

Object – Oriented Programming

LAB #2. VARIABLES AND STRINGS

Introduction

- Assistant : 신민아
- Office : IT/BT #402-2
- E-mail : minah741@naver.com

Java Language

- Java : Object Oriented Programming Language (OOP)
 - 객체가 작업을 수행한다.
 - 객체는 다른 객체의 작업의 영향을 받는다.
 - 객체의 작업을 Method라고 한다.
- Java Application : 메인 Method를 사용하는 Java

Java Language

Display 1.1 A Sample Java Program

```
1 public class FirstProgram
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello reader.");
6         System.out.println("Welcome to Java.");
7
8         System.out.println("Let's demonstrate a simple calculation.");
9         int answer;
10        answer = 2 + 2;
11        System.out.println("2 plus 2 is " + answer);
12    }
```

Annotations:

- Name of class (program) points to `FirstProgram`.
- The main method points to `main(String[] args)`.
- A red box highlights `int answer;` on line 9.

SAMPLE DIALOGUE 1

```
Hello reader.
Welcome to Java.
Let's demonstrate a simple calculation.
2 plus 2 is 4
```

1. 변수명은 숫자로 시작할 수 없다.
2. 모든 변수는 문자, 숫자, 밑줄로만 이루어져야 한다.

Keywords

- 일부 단어는 JVM Library에 의해 선언되어 있으므로 식별자로 사용할 수 없다.

ex) int, String, System, 등

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    int System = 0;  
  
    System.out.println("123");  
}  
  
public int() {  
  
}
```

Variables

- 프로그램 내에서 데이터를 저장하는 용도로 사용

ex) int x = 5;

x

5

- | | | |
|-----------|---|--------------|
| – float | } | 소수점 |
| – double | | |
| – boolean | | True / False |
| – char | | 한 글자의 문자 |
| – byte | } | 정수형 숫자 |
| – short | | |
| – int | | |
| – long | | |

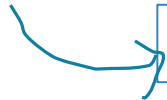
integer types

| Type | Bytes | Minimum value | Maximum value |
|-------|-------|--|--|
| byte | 1 | $-2^7 = -128$ | $2^7 - 1 = 127$ |
| short | 2 | $-2^{15} = -32,768$ | $2^{15} - 1 = 32,767$ |
| int | 4 | $-2^{31} = -2,147,483,648$ | $2^{31} - 1 = 2,147,483,647$ |
| long | 8 | $-2^{63} = -9,223,372,036,854,775,808$ | $2^{63} - 1 = 9,223,372,036,854,775,807$ |

Type Casting

```
int intVariable;  
intVariable = 42;
```

```
double doubleVariable;  
doubleVariable = intVariable;
```



doubleVariable의 값 : 42.0

다음과 같이 더 낮은 타입의 값에 모든 타입의 값을 할당할 수 있다.

byte → short → int → long → float → double

boolean type

- boolean 타입은 단 두가지 값만 갖는다.

- true
- false

- boolean의 특정 연산자

- &&
- ||
- !=
- ==

```
boolean x = true;  
boolean y = false;
```

```
System.out.println(x&&y);    // false  
System.out.println(x||y);    // true  
System.out.println(x!=y);    // true  
System.out.println(x==y);    // false
```

Constants

- 상수는 절대 바꿀 수 없는 값이다.
- 상수를 선언하는 방법은 다음과 같다.

```
final int x = 5;
```

Expressions

- 표현식은 다음과 같이 사용한다.

ex)

```
int expression = 4 + 2 * 5;
```

```
System.out.println(5 / 2.0);
```

- Java에서의 Expression 규칙
 - 각 연산자는 우선 순위가 있다.
 - * 와 / 연산자가 + 와 - 연산자보다 우선순위가 높다.
 - 부동 소수점이 사용되는 경우 결과는 부동소수점이다.

Expressions – Priority of Operators

| 우선순위 | 연산자 | 내용 |
|---|--------------|-------------------|
| <div>높음</div> <div>↑</div> <div>↓</div> <div>낮음</div> | (), [] | 괄호 |
| | !, ~, ++, -- | 부정, 증감 연산자 |
| | *, /, % | 곱셈, 나눗셈 |
| | +, - | 덧셈, 뺄셈 |
| | <, <=, >, >= | 비교 |
| | ==, != | Boolean 연산자 (비교) |
| | && | Boolean 연산자 (and) |
| | | Boolean 연산자 (or) |

The String Class

- String 클래스는 문자열을 저장하고 처리하는데 사용한다.
- 또한, String 클래스 내에는 문자열을 편하게 처리할 수 있는 여러 Method가 있다.

– String s = “Java is fun.”;

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| J | a | v | a | | i | s | | f | u | n | . |

The String Class

```
String a = "Hello";
```

```
String b = "World";
```

```
String c = a + b;
```

c

Hello World


- 두 개의 String 변수를 + 연산자를 사용하여 합칠 수 있다.

The String Class

```
String a = "Ten";
```

```
int n = 4;
```

```
String c = a + n;
```

c 

- 다음과 같이 문자열(String), 정수(int) 값을 합쳐 String 형식으로 변환이 가능하다.

The String Class

```
int a = 1;
```

```
int b = 2;
```

```
String c = a + b; ❌
```

- String 변수에 담더라도 더하는 값 중에 String 형식의 값이 없으면 에러가 발생한다.

String Class

| Method | 설명 |
|-------------|---------------------------------------|
| substring | 한 문자열에서 내용의 일부를 반환 |
| split | 문자열을 매개변수로 지정된 분리자로 나누어 문자열 배열 형태로 반환 |
| contains | 지정된 문자열이 포함되었는지 검사 |
| endsWith | 지정된 문자열로 끝나는지 검사 |
| equals | 지정된 문자열과 같은 지 검사 |
| replace | 문자열 중에 A를 B로 변경 |
| toLowerCase | 모든 문자열을 소문자로 변환 |
| toUpperCase | 모든 문자열을 대문자로 변환 |
| trim | 문자열의 양 끝의 공백을 제거 |
| valueOf | 지정된 값을 문자열로 변환 |
| length | 문자열 길이를 반환 |
| charAt | 해당 Index의 문자를 반환 |

substring

String substring(int begin)

String substring(int begin, int end)

한 문자열에서 일부만 추출하는 메소드

```
String s = "Java Programming.txt"
```

```
String s1 = s.substring(0,4);
```

```
String s2 = s.substring(10);
```

```
String s3 = s.substring( s.length() - 4 );
```

결과

s1 = "Java"

s2 = "amming.txt"

s3 = ".txt"

split

`String[] split(String regex)`

문자열을 지정된 분리자로 나누어 문자열 배열 형태로 반환한다.

```
String colors = "black,white,red,blue,yellow";
```

```
String[] color_arr = colors.split(",");
```

```
color_arr.length;
```

결과

```
arr[0] = "black"
```

```
arr[1] = "white"
```

```
arr[2] = "red"
```

```
arr[3] = "blue"
```

```
arr[4] = "yellow"
```

```
color_arr.length = 5
```

contains

Boolean contains(String s)

지정된 문자열이 포함되었는지 검사한다.

```
String s = "abcdefg";
```

```
Boolean b = s.contains("ef");
```

결과

b = true

endsWith

Boolean endsWith(String suffix)

문자열의 끝에 해당 문자열이 있는지 검사한다

↔startsWith(String prefix)

```
String file = "Hello.cpp";
```

```
Boolean b = file.endsWith("cpp");
```

결과

b = true

equals

Boolean equals(String s)

지정된 문자열과 같은지 검사한다.
대소문자를 구분한다.

```
String s = "Hello World";  
Boolean b = s.equals("Hello World");  
Boolean b2 = s.equals("hello world");
```

결과

b = true
b2 = false

※ 대소문자 구분 하지 않고 검사하는 메소드 : equalsIgnoreCase(String s)

compareTo

Boolean compareTo(String s)

지정된 문자열과 같은지 각 문자의 유니코드값에 근거해 검사한다.
반환 값은 int형이고, 대소문자를 구분한다.

```
String s = "Hello World";
```

```
int i = s.compareTo("Hello World");
```

```
int j = s.compareTo("hello world");
```

결과

i = 0

j = -32

※ 대소문자 구분 하지 않고 검사하는 메소드 : compareToIgnoreCase(String s)

replace

`String replace(String a, String b)`

문자열에 있는 a 문자열을 b로 변경한다.

`String s = "Minah Shin"`

`String n = s.replace("Shin", "Kim");`

결과

`n = "Minah Kim"`

toLowerCase

String toLowerCase()

모든 문자열을 소문자로 변환하여 반환한다.

```
String s = "Hello";
```

```
String n = s.toLowerCase();
```

결과

n = "hello"

toUpperCase

String toUpperCase()

모든 문자열을 대문자로 변환하여 반환한다.

```
String s = "Hello";
```

```
String n = s.toUpperCase();
```

결과

n = "HELLO"

trim

String trim()

문자열 양 끝의 공백을 제거한다.

```
String s = " Hello ";
```

```
String n = s.trim();
```

결과

n = "Hello"

valueOf

static String valueOf()

특정 값을 문자열로 변환하여 반환한다.

Boolean
char
int
long
float
double

String a = String.valueOf(true);

String b = String.valueOf(100);

String c = String.valueOf('c');

String d = String.valueOf(10.0);

결과

a = "true"
b = "100"
c = "c"
d = "10.0"

length

```
int length()
```

문자열의 길이를 반환한다.

```
String s = "Hello";
```

```
int n = s.length();
```

결과

n = 5

charAt

`char charAt(int index)`

해당 index의 문자를 반환한다.

`String s = "abcde";`

`char c = s.charAt(3);`

결과

`c = d`

실습

위의 메소드들을 활용하여 실습을 진행할 것

- 이름은 각자의 이름을 입력할 것(Scanner 클래스 사용)

공백

입력 : shin min ah, homework.ppt

출력 : Name Length(Korean) : 3
M.A.Shin submitted Homework.pdf

String class method 활용.

실습 – Scanner 사용법

1. import java.util.Scanner; 를 첫 줄에 입력
2. Scanner 객체를 main 메소드에 생성
3. 입력 값은 1줄로 입력하므로 nextLine() 메소드를 사용

```
1  import java.util.Scanner;
2
3  class Lab02
4  {
5      public static void main (String[] args)
6      {
7          Scanner scan = new Scanner(System.in);
8          String input = scan.nextLine();
9
10         System.out.println(input);
11     }
12 }
```

input Output

shin min ah, homework.ppt

input Output

Success #stdin #stdout 0.12s 35332KB

shin min ah, homework.ppt