

# Day15 (H1)

Byte primitive type  
Exception Handling  
File IO

20150828

Copyright (c) 2015 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

```

class MyStudent extends Student {
    int    StID;
    String Name;

    MyStudent() { super(); }
    MyStudent(String s, int i, int x, int y , int z)
    { super(x, y, z); setName(s); setStID(i); }

    ( int    getStID() { return StID; }
      String getName() { return Name; } ) 점과 함수

    ( void setStID( int    x ) { StID= x; }
      void setName( String x ) { Name= x; } ) 점과 함수

    void disp() {
        System.out.println( "-----");
        System.out.println( "Name= " + Name    );
        System.out.println( "StID= " + StID    );

        System.out.println( "Kor= " + Kor      );
        System.out.println( "Eng= " + Eng      );
        System.out.println( "Math=" + Math     );
        System.out.println( "GPA= " + Avg()    );
    }
}

```

accessor  
methods

mutator  
methods

점과 함수

점과 함수

```

public class Student {
    int Kor;
    int Eng;
    int Math;

    Student() { setKor(0); setEng(0); setMath(0); }
    Student(int x, int y, int z) { setKor(x); setEng(y); setMath(z); }

    int getKor () { return Kor; }
    int getEng () { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) { }
    void setEng ( int x ) { }
    void setMath( int x ) { }

    double Avg() { return (Kor+Eng+Math) / 3.0; }

    void disp() {
        System.out.println( "-----" );
        System.out.println( "Kor= " + Kor );
        System.out.println( "Eng= " + Eng );
        System.out.println( "Math=" + Math );
        System.out.println( "GPA= " + Avg() );
    }
}

```

클래스의 클래스

Super class의  
disp()

```

static void avg_mode( Student[] X, int mode) {

```

```

class MyStudent extends Student {
    int StID;
    String Name;

    MyStudent() { super(); }
    MyStudent(String s, int i, int x, int y, int z)
    { super(x, y, z); setName(s); setStID(i); }

    int getStID() { return StID; }
    String getName() { return Name; }

    void setStID( int x ) { StID= x; }
    void setName( String x ) { Name= x; }

    void disp() {
        System.out.println( "-----" );
        System.out.println( "Name= " + Name );
        System.out.println( "StID= " + StID );

        System.out.println( "Kor= " + Kor );
        System.out.println( "Eng= " + Eng );
        System.out.println( "Math= " + Math );
        System.out.println( "GPA= " + Avg() );
    }
}

```

클래스의 클래스

Sub class의  
disp()

```

public static void main(String[] args) throws IOException {
    // TODO Auto-generated method stub

    FileInputStream in = new FileInputStream("st.dat");
    Scanner sn = new Scanner ( in );

    MyStudent[] S = new MyStudent[5]; // S[i] : reference var

    int i; String name; int id, x, y, z;

    for (i=0; i<S.length; ++i) {
        name = sn.next();
        id = sn.nextInt();
        x = sn.nextInt();
        y = sn.nextInt();
        z = sn.nextInt();

        S[i] = new MyStudent( name, id, x, y, z );
    }

```

```

S[0].disp();
S[1].disp();
S[2].disp();
S[3].disp();
S[4].disp();

```

```

Student.avg_mode( S, 0);
Student.avg_mode( S, 1);
Student.avg_mode( S, 2);

```

```

young@young-Samsung-NB-System:~/workspace/Day15$
young@young-Samsung-NB-System:~/workspace/Day15$
young@young-Samsung-NB-System:~/workspace/Day15$ ls
bin src st.dat
young@young-Samsung-NB-System:~/workspace/Day15$ more st.dat
"Park" 20150001 99 45 50
"Kim" 20150002 88 55 80
"Lee" 20150003 77 65 90
"Baker" 20150004 66 75 80
"John" 20150005 55 85 90

```

# format ( )

```
void disp() {  
    System.out.format( "-----\n");  
    System.out.format( "Name= \t %8s\n", Name );  
    System.out.format( "StID= \t %8d\n", StID );  
  
    System.out.format( "Kor= \t %8d\n", Kor );  
    System.out.format( "Eng= \t %8d\n", Eng );  
    System.out.format( "Math= \t %8d\n", Math );  
    System.out.format( "GPA= \t %8.2f \n", Avg() );  
}
```

%s String

%d 정수형

%f 소수점

%8s ← 8칸 →

%8d ← 8칸 →

%8f ← 8칸 →

\t tab (default 3 8칸)

\n newline

%8.2f 소수점 이하 2자리

8칸

```
-----  
Name=      "Park"  
StID=     20150001  
Kor=       99  
Eng=       45  
Math=      50  
GPA=       64.67  
-----  
23칸
```

```

public static void readRecord( MyStudent[] S ) throws IOException {
    FileInputStream in = new FileInputStream("st.dat");
    Scanner sn = new Scanner ( in );

    int i;

    for (i=0; i<S.length; ++i) {

        S[i] = new MyStudent(
            sn.next(), // name
            sn.nextInt(), // id
            sn.nextInt(), // Kor
            sn.nextInt(), // Eng
            sn.nextInt() // Math
        );

    }

}

```

↑  
must  
be specified  
here

```

public static void main(String[] args) throws IOException {
    // TODO Auto-generated method stub

    MyStudent[] S = new MyStudent[5]; // S[i] : reference var
                                        is handled here

    try {
        readRecord( S ); → Can throw exceptions
    } catch (IOException e) { ✓ catch here
        System.out.println( "st.dat cannot be found..." );
    }

    S[0].disp();
    S[1].disp();
    S[2].disp();
    S[3].disp();
    S[4].disp();

    Student.avg_mode( S, 0 );
    Student.avg_mode( S, 1 );
    Student.avg_mode( S, 2 );

}

```

# Print Stream

```
import java.io.*;

class MySys {
    static PrintStream out = new PrintStream( System.out );
}

public class PrintTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        MySys.out.println("Hello");
        MySys.out.println( 123 );
    }
}
```

class static  
out field

System.out

↑ static field

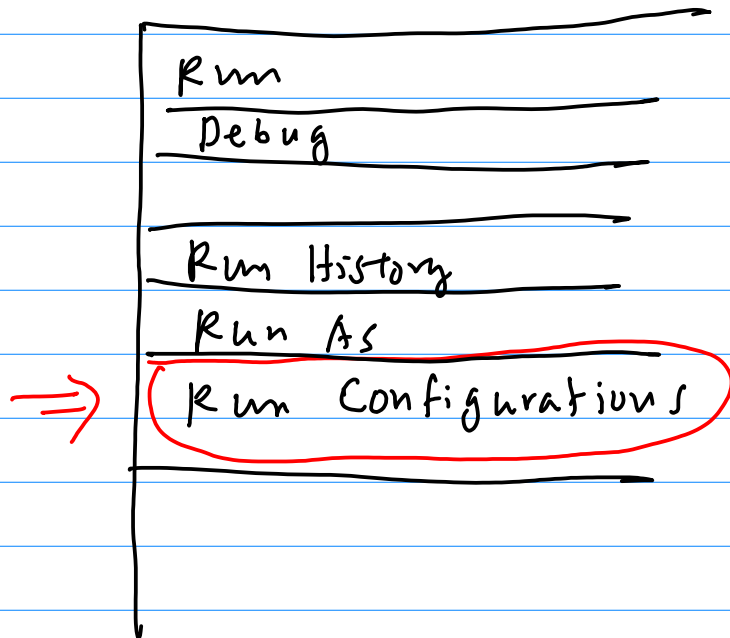
of PrintStream class type

# Argument Test

```
public class ArgumentTest {  
  
    /**  
     * @param args  
     */  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int i;  
        for (i=0; i<args.length; ++i) {  
            System.out.println( "args[" );  
            System.out.println(    i    );  
            System.out.println( "]= " );  
            System.out.println( args[i] );  
        }  
    }  
}
```

Menu Bar of

Run -





*Select Arguments tab*

The image shows the Eclipse Run Configurations dialog box. The title bar reads "Run Configurations" and the subtitle is "Create, manage, and run configurations". Below this, it says "Run a Java application" with a green play button icon. The dialog is divided into two main sections. On the left is a tree view of configurations, with "ArgumentTest" selected under "Java Application". On the right is the configuration details for "ArgumentTest". At the top of this section, the "Name" is "ArgumentTest". Below that, there are tabs for "Main", "Arguments", "JRE", "Classpath", "Source", and "Environment". The "Arguments" tab is selected and circled in red. A red arrow points from the handwritten text "Select Arguments tab" to this tab. The "Project" field contains "Day15" and has a "Browse..." button. The "Main class" field contains "ArgumentTest" and has a "Search..." button. There are three checkboxes: "Include system libraries when searching for a main class", "Include inherited mains when searching for a main class", and "Stop in main", all of which are currently unchecked. At the bottom of the dialog, there are buttons for "Apply", "Revert", "Close", and "Run".

Run Configurations

Create, manage, and run configurations

Run a Java application

Name: ArgumentTest

Main Arguments JRE Classpath Source Environment

Project: Day15 Browse...

Main class: ArgumentTest Search...

Include system libraries when searching for a main class

Include inherited mains when searching for a main class

Stop in main

Apply Revert

Close Run

**Run Configurations**  
 Create, manage, and run configurations  
 Run a Java application

Name: **ArgumentTest**

Main **Arguments** JRE Classpath Source Environment

Program arguments:  
string1 string2 string3 string4  
 ↓ ↓ ↓ ↘  
 args[0] args[1] args[2] args[3]

VM arguments:

Working directory:  
 Default: `${workspace_loc:Day15}`  
 Other:

Buttons: Apply, Revert, Close, Run

```
<terminated> ArgumentTest [Java Application] /usr/lib/jvm/java-7-openjdk
args[0]= string1
args[1]= string2
args[2]= string3
args[3]= string4
```

\* main 인덱스 0

0, 1, 2, 3

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub
```

```
    String name = args[0];  
    int id = Integer.parseInt(args[1]);  
    int kor = Integer.parseInt(args[2]);  
    int eng = Integer.parseInt(args[3]);  
    int math = Integer.parseInt(args[4]);
```

Wrapper class

int 3 byte

```
    System.out.println( "name= " + name );  
    System.out.println( "id = " + id );  
    System.out.println( "kor = " + kor );  
    System.out.println( "eng = " + eng );  
    System.out.println( "math= " + math );
```

```
}
```

# Wrapper class

## Primitive Wrapper Class Constructor Argument

<u>b</u> oolean	<b>B</b> oolean	boolean or String
<u>b</u> yte	<b>B</b> yte	byte or String
<u>c</u> har	<b>C</b> haracter	char
<u>i</u> nt	<b>I</b> nteger	int or String
<u>f</u> loat	<b>F</b> loat	float, double or String
<u>d</u> ouble	<b>D</b> ouble	double or String
<u>l</u> ong	<b>L</b> ong	long or String
<u>s</u> hort	<b>S</b> hort	short or String

Method	Purpose
parseInt(s)	returns a signed decimal integer value equivalent to string s
toString(i)	returns a new String object representing the integer i
byteValue()	returns the value of this Integer as a byte



