Basic Concepts of Object, Variable, and Method

Byeongjoon Noh

Dept. of AI/Bigdata, SCH Univ.

powernoh@sch.ac.kr

Contents

1. Basic data representation in computer

Data representation

Data

- Symbols that represent people, events, things, and ideas
- A name, a number, colors in a photograph, notes in a musical composition

Data representation

- A form in which data is stored, processed, and transmitted
- Device stores data in digital formats that can be handled by electronic circuitry

Data representation

Binary code

- 0s and 1s used to represent digital data
- A “bit” is a 0 or 1 used in the digital representation of data
- A digital files is a named collection of data that exits on a storage medium
  - Hard disk, USB, external storage

Bit and Byte

- Bit: The smallest unit of data in computer for a single binary value
- Byte: A group of binary digits (bits), usually 8 bits

Bit	One binary digit
Byte	8 bits
Kilobit	1,024 or 2^{10} bits
Kilobyte	1,024 or 2^{10} bytes
Megabit	1,048,576 or 2^{20} bits
Megabyte	1,048,576 or 2^{20} bytes
Gigabit	2^{30} bits
Gigabyte	2^{30} bytes
Terabyte	2^{40} bytes
Petabyte	2^{50} bytes
Exabyte	2^{60} bytes

Data representation

Number range using bit

- 1 bit: only 0, 1 (2 numbers)
- 2 bit: 00, 01, 10, 11 (4 numbers)
- 3 bit: 000, 001, 010, 011, 100, 101, 110, 111 (8 numbers)
- N bit: 2^N numbers

0	1

0	0	0	1	1	0	1	1

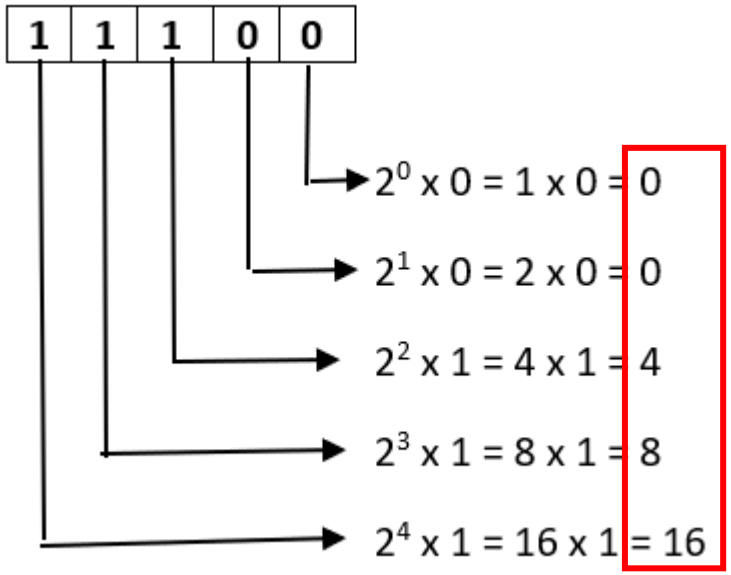
0	0	0	0	0	1	0	1	0	0	1	1
1	0	0	1	0	1	1	1	0	1	1	1

Data representation

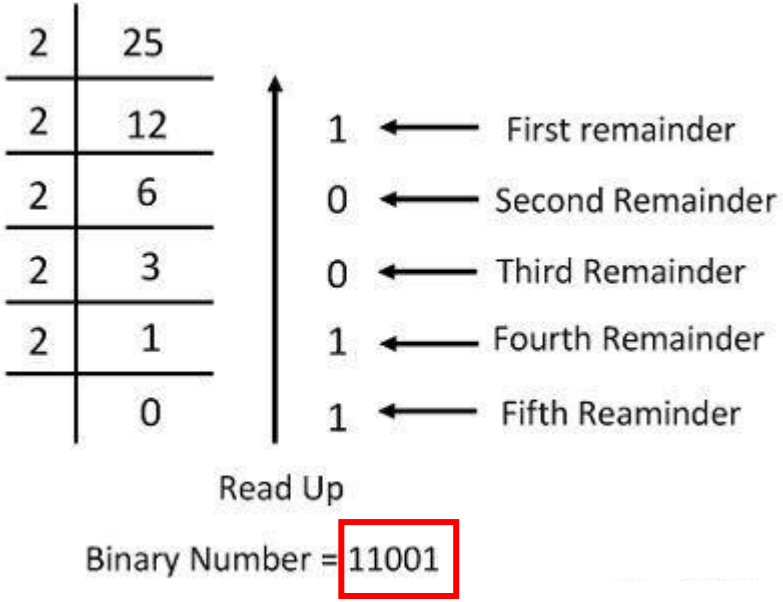
Binary and Decimal

Decimal (BASE 10)	0	1	2	3	4	5	8	10
Binary (BASE 2)	0	1	10	11	100	101	1000	1010

- Binary to Decimal



- Decimal to Binary



Data representation

How to handle other number system?

- Octal number: 0~7
- Hexadecimal number: 0~9, A~F

- Think
  - How to convert hexadecimal to decimal number
  - How to convert octal to binary number

  - Ex1) $(AB1)_{16}$ ➔ $(??)_{10}$
  - Ex2) $(1071)_8$ ➔ $(??)_2$

DECIMAL (BASE 10)	BINARY (BASE 2)	OCTAL (BASE 8)	HEXADECIMAL (BASE 16)
0	00000	0	0
1	00001	1	1
2	00010	2	2
3	00011	3	3
4	00100	4	4
5	00101	5	5
6	00110	6	6
7	00111	7	7
8	01000	10	8
9	01001	11	9
10	01010	12	A
11	01011	13	B
12	01100	14	C
13	01101	15	D
14	01110	16	E
15	01111	17	F
16	10000	20	10
Examples			
255	11111111	377	FF
256	100000000	400	100

Data representation

Hexdecimal to decimal number

- $(AB1)_{16}$

$$A * 16^2 = 10 * 256 = 2560$$
$$B * 16^1 = 11 * 16 = 176$$
$$1 * 16^0 = 1 * 1 = 1$$
$$2560 + 176 + 1 = 2737$$

Octal to binary number

- $(1071)_8$

$1 \rightarrow (001)_2$
$0 \rightarrow (000)_2$
$7 \rightarrow (111)_2$
$1 \rightarrow (001)_2$
$(001\ 000\ 111\ 001)_2$

Data representation

Text representation

- Text is commonly referred to as “character” in computer language
- Character data is composed of letter, symbos, and numeral that are not used in calculations
  - Name, address, hair color, etc.
- Employing several types of codes to represent character data
  - ASCII
    - American Standard Code for Information Interchange
    - A → (0)1000001(binary)= 65(decimal)
  - Unicode
    - 16 bits code, 65000 characters
  - UTF-8
    - A variable-length coding scheme with 7 bits

Data representation

Text representation

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Complement (보수)

What is complement?

- 보충 해주는 수 (두 수의 합이 진법의 밑수(N)가 되게 하는 수)
- Example
  - (10진수) 4의 10의 보수 = 6
  - (10진수) 100의 10의 보수 = 900 (100+900 = 1000 = 10^3)
  - (10진수) 3의 17의 보수 = 14

- Why?
  - 컴퓨터에서 음의 정수를 표현하기 위해 고안됨
  - 컴퓨터는 Adder (가산기) 만으로 사칙연산을 수행함
  - 즉, 뺄셈연산은 덧셈으로 형식을 변환하여 계산 ➔ A – B = A + (-B)

Complement (보수)

1의 보수

- 각 자릿수의 값이 모두 1인 수에서 주어진 2진수를 뺌
- 컴퓨터에서 기본적으로 8bit (1byte)를 기본값으로 하며 7bit를 수의 표현에 활용

  - 첫 bit는 sign bit로 할당
- Example
  - 0001 = +1
  - 1001 = -1
  - 000 = +0? -0

Complement (보수)

1의 보수법의 음수표현

- 011 → +3
- 010 → +2
- 001 → +1
- 000 → +0
- 111 → -0
- 110 → -1
- 101 → -2
- 100 → -3

Complement (보수)

2의 보수법을 활용한 음수 표현

- 1의 보수에 1을 더함
- Example
  - 0111 → 1000 → 1001
  - 0110 → 1001 → 1010

1의 보수법

- +3 → 011
- +2 → 010
- +1 → 001
- +0 → 000
- -0 → 111
- -1 → 110
- -2 → 101
- -3 → 100

2의 보수법

- +3 → 011
- +2 → 010
- +1 → 001
- +0 → 000
- -0 → 111 → 000
- -1 → 110 → 111
- -2 → 101 → 110
- -3 → 100 → 101
- -4 → 100

Complement (보수)

2의 보수법을 통한 정수의 표현 범위

- n bit ➔ $-2^{n-1} \sim 2^{n-1} - 1$
- 3 bit ➔ -4 ~ 3 or 0 ~ 8
- 8 bit ➔ -128 ~ 127 or 0~255
- 16 bit ➔ -32768 ~ 32767 or 0 ~ 65535

2. Concept of object

Object and OOP

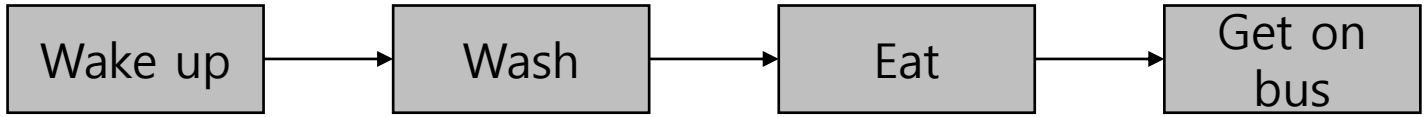
Objects

- A thing, both tangible and intangible
- Account, student, vehicle, product, delivery, order, etc.

Object-oriented programs (OOP) use <u>objects (객체)</u>

What is OOP?

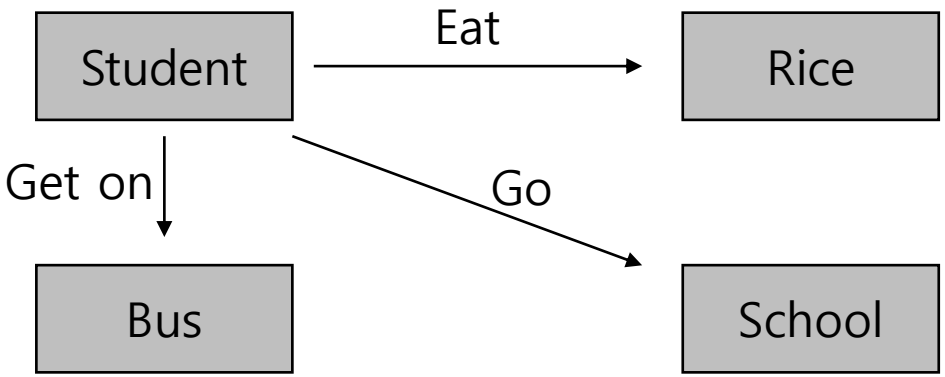
- Procedure-oriented programming (객체 지향 프로그래밍)
  - Programming with flow of time or event



Object and OOP

OOP

- Object/function-based flow in program



How to develop OOP?

- Define objects
- Design/implement functions providing to each object
- Implement cooperation between objects and functions

Object and OOP

```
public class User {
    int userId;
    String userName;
    String userGrade;
}

public class Order {
    int orderId;
    String buyerId;
    String sellerId;
    int productId;
    String orderDate;
}
```

In your life, what is object & object-based function

- Log in as a member to an online shopping mall,
- Select one of products, and
- Sold by several sellers and place an order

- Stopped by a coffee shop at Starbucks,
- Ordered ice cafe latte

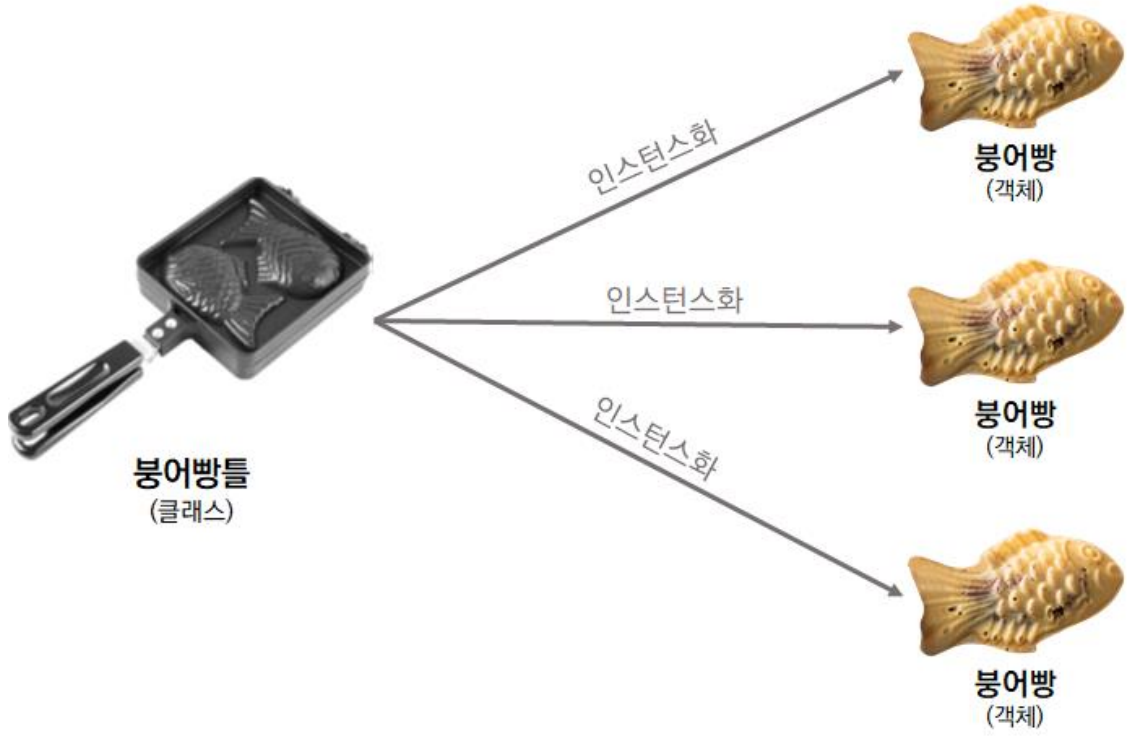
In computer program,

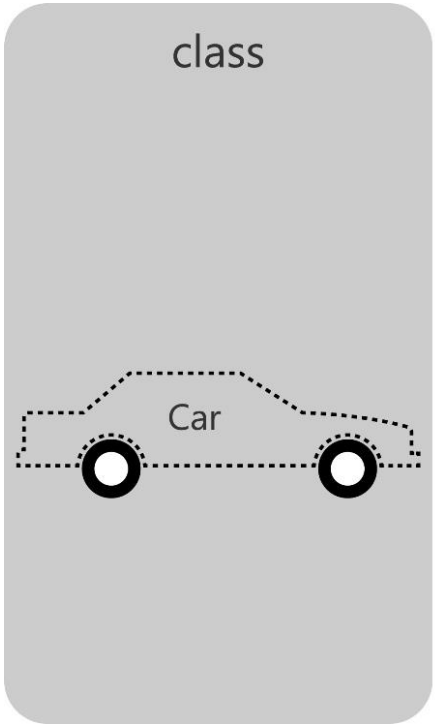
- Provide a definition for objects
  - How they behave and what kinds of information they maintain
  - Called a <u>Class</u>

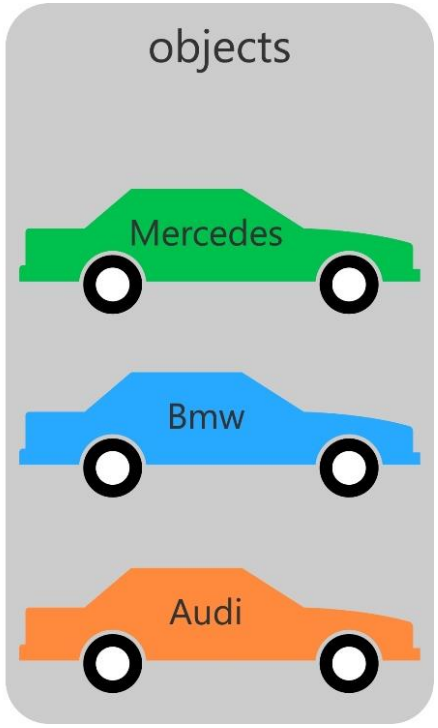
Class

Class is blueprint of Object

- An <u>object</u> is called an <u>instance</u> of a <u>class</u>







Class

Java 에서 class 구현

```java
public class Order {
    int orderId;
    String buyerId;
    String sellerId;
    int productId;
    String orderDate;
}
```

```java
public class Student {
    int studentNumber;
    String studentName;
    int majorCode;
    String majorName;
    int grade;
}
```

```java
public class UserInfo {
String userId;
String userPassWord;
String userName;
String userAddress;
int phoneNumber;
}
```

인스턴스화 (객체 만들기)

```java
…
Order order1;
Order order2;
Order order3;
```

```java
…
Student Byeongjoon;
Student Sunghoon;
Student Minji;
```

3. Variable

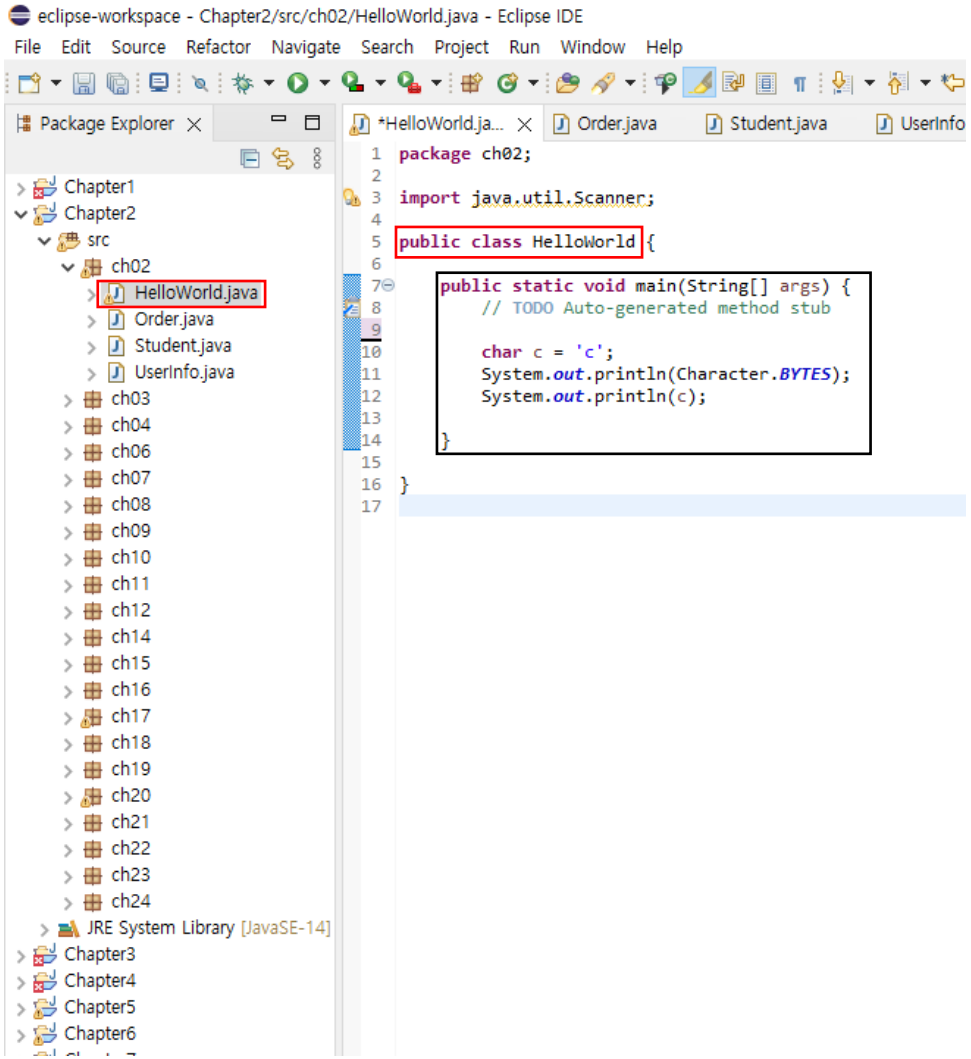
Java project structure in Eclipse

클래스

```java
/*
* 소스 파일 : Hello.java
*/
public class Hello {

   public static int sum(int n, int m) {
      return n + m;
   }

   // main() 메소드에서 실행 시작
   public static void main(String[] args) {
      int i = 20;
      int s;
      char a;

      s = sum(i, 10); // sum() 메소드 호출
      a = '?';
      System.out.println(a); // 문자 '?' 화면 출력
      System.out.println("Hello"); // "Hello" 문자열 화면 출력
      System.out.println(s); // 정수 s 값 화면 출력
   }
}
```

메소드

```
?
Hello
30
```



Java project structure in Eclipse

Java Project ➔ Package ➔ Class

Project

- 1개의 거대한 프로젝트
- Package들의 집합

Package

- Class, Interface, Enum 등 Instance(인스턴스)의 집합

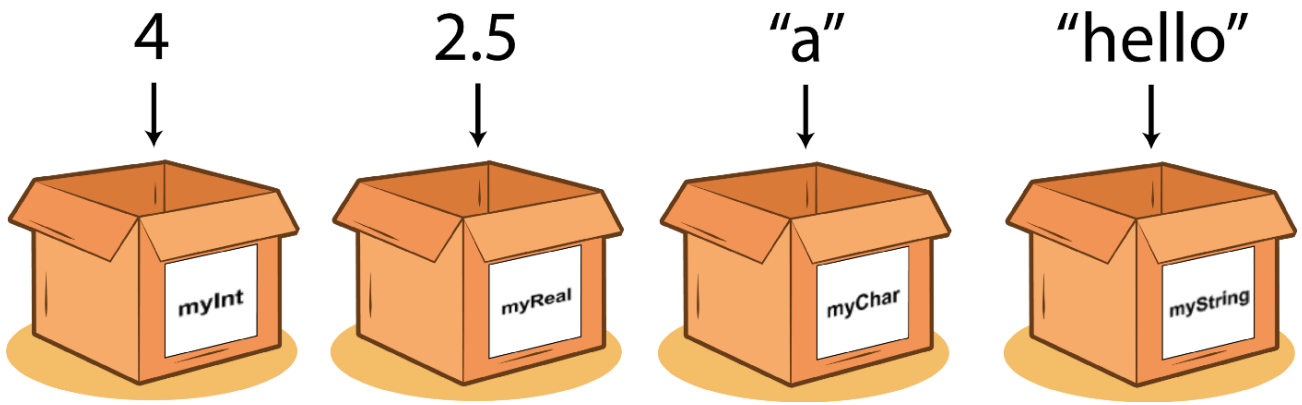
Class

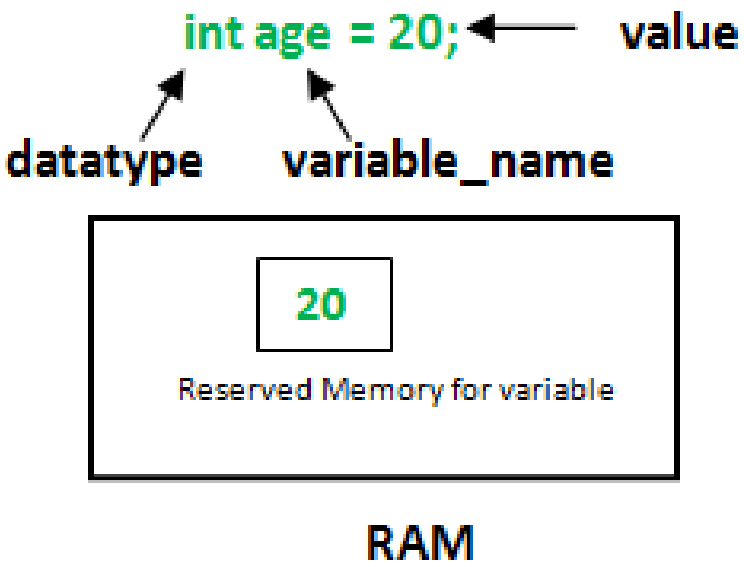
- Function(함수), Method(메소드) 등 으로 구성된 로직

Variable

Variable

- 컴퓨터 **메모리**에 **값**을 저장하도록 하는 **공간**





Variable

Variable 예제

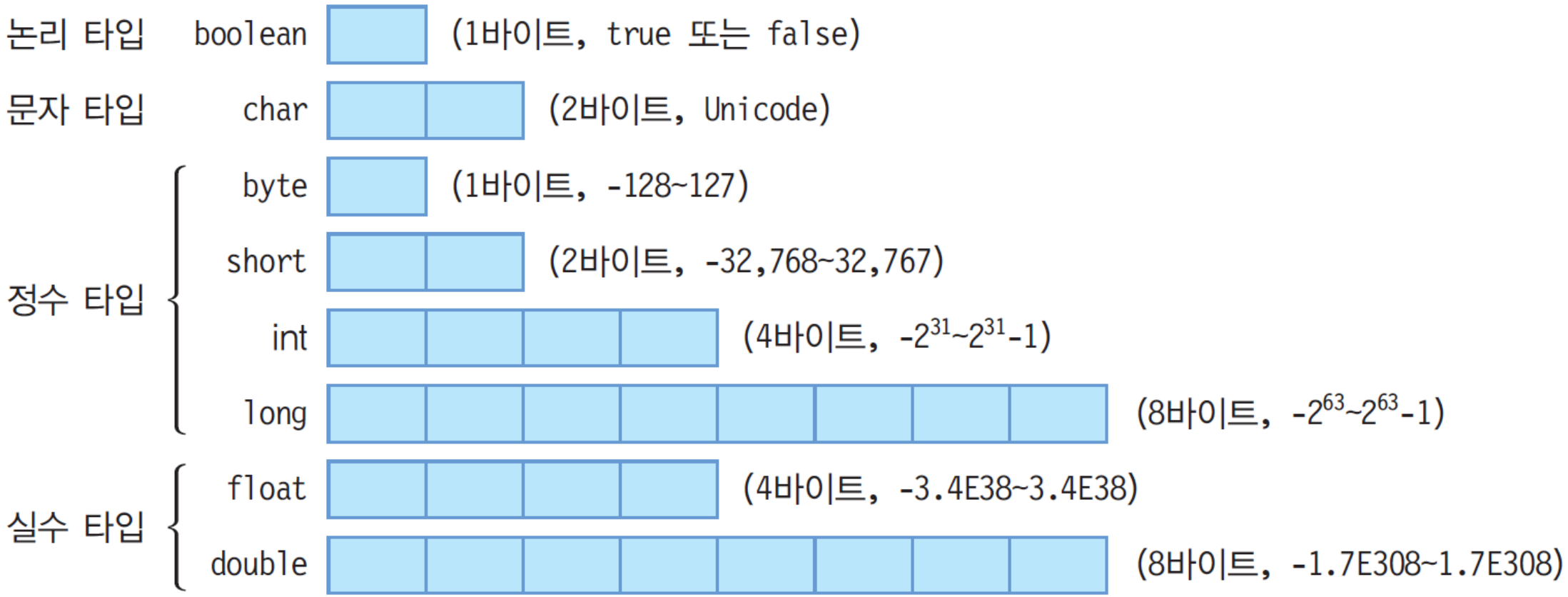
```
public class Chap2 {
    public static void main(String[] args) {
        int n1 = 10;
        int n2 = 20;
        int total = n1 + n2;
        System.out.println(total);
    }
}
```

값(결과)의 출력

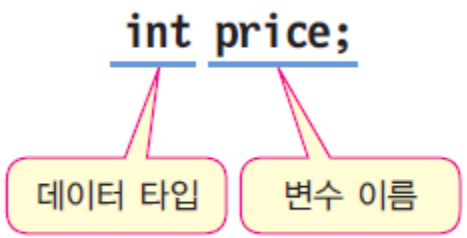
- System.*out*.println(…)

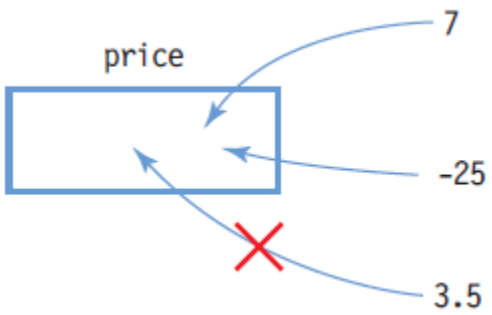
Variable

Data type



Variable





Declaration & Assignment

- Example

```
int radius;
char c1, c2, c3; // 3 개의 변수를 한 번에 선언한다.
double weight;
```

- 변수의 선언과 동시에 초기값 지정

```
int radius = 10;
char c1 = 'a', c2 = 'b', c3 = 'c';
double weight = 75.56;
```

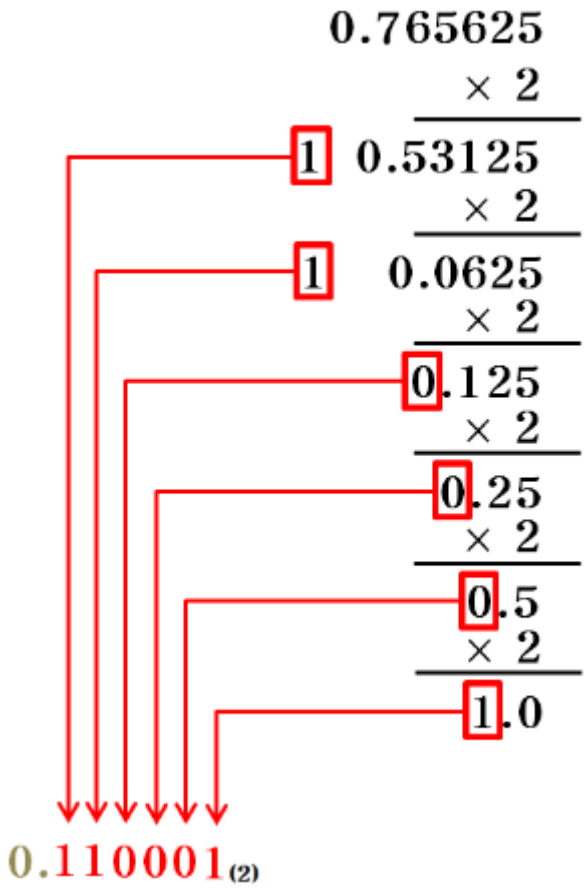
- 값 대입

```
radius = 10 * 5;
c1 = 'r';
weight = weight + 5.0;
```

Variable

부동 소수점 표현하기

- 10진수 11.765625 의 2진수 변환
- 정수부 변환
  - 11 → 1011
- 실수부 변환
  - 0.765625 → 0.110001
- ➔ 1011.110001

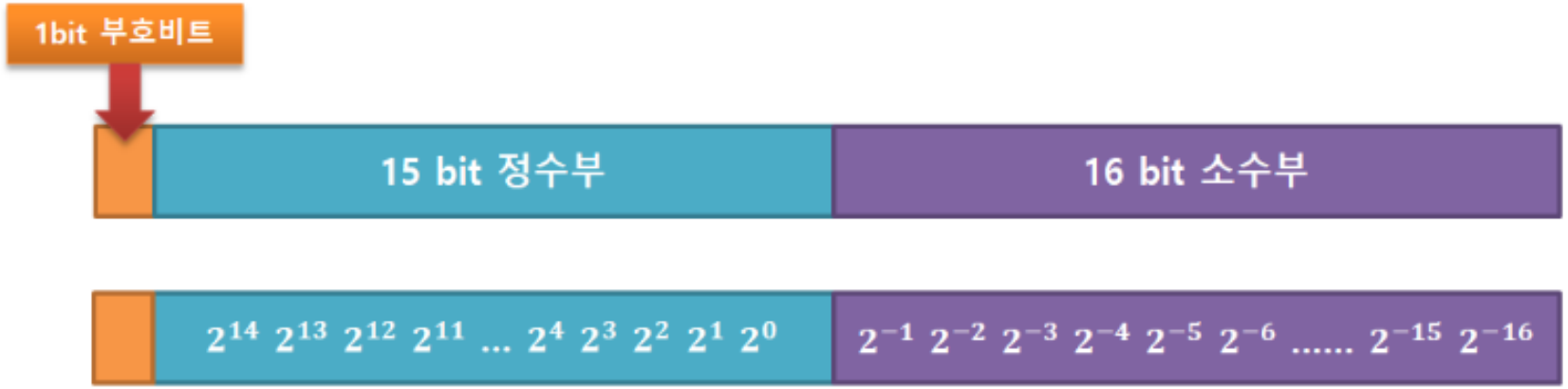


Variable

부동 소수점 표현하기

- 무한소수는?
- Ex) 0.1 ➔ 0.0001100110011…
- 정확한 변환이 불가능!

In memory assign…



- 5.625 → 

4. Method

Function (Method)

Function?

- 하나의 기능을 수행하는 일련의 코드
- 구현된(정의된) 함수는 “호출” 하여 사용
- 기능이 끝나면 제어가 “반환”됨
- 하나의 함수를 여러 곳에서 동일한 방식으로 호출하여 사용 가능

Function Definition

함수의 구성

- Name, Parameters, Return Value, and Body

함수 “정의” 하기

```
public static int addNum(int num1, int num2) {
    int result;
    result = num1 + num2;
    return result;
}
```

함수 호출

```
public static void main(String[] args) {
    int n1 = 10;
    int n2 = 20;
    int total = addNum(n1, n2);
    System.out.println(total);}
```

Identifier naming

식별자 (identifier)란?

- 클래스, 변수, 상수, 메소드 등에 붙이는 이름

식별자 작성 원칙

- 영어, 숫자, _ (under bar), $로 구성된다.
- 숫자로 시작할 수 없다.
- 대/소문자 구분한다.
- 길이의 제한은 없다.
- 예약어(키워드)는 사용할 수 없다.

Identifier naming

Naming 규칙

- “의미를 갖도록 한다”
- user_name, userName, UserName, …
- nCnt, bFlag, strInput, …

Identifier naming

변수 이름 예시

```
int    name;
char  student_ID;                    // '_' 사용 가능
void  $func() { }                    // '$' 사용 가능
class   Monster3 { }                     // 숫자 사용 가능
int    whatsyournamemynameiskitae;   // 길이 제한 없음
int    barChart;  int barchart;          // 대소문자 구분. barChart와 barchart는 다름
int    가격;                          // 한글 이름 사용 가능
```

잘못된 예

```
int        3Chapter;        // 식별자의 첫문자로 숫자 사용 불가
class      if { }           // 자바의 예약어 if 사용 불가
char       false;           // false 사용 불가
void       null() { }       // null 사용 불가
class      %calc { }       // '%'는 특수문자
```

Identifier naming

Developer's rule!

- 헝가리안 네이밍 룰

```
public class HelloWorld {}
class Vehicle {}
class AutoVendingMachine {}

int iAge;                    // iAge의 i는 int의 i를 표시
boolean bIsSingle;           // bIsSingle의 처음 b는
boolean의 b를 표시
String strName;   // strName의 str은 String의 str을 표시
public int iGetAge() {}      // iGetAge의 i는 int의 i를 표시

final static double PI = 3.141592;
```

*Java keyword

abstract	continue	for	new	switch
assert	default	if	package	synchronized
boolean	do	goto	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

End of slide