.0d/0.0D 접미사 생략 가능)

Resource: 자바의 정석