3 OMPUTER PROGRAMMING

JAVA Data Type

CONTENTS

- Analysis of Java application program
- JAVA Data Type
- Method (println / print / printf)

Analysis of application program -Comment

```
class Exam {            //
   int c;
   public int add(int a, int b)  {
         c = a + b;
         return c;
     }
}

class ExamTest {
  /*   */
   public static void main(String args[]) {
          int sum;
          int x, y;
          x = Integer.parseInt(“1”); //args[0]);
          y = Integer.parseInt(“2”); //args[1]);
                //
         Exam examobject = new Exam();
         sum = examobject.add(x,y);
        System.out.println("입력한 값의 합은 " + sum + "입니다");
   }
}
```

Analysis of application program – Class Definition

```
class Exam {
   int c;
   public int add(int a, int b)  {
         c = a + b;
         return c;
     }
}

public class ExamTest {
  /*   */
   public static void main(String args[]) {
          int sum;
          int x, y;
          x = Integer.parseInt(args[0]);
          y = Integer.parseInt(args[1]);
          //
         Exam examobject = new Exam();
         sum = examobject.add(x,y);
        System.out.println("입력한 값의 합은 " + sum + "입니다");
   }
}
```

Analysis of application program – main() method

- a special method that must exist in Java application program
- generally, Java application program creates an object of another class in main( ) method, and then gets results after sending message to the class.

```
class Exam {
    int c;
   public int add(int a, int b)  {
         c = a + b;
         return c;
     }
}
public class ExamTest {
   public static void main(String args[]) {
          int sum;
          int x, y;
          x = Integer.parseInt(“1”); //args[0]);
          y = Integer.parseInt(“2”); //args[1]);
          Exam examobject = new Exam();
          sum = examobject.add(x,y);
         System.out.println("입력한 값의 합은 " + sum + "입니다");
   }
}
```

Analysis of application program – main() method

- passes necessary information at runtime
- string type array – string object
- command-line arguments entered in the order they are stored in an array.

```
public class ExamTest {
   public static void main(String args[]) {
          int sum;
          int x, y;
          x = Integer.parseInt(args[0]);
          y = Integer.parseInt(args[1]);
          Exam examobject = new Exam();
          sum = examobject.add(x,y);
         System.out.println("입력한 값의 합은 " + sum + "입니다");
   }
}
```

Analysis of application program
– object creation and message passing

```
class Exam {
   int c;
   public int add(int a, int b)  {
        c = a + b;
        return c;
    }
}
```

```
public class ExamTest {
   public static void main(String args[]) {
        int sum;
        int x, y;
        x = Integer.parseInt(args[0]);
        y = Integer.parseInt(args[1]);
        Exam examobject = new Exam();
        sum = examobject.add(x,y);
        System.out.println("입력한 값의 합은 " + sum + "입니다");
   }
}
```

Exam 클래스로부터
객체 examobject를 생성

Exam 클래스의
add() 메소드 연산하여
반환 값은 sum 변수에 저장

Analysis of application program
– usage of standard out

- use system class, out object and println( ) method for standard out.
- println() mehtod
  - taking a string parameter, output to the screen

Analysis of application program – screen shots

Exam.java

```
1 class Exam {
2     int c;
3     public int add(int a, int b)  {
4          c = a + b;
5          return c;
6      }
7 }
```

ExamTest.java

```
 1 public class ExamTest {
 2    public static void main(String args[]) {
 3             int sum;
 4             int x, y;
 5             x = Integer.parseInt(args[0]);
 6             y = Integer.parseInt(args[1]);
 7             Exam examobject = new Exam();
 8             sum = examobject.add(x,y);
 9            System.out.println("입력한 값의 합은 " + sum + "입니다");
10    }
11 }
```

JAVA Data Type

☐ Date Type

- ■ literal (상수) : fixed value not to be changed
  - ➢ Integer literal:
    - ✓ Decimal (10진수) : 10, 15, 40, …
    - ✓ Octal (8진수) : 04, 010, 0100, …
    - ✓ Hexadecimal (16진수) : 0x5, 0xA, 0x8, …
    - ✓ Long type : 10L, 034L, 0x2AL, ….
  - ➢ real literal:
    - ✓ floating data type: 12345.5, 0.333
    - ✓ exponent type : 1.234E4, 0.91E-3
    - ✓ float : 1234.5f, 0.00234f
  - ➢ character literal: using ‘ ‘(single quatation), ‘A’, ‘b’, ‘3’, ‘*’, ‘\a’, …
    - ✓ escape sequence: ex, enter, tab etc, control character

JAVA Data Type

ASCII table

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.pubblinet.com

JAVA Data Type

- logical constant:

```
public class Main {
public static void main(String[] args) {

    System.out.println(100);          // 정수 : 소수점이 없는 수
    System.out.println(10.5);         // 실수 : 소수점이 있는 수
    System.out.println('a');          // 문자 : 단일 따옴표로 묶어줌
    System.out.println(true);         //(4) 논리값 : true, false

  }

}
```

JAVA Data Type

```
1  class IntLiteral{
2      public static void main(String args[]){
3      int intValue = 10;
4      int octValue = 010;
5      int hexValue = 0x10;
6      System.out.println("10진수 표현: " + intValue);
7      System.out.println(" 8진수 표현: " + octValue);
8      System.out.println("16진수 표현: " + hexValue);
9     }
10 }
```

8진수를 나타나기 위해 명시적으로 **0**을 덧붙임

16진수를 나타나기 위해 명시적으로 **0x**를 덧붙임

JAVA Data Type

☐ Numeric data type

```
1  public class DatatypeTest{
2   public static void main(String args[]) {
3     long l = 1004L;
4     float f = 1.004f;
5     double d2 = 1.004e2;
6     double d3 = 1.004e-2;
7
8     System.out.println("long형 데이터 i = " + l);
9     System.out.println("float형 데이터 i = " + f);
10    System.out.println("양의 지수형 데이터 i = " + d2);
11    System.out.println("음의 지수형 데이터 i = " + d3);
12   }
13 }
```

long 형을 나타내기 위해서는 'l', 'L'을 지정

float 형을 나타내기 위해서는 'f', 'F'을 지정

지수형을 나타내기 위해서는 'e', 'E'을 지정

JAVA Data Type

Declaration Variables

- format : data type variable_name;
- basic data type (java data type)
  - basic 2 data types
    - ✓ numeric type
    - ✓ boolean :
    - ✓ class
    - ✓ interface
    - ✓ array

Rule of specifying variable identifier

- available of combining english character, numeric, underline character
- no available keyword
- naming given for its role as

Variable and Data type

Variable

- name given for memory location of value stored
- Java variables must be declared before use.

Data type

- primitive type
- reference data type

Variable and Data type

타입	설명	키워드	크기	범위
character	16비트 유니코드 문자 데이터	char	16	'\u000' ~'\uFFFF'
boolean	참/거짓 값	boolean	8	true/false
byte	부호를 가진 8비트 정수	byte	8	$-2^{7} \sim 2^{7}-1$
short	부호를 가진 16비트 정수	short	16	$-2^{15} \sim 2^{15}-1$
integer	부호를 가진 32비트 정수	int	32	$-2^{31} \sim 2^{31}-1$
long	부호를 가진 64비트 정수	long	64	$-2^{63} \sim 2^{63}-1$
float	부호를 가진 32비트 부동소수점	float	32	-3.40292347E38~ 3.40292347E38
double	부호를 가진 64비트 부동소수점	double	64	-1.79769313486231570E308~ 1.79769313486231570E308

Variable and Data type: character data type

- use to represent a character
- use single quotation

```
char grade1='A';
char grade2='\u0041';
char years='2';
```

특수 문자	설명
개행 문자, linefeed	'₩n' //₩u000A
리턴, carrage return	'₩r' //₩u000D
탭, tab	'₩t' //₩u0009
백스페이스, backspace	'₩b' //₩u0008
폼피드, formfeed	'₩f' //u000C
따옴표, single quote	'₩'' //u0027
쌍따옴표, double quote	'₩"' //u0022
백슬러쉬, backslash	'₩₩' //u005C

Identifier

identifier

- Name in a class seperating variables, constants, methods, arrays, strings and user-defined classes or methods

Cases of identifier

- Class name starts with uppercase.
- Methods, variables, arrays and strings names start with lowercase.

식별자		설명
사용 가능한 식별자	strName	문자만으로 구성된 식별자는 사용가능하다.
	str_name	'_'와 '$'를 식별자 사용할 수 있다.
	strName01	첫 자가 문자임으로 숫자 01을을 사용할 수 있다.
	문자이름	한글을 식별자로 사용할 수 있다.
사용할 수 없는 식별자	01strName	첫 자는 문자로 시작해야 하기 때문에 사용할 수 없다.
	str-name	특수기호 '-'는 식별자로 사용할 수 없는 문자이다.
	char	자바에서 정한 예약어는 식별자로 사용할 수 없다.

Keword

Keyword

- No available for identifying variables, class name etc
- Reserved word for use as a special-purpose word in Java

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

Identifier and Keyword

Java 16-bit Unicode(UTF-8) used

- Unicode is the character code system designed to support for multiple languages in the world.
- Currently letters of 34,168 and a maximum of 65536 encoded letters can be encoded
- Unicode character set includes the conventional ASCII code character table.
- www.unicode.org

Variable and Data type

☐ Practice (check the error portion of line units)

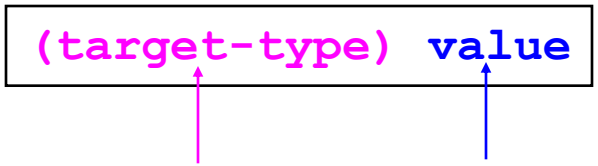
```
public static void main(String[] args) {
        a=1        //error
        a = 1;    //error-선언된 변수가 없어서
        int a;    //변수 선언하고
        a = 1;   //변수에 값을 저장
        a = 34.5;  //error
        System.out.println(a);

        1 = 2;   //error-상수는 값을 변경할 수 없다.
        a = 2;   //변수는 값을 변경할 수 있다.
                 //마지막에 대입한 값만 유지됨
        System.out.println(a);
        float m = 2.3          // error
        float n = 2.3f;
        double m = 2.3 ;   // 실수형 저장 위해 double형 변수 선언
        boolean g = true   // 논리값 저장

        int  j;
        j = 128
    }
```

Type Casting

- **Type casting occurs when assigning a value of source type (domain) into target type**
- **Widening casting**
  - Conversion taken place automatically
  - Values can be stored without loss because target type is wider than source type.
- **Narrowing casting**
  - Use explicit casting syntax conversion
  - No available if length of target type value is narrower than length of source type.

Type Casting

- Explicit syntax format




```
int a;
byte b;
…
b=(byte) a;
```

정수값→바이트 형으로

Case of Type Casting

```
public class Conversion{
  public static void main(String[] args)  {
    byte  w=10;       short x=128;    int   y=1234;   double z= 555.123;
    System.out.println("데이터 축소 형변환 결과입니다");
    w = (byte) y;
    System.out.println("정수형 값 1234를 byte로 변환결과: " + w);
    y = (int) z;         //
    System.out.println("double 값 555.123을 int로 변환결과: " + y);
    w = (byte) z;
    System.out.println("double 값 555.123을 byte로 변환결과: " + w);
    x=w;                 //
    System.out.println("byte형을 정수형으로 암시적인 형변환결과 : " + x);
    x=(short)y;          //
    System.out.println("정수형을 short형으로 명시적인 형변환결과 : " + x);
                         //
  }
}
```

Method (println / print / printf)

METHOD

- println( ) :
  - ln means an abbreviation for line.
  - line break automatically after printing contents described in a method
- print( ) :
  - Print just contents described in the method, not new line
- printf ( ) :
  - f means an abbreviation for format.
  - Method for printing out in the form of what we desire
  - Use format specifier d(decimal), c(character), etc, followed by %

For instance

Conclusion

- Analysis of Java application program
- JAVA Data Type
- Variables and Data Type
- Type Casting
- Method (println / print / printf)