

# JAVA 입출력 정리

by. 이창주

## 입력

### System.in

- InputStream
- ```
InputStream in = System.in; // System.in.read();
```
- 자바의 **내장 클래스** (java.lang) import 하지 않아도 사용 가능
  - ≠ String도 클래스이지만 java.lang에 속해있어 import하지 않아도 사용가능
- InputStream의 read 메서드는 **1byte**의 사용자의 입력을 받아들인다.
- **아스키 코드값을 int**로 해서 저장
- **엔터키**를 입력해야 사용자의 입력이 종료
- **IOException** 필요
  - 입력값을 읽어들이는 때 발생할 수 있다.
  - main method의 **throws**를 통해 예외처리를 뒤로 **미루기** 가능

```
import java.io.IOException;
import java.io.InputStream;
// 따로 객체를 만들지 않으면 생략 가능

public class System_in {

    public static void main(String[] args) throws IOException {

        int Number = System.in.read() - 48;
        int Number2 = System.in.read(); // 아스키 코드값 출력

        char Char = (char) System.in.read();
        // (char) 안붙이면 오류 int or Byte

        System.out.println("Number : " + Number);
        System.out.println("Number2 : " + Number2);
        System.out.println("Char : " + Char);

        // ex ) input 5a output Number : 5 Char: a
        // ex ) input 5 a output Number : 5 Char :

        byte[] Byte = new byte[4096];

        System.in.read(Byte, 0 , 4096);

        String StrByte = new String(Byte);
```

```

        System.out.print("StrByte : " + StrByte);

/*      99AABCDEFGHGIJKLMNOPQRSTUVWXYZ99
      Number : 9
      Char : A
      StrByte : AABCDEFGHGIJKLMNOPQRSTUVWXYZ*/

    InputStream in = System.in;

    int Number3 = System.in.read() - 48;
    int Number4 = System.in.read(); // 아스키 코드값 출력

    System.out.println("Number3 : " + Number3);
    System.out.println("Number4 : " + Number4);

}
}

```

```

Input : 99AABCDEFGHGIJKLMNOPQRSTUVWXYZ99
Output :Number : 9
        Char : A
        StrByte : AABCDEFGHGIJKLMNOPQRSTUVWXYZ

```

## InputStreamReader

- 읽어들이 값을 항상 아스키코드 값으로 해석해야하는 방식 불편
- 입력한 문자값을 그대로 출력
  - -> 바이트 대신 **문자**로 입력 스트림을 입력 **char**

- ```

import java.io.IOException;
import java.io.InputStreamReader; // Import 필요

public class InputSreamReader {
    public static void main(String[] args) throws IOException{

        InputStreamReader reader = new InputStreamReader(System.in);
        char[] a = new char[3];

        reader.read(a);
        // or reader.read(a,0,3);
        // char 타입으로 읽어드림 char타입 배열 필요
        // Byte,int -> char

        byte[] Byte = new byte[3];

        // 오류발생
        // reader.read(Byte);
        System.out.println(a);

    }
}

```

```
Input : abc
Output : abc
```

-----

```
Input : abcd
Output : abc -> 고정된 길이로만 스트림을 읽음 BufferedReader 필요성
```

## BufferedReader

- 고정된 길이로만 스트림을 읽어야 한다는 점이 불편
- 사용자가 엔터키를 입력할 때 까지 사용자의 입력을 전부 받아들이지 수는 없을까?
- 문자열(String)로 입력받음
- BufferedReader 사용법

- ```
// BufferedReader 사용

// 1.Import
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.BufferedReader;
// 2. 메인메소드에 throws IOException 예외처리 추가
public static void main(String[] args) throws IOException {
}

// 3. br객체생성
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

// 4. br.readLine() 등 메소드 이용하여 String으로 입력받기
```

- ```
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.BufferedReader;

public class BufferedReader_Practice {

    public static void main(String[] args) throws IOException {
        InputStream in = System.in;
        InputStreamReader reader = new InputStreamReader(in);
        BufferedReader br = new BufferedReader(reader);

        /*
        BufferedReader 클래스
        public BufferedReader(Reader in) {
            this(in, defaultCharBufferSize);
        }*/
        String a = br.readLine();
        System.out.println(a);

        int b = Integer.parseInt(br.readLine());
        System.out.println(b);
```

```

        // 숫자 사이에 공백이 있다면 java.lang.NumberFormatException:
        // -> 공백 처리 필요

    }
}

```

```

input : abcde
output : abcde

input : abcdef
Output : abcdef

```

BufferedReader 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html>

## 정리

1. **InputStream** = **byte**으로 입력
2. **InputStreamReader** = **character**으로 입력
3. **BufferedReader** = **String**으로 입력

## Scanner

- J2SE 5.0 부터 Scanner 라는 `java.util.Scanner` 클래스가 새로 추가

```

import java.util.Scanner; // import 필요

public class Scanner_practice {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        /* public Scanner(InputStream source) {
            this(new InputStreamReader(source), WHITESPACE_PATTERN);
        }*/

        int n = sc.nextInt();
        int m = sc.nextInt(); // enter & 공백 넘어 입력받음

        System.out.println(n+m);

        String a = sc.nextLine(); // 한줄

        System.out.println(a);

        String b = sc.next(); // 단어 입력 종료됨
        System.out.println(b);

    }
}

```

```

input :
3 4 abc abc
abc

output :

7
abc abc <- 주의! nextInt후에 공백이 남아있음 공백이 남아있는 채로 읽음
abc

```

- 다른 메서드 알아보기

Scanner 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html>

## Scanner / Buffered 차이

- `Scanner` 느린 이유는, 입력을 읽는 과정에서 정규 표현식을 적용하고, 입력값 분할, 파싱 과정을 스스로 제공
- 실제로, `Scanner` 에서 제공하는 `.nextInt()`, `.nextDouble()` 메서드는 잘못 입력하면 입력 단계에서부터 예외가 발생.
- `BufferedReader` 는 모든 입력을 Char형으로, 버퍼를 사용하여 받는다.
- 하나의 글자에 대해 전달이 이루어지는 것이 아닌, 전체 입력(혹은 버퍼 단위)에 대해서만 전달되기 때문에 **속도 부분에서 매우 유리할 수 밖에 없다.**
- 그래서 속도는 빠르지만, 사용자가 사용하기 편하게 조작하려면 **별도의 메서드를 호출**해야한다.

## String을 나누어서 변수 입력 받기

### String.substring / StringBuilder

```

// 백준 15552 빠른 A+B

import java.io.*;

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

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(System.out));
        StringBuilder sb = new StringBuilder();
        int n = Integer.parseInt(br.readLine());

        for (int i = 0; i < n; i++) {
            String line = br.readLine();
            int split = line.indexOf(" ");

```

```

        sb.append(Integer.parseInt(line.substring(0,
split))+Integer.parseInt(line.substring(split+1))).append('\n');
    }
    bw.write(String.valueOf(sb));
    bw.flush();
    bw.close();
}
}

```

`StringBuilder` 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/lang/StringBuilder.html>

## String.split

사용법

`String`의 기본 메서드를 이용하여 나눔  
`String[] tokens = String.split("나누는 기준으로 할 값");` 토큰화 되어 배열에 저장

```

public static void main(String[] ar){
    String str="hello string split";
    String[] tokens=str.split(" ");

    for(int i=0;i<tokens.length;i++){
        System.out.println(tokens[i]);
    }
}

```

input :  
 output :  
 hello  
 string  
 split

`String` 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/lang/String.html>

## StringTokenizer

사용법

```

import java.util.StringTokenizer; // import 필수

StringTokenizer st = new StringTokenizer(String, ["나누는 기준으로 할값"] 생략시
default" "), [true or false]나누는 기준을 토큰에 추가할지 여부 생략시 default false )

st.nextToken() - > 나누는 기준 이후 토큰화
st.countTokens() -> 남아있는 토큰 숫자
st.hasMoreTokens() -> 토큰이 남아있는지 여부 리턴

```

```

import java.util.StringTokenizer; // import 필수

```

```

public static void main(String[] ar){
    String str="this==string-includes=delims";
    StringTokenizer stk=new StringTokenizer(str,"==");
    System.out.println(str);
    System.out.println();

    System.out.println("total tokens:"+stk.countTokens());
    System.out.println("=====tokens=====");
    while(stk.hasMoreTokens()){
        System.out.println(stk.nextToken());
    }
    System.out.println("total tokens:"+stk.countTokens());
}

```

/\*

Output:

this==string-includes=delims

total tokens:4

=====tokens=====

this

string

includes

delims

total tokens:0

\*/

```

import java.util.StringTokenizer; // import 필수

```

```

public static void main(String[] ar){

    String str="this-string-includes=delims";
    StringTokenizer stk=new StringTokenizer(str,"==",true);
    System.out.println(str);
    System.out.println();

    System.out.println("total tokens:"+stk.countTokens());
    System.out.println("=====tokens=====");
    while(stk.hasMoreTokens()){
        System.out.println(stk.nextToken());
    }
    System.out.println("total tokens:"+stk.countTokens());
}

```

/\*

Output:

this-string-includes=delims

total tokens:7

=====tokens=====

this

-

string

```
-
includes
=
delims
total tokens:0
*/
```

StringTokenizer 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/util/StringTokenizer.html>

## 출력

### Print / printf / println 등등

- Import 필요 X

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

        System.out.print(3+"\n");
        System.out.println("안녕하세요");
        System.out.println("안녕하세요" + 3);
        // 자동 엔터
        System.out.printf("%d %c %s", 1 , 'a' , "abc\n");
        // 추가 형식 지정자 알아보기

    }
}
```

형식지정자 : <https://keep-cool.tistory.com/15>

- 한번에 출력은 어떻게?

### BufferedWriter

사용법

```
// BufferedWriter 사용

// 1.Import
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.BufferedWriter;

// 2. 메인메소드에 throws IOException 예외처리 추가
public static void main(String[] args) throws IOException {
}

// 3. bw객체생성
BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(System.out));

// 4. bw.write() 등 메소드 이용하여 출력
```



bw.wirte() -> 스트림 저장  
bw.flush() -> 남아있는 데이터 출력  
bw.close() -> 스트림을 닫음

```
import java.io.BufferedReader;
import java.io.BufferedWriter; // Import필요
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;

public class BufferedWriter_Practice {
    public static void main(String[] args) throws IOException{

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(System.out));

        String s = br.readLine();

        bw.write(s+"\n"); // IOException 필요 입력

        String a = br.readLine();

        bw.write(a+"\n");
        // \n = bw.newLine();

        bw.flush(); //남아있는 데이터를 모두 출력시킴
        bw.close(); //스트림을 닫음

        // 입력 :
        // "안녕하세요"
        // "안녕히계세요"
        // 한꺼번에 출력됨
    }
}
```

BufferedWriter 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/io/BufferedWriter.html>

#### 주의점

- 스트림 크기가 꽉차면 자동 flush() 되는 문제가있음
- StringBuilder / StringBuffer를 통해 보완

## StringBuilder / StringBuffer

```
StringBuilder sb = new StringBuilder();

method:
append() 추가
toString() String으로 변환
```

```
// StringBuilder
```

```

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Sb_practice {
    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

        String a = br.readLine();
        String b = br.readLine();
        String c = br.readLine();

        StringBuilder sb = new StringBuilder();

        sb.append(a+" ");
        sb.append(b+" ");
        sb.append(c+" ");

        String s = sb.toString();
        System.out.println(s);
    }
}

/*
input :
a
b
c
output :
a b c
*/

```

StringBuilder 공식문서 <https://docs.oracle.com/javase/8/docs/api/java/lang/StringBuilder.html>

```

// StringBuffer

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class SBuffer_Practice {
    public static void main(String[] args) throws IOException{

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

        StringBuffer sb = new StringBuffer();

        String a = br.readLine();
        String b = br.readLine();
        String c = br.readLine();
    }
}

```

```

        sb.append(a + " ");
        sb.append(b + " ");
        sb.append(c + " ");

        String d= sb.toString();
        System.out.println(d);
    }
}

/*
input :
i'm
not
Iron man

output:
i'm not Iron man
*/

```

- **String** : 문자열 연산이 적고 멀티쓰레드 환경일 경우
- **StringBuffer** : 문자열 연산이 많고 멀티쓰레드 환경일 경우
- **StringBuilder** : 문자열 연산이 많고 단일쓰레드이거나 동기화를 고려하지 않아도 되는 경우

- StringBuffer 설명 : <https://wikidocs.net/276>
- 다양한 method 알아보기

StringBuffer 공식문서 : <https://docs.oracle.com/javase/8/docs/api/java/lang/StringBuffer.html>

## 참조

- InputStream / OutputStream / 파일입출력 : <https://velog.io/@duck-ach/23.-%EC%9E%85%EC%B6%9C%EB%A0%A5-%EC%8A%A4%ED%8A%B8%EB%A6%BC-InputStream-OutputStream-JAVA>
- 자바 입력 뜯어보기 : <https://st-lab.tistory.com/41>
- 점프투자바 입출력 : <https://wikidocs.net/226>
- BufferedReader 알고쓰자 : <https://velog.io/@roycewon/BufferedReader%EB%A5%BC-%EC%95%8C%EA%B3%A0-%EC%93%B0%EC%9E%90>
- BufferedReader & BufferedWriter : <https://jhnyang.tistory.com/92>
- StringTokenizer & split : <https://reakwon.tistory.com/90>
- 자바8 공식문서 : <https://docs.oracle.com/javase/8/docs/api/overview-summary.html>
- String/StringBuffer/StringBuilder : <https://ifuwanna.tistory.com/221>
- 공식문서 자바 8 기준

## 문제풀이

# BOJ\_15552번\_빠른 A+B

## Scanner로 풀이

```
import java.util.Scanner;


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

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();

        for (int i = 0; i < n; i++) {
            System.out.println(sc.nextInt()+sc.nextInt());
        }
        sc.close();
    }
}

//시간초과
```

문제	결과	메모리	시간	언어
 15552	시간 초과 (33%)			Java 8 / 수정

## BufferedReader / BufferedWriter로 풀이

```
import java.io.*;

// split이용
public class Main {
    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new
        InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new
        OutputStreamWriter(System.out));

        //split이용하여 출력하기 위한 StringBuilder
        StringBuilder sb = new StringBuilder();

        //반복문 순회 숫자 n
        int n = Integer.parseInt(br.readLine());

        // 반복문을 통해 덧셈할 숫자 입력 받기
        for (int i = 0; i < n; i++) {
            String line = br.readLine();
            int split = line.indexOf(" ");
            int a = Integer.parseInt(line.substring(0, split));
            int b = Integer.parseInt(line.substring(split+1));
            sb.append(a+b);
            sb.append("\n");
        }
        bw.write(String.valueOf(sb));
    }
}
```

```

        bw.flush();
        bw.close();
    }
}

```

문제	문제 제목	결과	메모리	시간	언어	코드 길이
15552	빠른 A+B	맞았습니다!!	159276 KB	612 ms	Java 8	707 B

```

import java.io.*;
import java.util.StringTokenizer;

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

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(System.out));
        StringBuilder sb = new StringBuilder();
        int n = Integer.parseInt(br.readLine());

        for (int i = 0; i < n; i++) {
            StringTokenizer st = new StringTokenizer(br.readLine(), " ");
            int a = Integer.parseInt(st.nextToken());
            int b = Integer.parseInt(st.nextToken());
            sb.append(a+b);
            sb.append("\n");
        }
        bw.write(String.valueOf(sb));
        bw.flush();
        bw.close();
    }
}

```

문제	문제 제목	결과	메모리	시간	언어
15552	빠른 A+B	맞았습니다!!	241292 KB	728 ms	Java 8

## BOJ\_2869번\_달팽이는 올라가고 싶다.

### Scanner로 풀이

```

import java.util.Scanner;

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

        Scanner sc = new Scanner(System.in);

        int a = sc.nextInt();
        int b = sc.nextInt();
        int v = sc.nextInt();
    }
}

```

```

        int today = a - b;
        int c = v - b;
        int n = c / today;
        if(c % today != 0) n++;

        System.out.println(n);
    }
}

```

문제	결과	메모리	시간	언어
🏆 2869	맞았습니다!! ✓	12824 KB	116 ms	Java 8 / 수정

## BufferedReader / BufferedWriter로 풀이

```

import java.io.*;
import java.util.StringTokenizer;
public class Main {
    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(System.out));

        StringTokenizer st = new StringTokenizer(br.readLine(), " ");
        int a = Integer.parseInt(st.nextToken());
        int b = Integer.parseInt(st.nextToken());
        int v = Integer.parseInt(st.nextToken());

        int today = a - b; // 하루에 올라가는 거리
        int c = v - b; // 총올라가야할 거리 수정 만약 a만큼 올라가고 정상에 도착한다면 떨어지지 않기 때문에
        //마지막날 a만큼 올라가고 b만큼 떨어지지 않음

        int n = c / today; // 총올라가야할 거리를 하루에 올라가는 거리로 나눔
        if(c % today != 0) n++; // Int형식을 나누면 소수점 이하 버림
        // 소수점이 존재한다면 하루 더필요

        bw.write(n + ""); // 숫자로 변환 편하게
        bw.flush();
        bw.close();
    }
}

```

문제	결과	메모리	시간	언어
🏆 2869	맞았습니다!! 🚩	11568 KB	76 ms	Java 8 / 수정