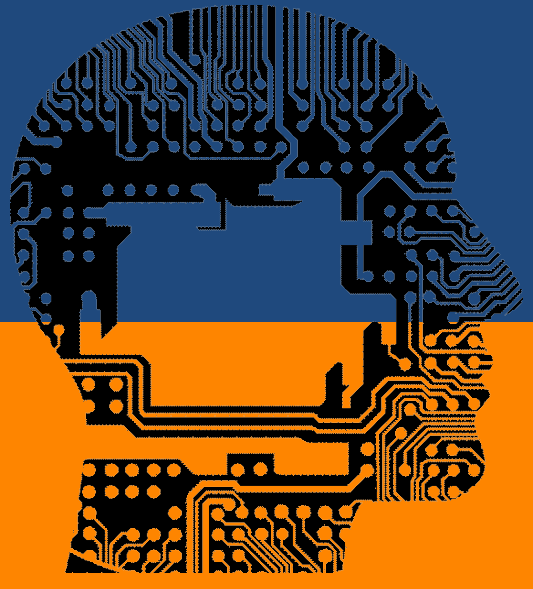
한국IT진흥부설
정보보호교육학원 아이섹

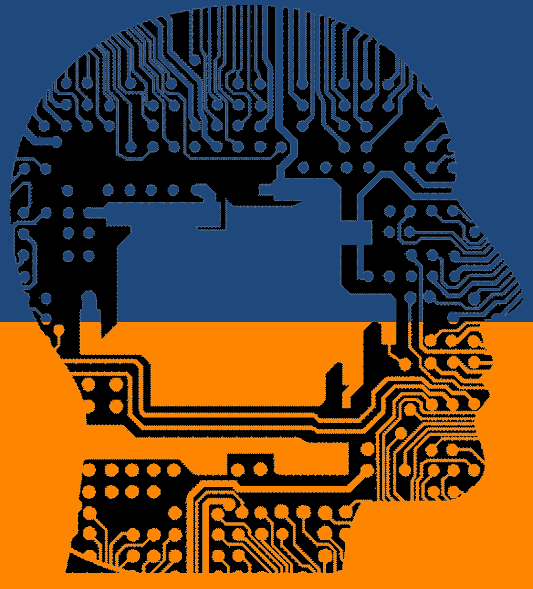


자바 프로그래밍 기초

데이터 타입

By SoonGu Hong(Kokono)

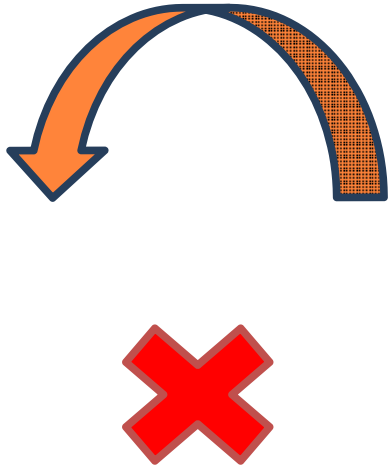
한국IT진흥부설 정보보호교육학원 아이섹



자바 프로그래밍 기초

1. 기본 데이터 타입 (Primitive Data Type)

1-1. 데이터 타입(자료형)이란?



< 변수(Variable) >

< 자료(data) >

만약에 국물을 평평한 접시에 담으면 흘러넘쳐버리죠?
또한 뚝배기에 케이크를 담으면 케이크가 부서지기 쉽겠죠?
같은 이유로 변수도 데이터의 형태에 맞는 데이터 타입을 지정해줘야 합니다.

1-2. 자바 기본 데이터 타입의 종류

	유형	크기	범 위		초기값
정수형 Integral	byte	1 byte	(-128 ~ 127)	$-2^7 \sim 2^7 - 1$	0
	short	2 byte	(-32,768 ~ 32,767)	$-2^{15} \sim 2^{15} - 1$	0
	int	4 byte	(-2,147,483,648 ~ 2,147,483,647)	$-2^{31} \sim 2^{31} - 1$	0
	long	8 byte	(-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807)	$-2^{63} \sim 2^{63} - 1$	0L
실수형 Floating-Point	float	4 byte	(7자리의 소수)	$\pm 1.401e^{-45}F \sim 3.402e^{+38}F$	0.0F
	double	8 byte	(15자리의 소수)	$\pm 4.940e^{-324} \sim 1.797e^{+308}$	0.0D
논리형 Logical	boolean	1 bit	true 또는 false		false
문자형 Textual	char	2 byte	₩u0000 ~ ₩uFFFF		'₩u0000'
	String	가변적	각 위치에서 ₩u0000 ~ ₩uFFFF		null
참조 타입					null

1-3-1. 정수형(Integral)

< 정수형 데이터 저장 예시 >

ex) byte a = 8;

0	0	0	0	1	0	0	0

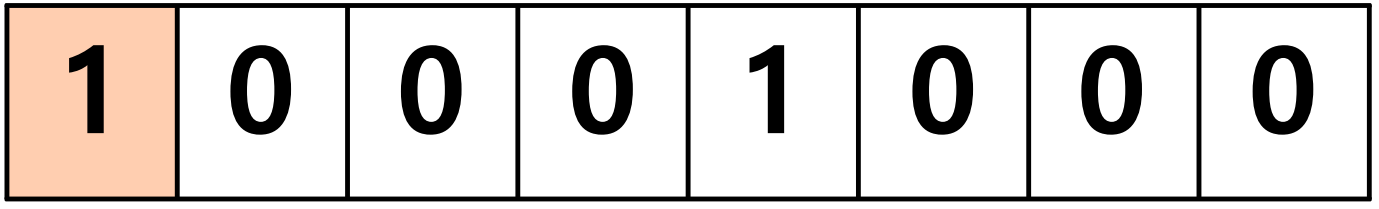
byte타입은 메모리에 1바이트(8비트)만큼의 용량을 할당받습니다.

따라서 2진수정수 8자리를 담을 수 있습니다. 그러면 최대 저장 숫자는 11111111이 되기 때문에 0~255를 표현해야 하는데 실제로는 -128~127까지로 되어있죠?? 왜 그럴까요?

맞습니다! 음의 정수도 표현해야 합니다!

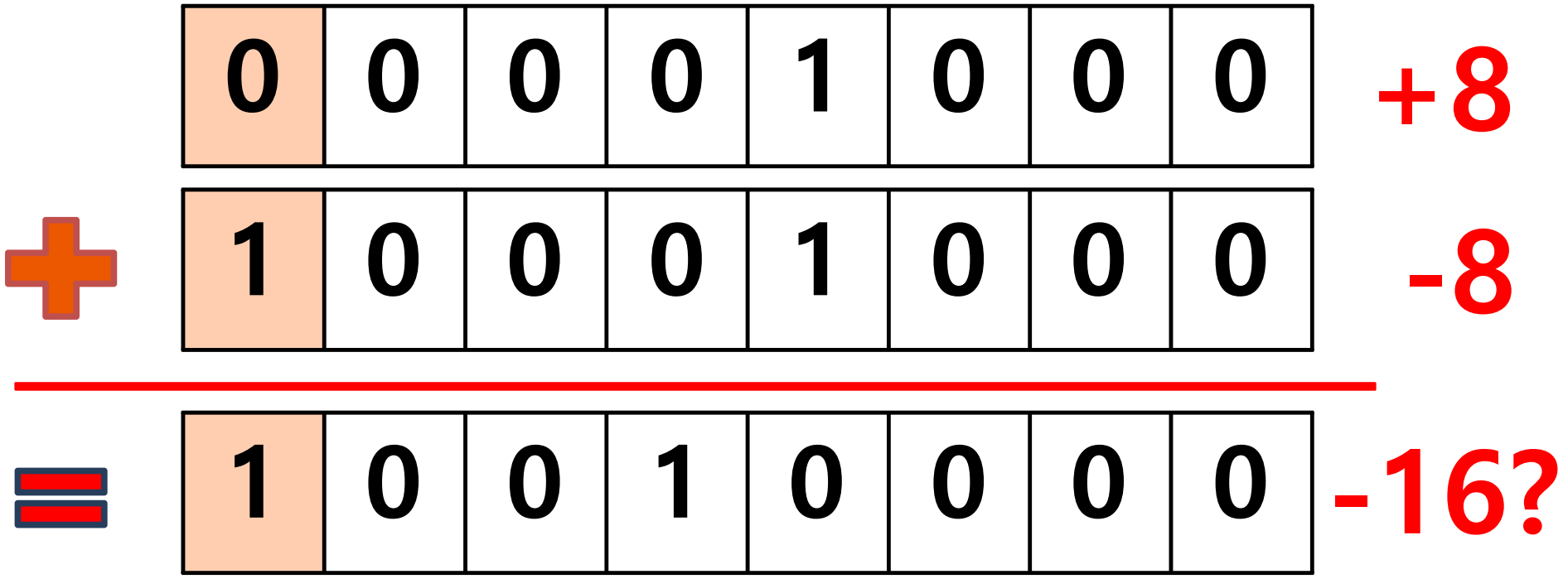
1-3-2. 컴퓨터의 정수 표현 방식

가장 왼쪽 비트를 MSB(Most Significant Bit)라 부릅니다.
0이면 양수를 의미하며 1이면 음수를 의미합니다.
그러면 음수 8은 어떻게 표현할까요??



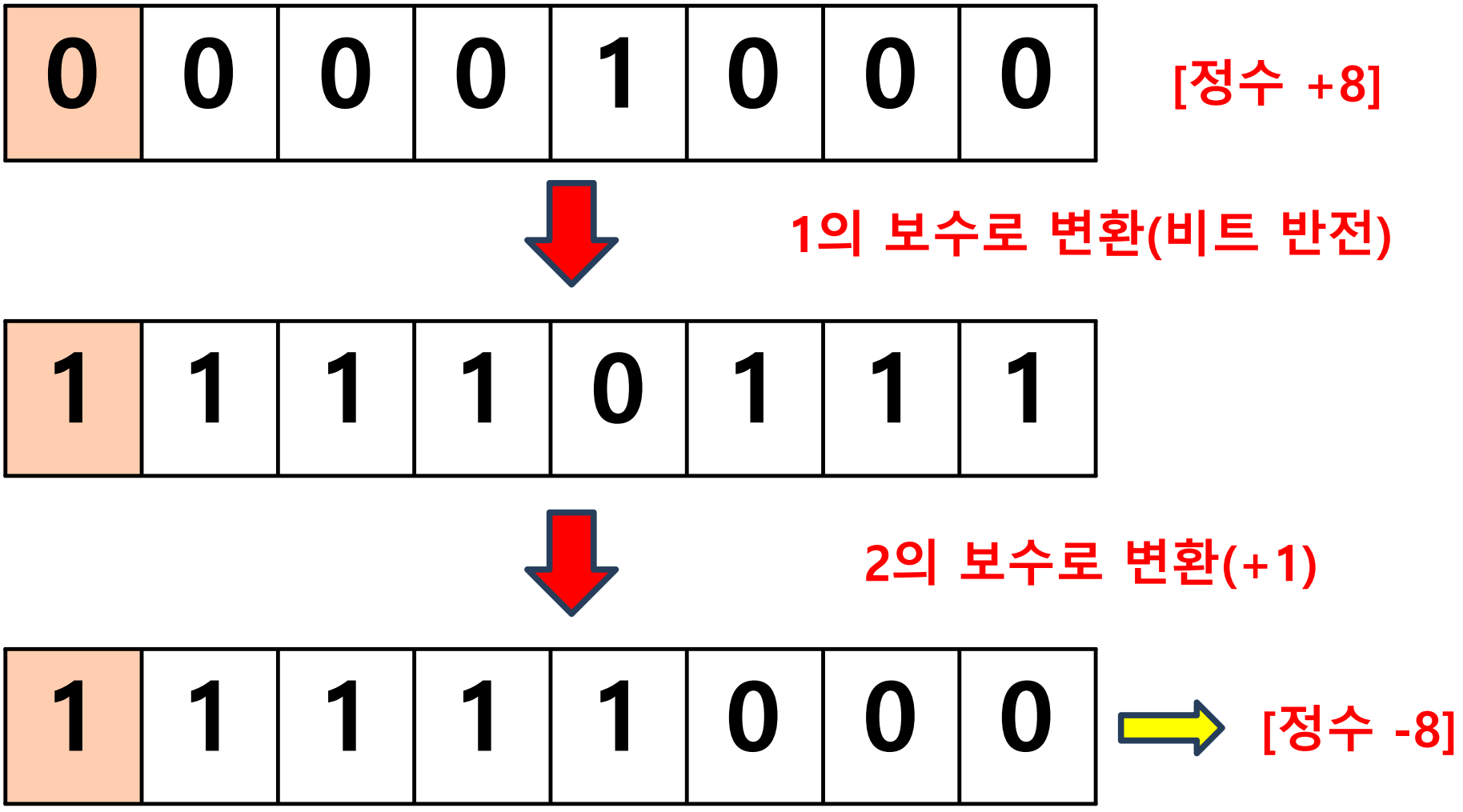
-8?

1-3-2. 컴퓨터의 정수 표현 방식



우리가 단순히 생각했을 때의 방법으로 음수를 표현하게 되면 위와 같이 양수8과 음수8을 더했을 때 0이 아닌 -16이라는 이상한 값이 나오게 됩니다. 따라서 컴퓨터는 음수를 표현할 때 **2의보수**를 취하여 표현합니다.

1-3-3. 컴퓨터의 음수 표현

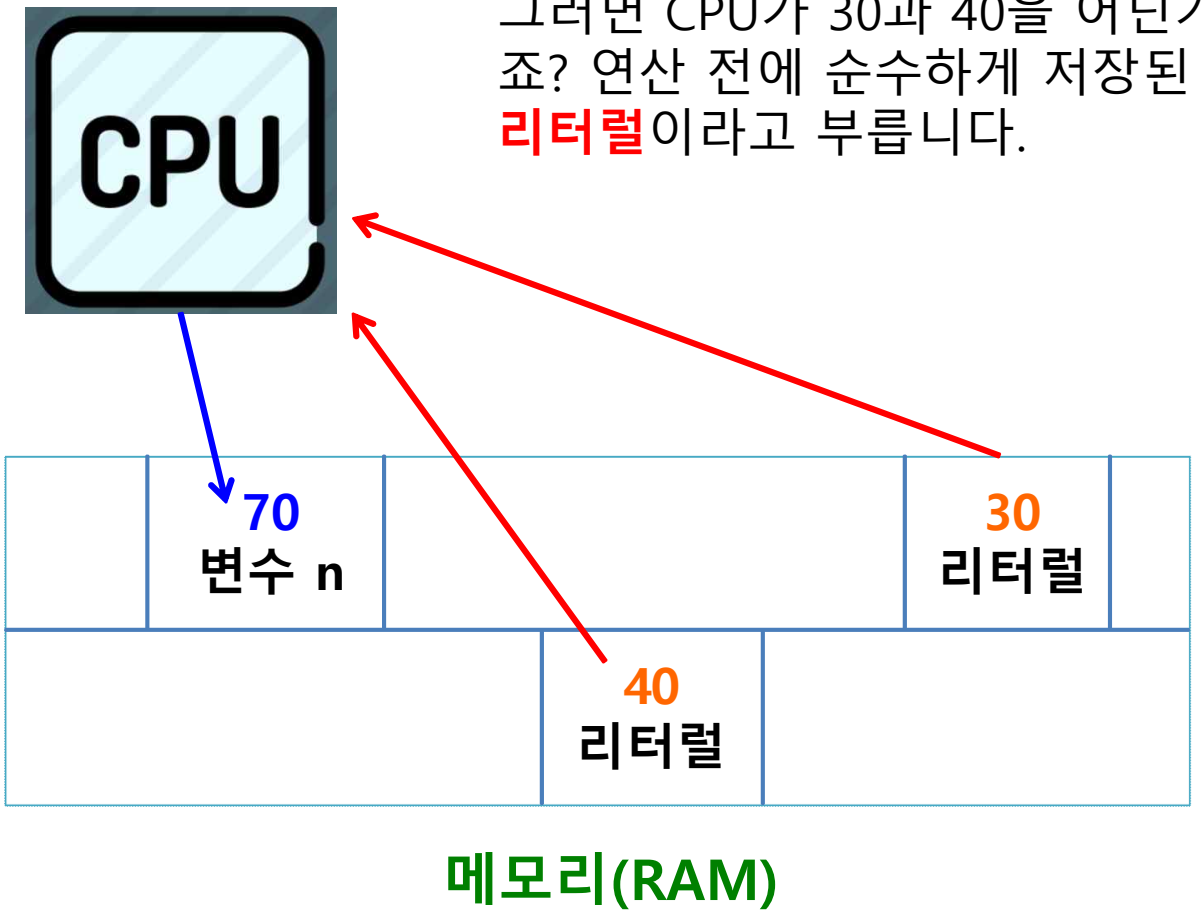


1-3-4. 검증

1-3-5. 리터럴(Literal)이란?

```
int n = 30 + 40;
```

- 왼쪽과 같은 코드에서 메모리상에 변수 n을 만드는 것보다 선행되는 연산이 무엇인가요?? 바로 30과 40을 덧셈하는 연산이 우선이죠? 그러면 CPU가 30과 40을 어딘가에서 찾아야겠죠? 연산 전에 순수하게 저장된 30과 40을 **정수 리터럴**이라고 부릅니다.



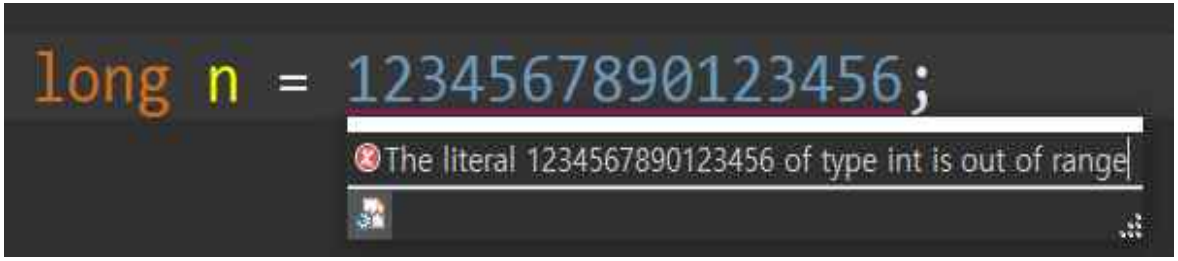
1-3-6. 리터럴의 기본 타입

리터럴도 메모리에 저장되는 데이터라고 했습니다.
그러면 리터럴도 데이터 타입을 가지고 있겠죠!

< 리터럴의 데이터 타입 >

1. 정수형: **int**
2. 실수형: **double**
3. 논리형: **boolean**
4. 문자형: **char**

1-3-7. 정수 타입에서 주의해야 할 long타입



- long타입은 8바이트나 할당받고 있는데 왜 위의 리터럴을 저장할 수 없을까요??

- 이유는 정수리터럴의 저장범위때문입니다. 정수 리터럴의 타입이 뭐라고했죠? 바로 int입니다. 약 23억까지의 정수저장범위를 가지고 있죠.

```
long n = 1234567;
```

- 따라서 int의 범위를 벗어나는 정수 리터럴은 long타입이라고 알려줘야 합니다. 방법은 리터럴의 끝에 접미사 **알파벳 L**(소문자가능)을 붙여줍니다.

```
long n = 1234567890123456L;
```

1-4-1. 실수형(floating point)

0	0	0	0	0	0	1	0

<소수점 이상>

0	0	0	0	1	0	0	0

<소수점 이하>

- 단순하게 생각해서 소수점 이상과 소수점 이하를 분리하여 표현한다고 생각해봅시다. 그러면 위의 결과는 2.8이 되겠죠?

- 그러면 위와 같이 표현했을 때 2.812313213과 같은 2.8과 2.9사이에 있는 실수를 표현할 때 개수가 제한될 것입니다.

- 따라서 정밀도를 포기한 대신 표현범위를 넓힌 다음과 같은 방식으로 실수를 표현합니다.

0	0	0	0	0	0	1	0

<e>

$$\pm(1.m)\times 2^{e-127}$$

0	0	0	0	1	0	0	0

<m>

1-4-2. 실수의 오차 검증

```java
int i;
double n = 0.0;

for(i=0; i<100; i++)
    n += 0.1;

System.out.println("0.1을 100번 더한 결과: " + n);
```

```
0.1을 100번 더한 결과: 9.999999999999998
```

이것은 자바언어의 문제가 아니다!
이론적으로 오차없는 실수를 표현할 수 있는
컴퓨팅 환경은 존재하지 않는다!

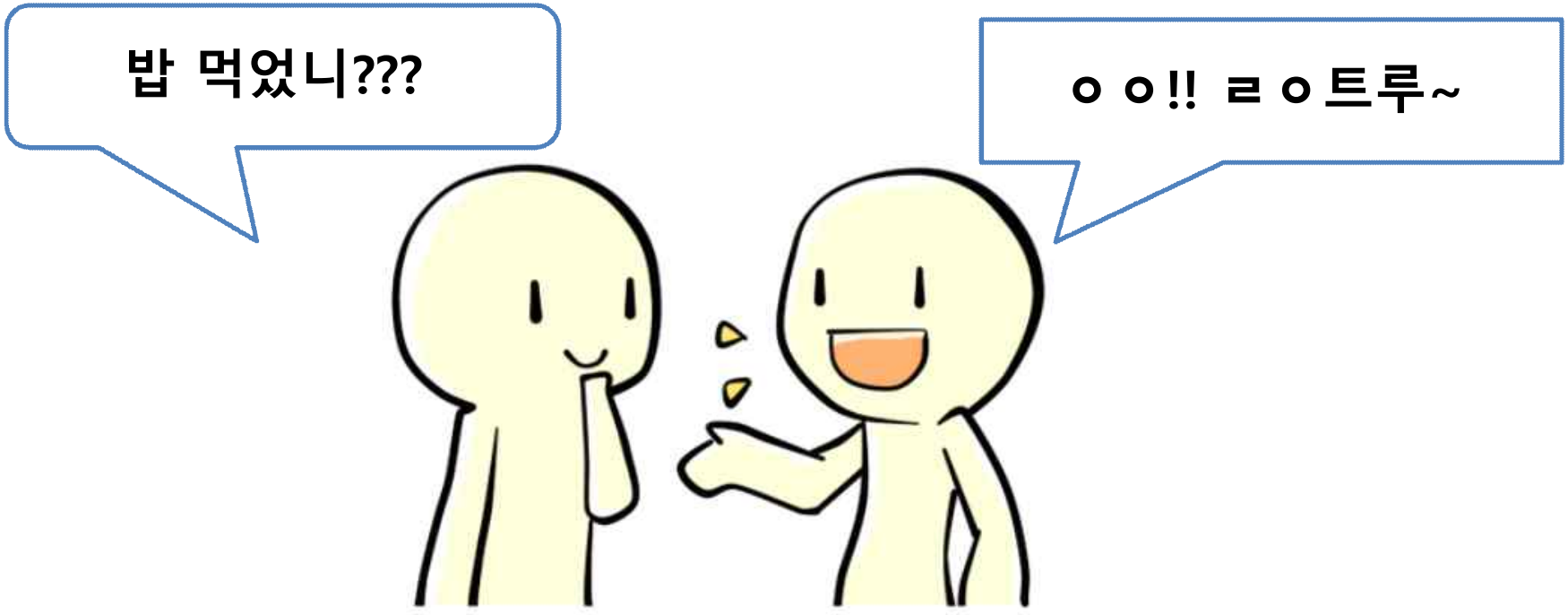
1-4-3. 실수의 저장 범위

```
float f2 = 1.234567891234F;
double d2 = 1.234567891234;
System.out.println(f2);
System.out.println(d2);
```

```
1.2345679
1.234567891234
```

float타입은 4바이트, double타입은 8바이트만큼 할당되어있기 때문에 소수점 이하자리수의 표현범위가 다릅니다. 실수 리터럴의 타입은 double이므로 float로 저장하기 위해서는 접미사 F를 붙입니다.

1-5-1. 논리형(boolean)



우리는 컴퓨터에게 **논리의 참, 거짓**을 표현하게 해야 할 때가 있습니다. 이를테면 크기의 대소비교 같은 경우가 있겠죠? 그럴 때 컴퓨터가 답변하는 논리값의 형태가 **boolean**입니다.

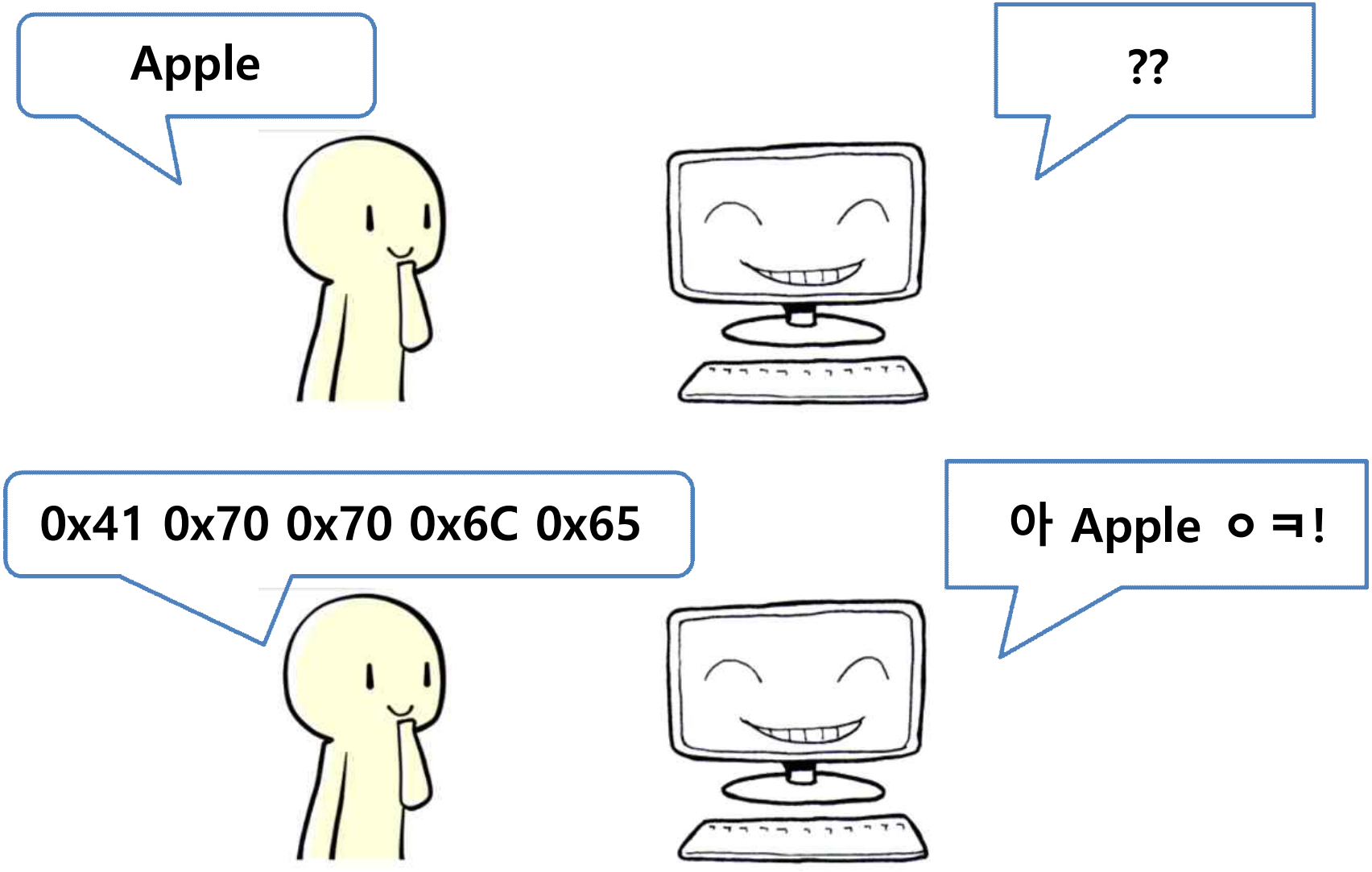
1-5-2. 논리 상수 true, false

```
boolean b1 = true;
boolean b2 = false;

boolean b3 = True; //(X)
boolean b4 = 0; //(X)
boolean b5 = "true"; //(X)
```

자바에서는 boolean타입 변수에
오직 소문자 true, false만 저장할 수 있습니다.
C언어처럼 정수 0과 1로 논리를 표현하는 것은
불가능합니다.

1-6-1. 문자형(character)



1-6-2. 아스키 코드와 유니코드

- 컴퓨터는 숫자로 모든것을 표현하기 때문에 사람의 문자 체계를 인식시키려면 상호간의 약속이 필요합니다.

- 그래서 숫자(16진수)를 문자로 매핑한 부호체계를 **아스키 코드**라고 부릅니다.

- 아스키 코드는 1바이트를 사용하여 문자를 표현합니다. 그래서 0부터 127까지 각각의 정수에 **숫자, 기호, 알파벳 대/소문자** 등을 매핑해 놓았습니다.

- 하지만 1바이트라는 용량제한 때문에 전 세계의 다양한 문자를 호환할 수 없는 문제로 인하여 2~3바이트의 공간에 문자들을 할당한 **유니코드**가 아스키코드를 대체하여 사용되고 있습니다.

1-6-3. 문자형 타입 char

```
char c1 = 'A';
char c2 = 66;
System.out.println(c1);
System.out.println(c2);

char c3 = '\uAC00';
System.out.println(c3);
```

```
A
B
가
```

- char타입 변수에는 **단일 문자**를 저장할 수 있습니다.

- **홑따옴표(')**안에 저장할 문자를 담아 대입합니다.

- 정수를 대입할 시 해당 **정수의 16진수값**에 매핑된 **유니코드**체계 문자를 대입합니다.

- 흔하게 사용하지는 않지만 탈출문자 ₩u를 사용하여 직접 16진수 유니코드값을 대입할 수도 있습니다.

1-7. 문자열형(String)

```
String s1 = "my dream ";
String s2 = "is a programmer!";

System.out.println(s1);
System.out.println(s2);
```

```
System.out.println(s1 + s2);
System.out.println(s1 + s2 + " hello~~");
```

```
my dream
is a programmer!
my dream is a programmer!
my dream is a programmer! hello~~
```

- String타입 변수에는 **문자열**을 저장할 수 있습니다.

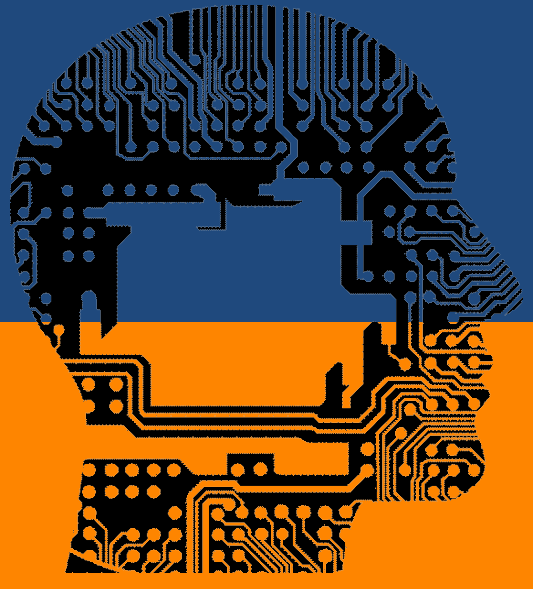
- **겹따옴표("")**안에 저장할 문자열을 담아 대입합니다.

- 문자열의 글자 수만큼 **가변적**인 데이터 크기를 가집니다.

- 문자열의 덧셈연산은 문자열을 이어붙이는 **결합연산**이 수행됩니다.

- **String**은 기본 데이터타입이 아니고 **참조 데이터타입**입니다. 자세한 내용에 대해선 객체와 클래스 파트에서 다루므로 지금은 넘어가세요! ^^

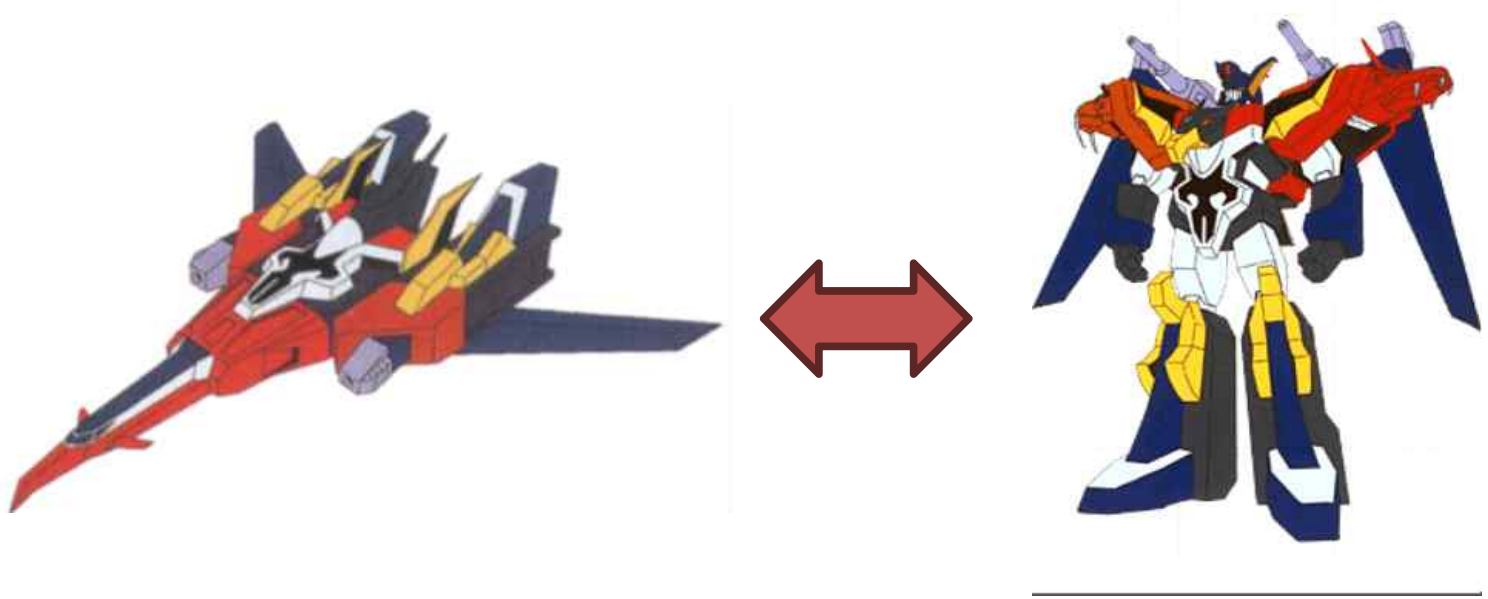
한국IT진흥부설
정보보호교육학원 아이섹



자바 프로그래밍 기초

2. 형 변환 (Type Casting)

2-1. 형 변환이란?



형 변환(type casting)이란 데이터들이
서로 **형태를 바꾸는 것**을 의미합니다.
형 변환은 묵시적 형변환과 명시적 형변환
2가지 경우로 나뉩니다.

2-2-1. 묵시적 형변환(promotion)

byte < short = char < int < long < float < double

묵시적 형변환이 일어나는 방향

작은 크기의 데이터 타입을 큰 데이터 타입으로
변환할 때는 자동으로 형 변환이 일어납니다.

2-2-2. 묵시적 형변환 예시

```java
byte b = 10;
int i = b;

char c = '가';
int j = c;
System.out.println(j);

int k = 500;
double d = k;
System.out.println(d);
```

```
44032
500.0
```

2-3-1. 명시적 형변환(type casting)

byte < short = char < int < long < float < double

명시적 형변환이 일어나는 방향

큰 크기의 데이터 타입을 작은 데이터 타입으로 변환할 때는 **형 변환 연산자**를 사용해서 직접 변환을 **명시**해야 합니다.

2-3-2. 명시적 형변환 예시

< 형 변환 연산자 >
(변환할 타입명)
ex) (int)

```
double d = 4.81234;
int i = (int)d;
System.out.println(i);

int k = 1000;
byte b = (byte)k;
System.out.println(b);
```

```
4
-24
```

2-4-1. 연산시 일어나는 형변환1

다른 데이터 타입끼리 연산을 진행할 시
크기가 작은 데이터가 큰 데이터로
자동 형변환된 후 연산이 진행됩니다.

```
char c = 'B';
int i = 2;
System.out.println(c + i);
```

```
c+i의 결과: 68
```

크기가 작은 char가 int로 변환됨!!

2-4-2. 연산시 일어나는 형변환2

int보다 작은 크기(byte, short, char)의 연산은
자동으로 결과값이 int로 변환되어 처리됩니다.

```
char c1 = 'A';
char c2 = 'B';
```

```
c1+c2: 131
```

2개의 char가 모두 int로 변환됨!!

감사합니다
THANK YOU