Day14 (H1)

20150827

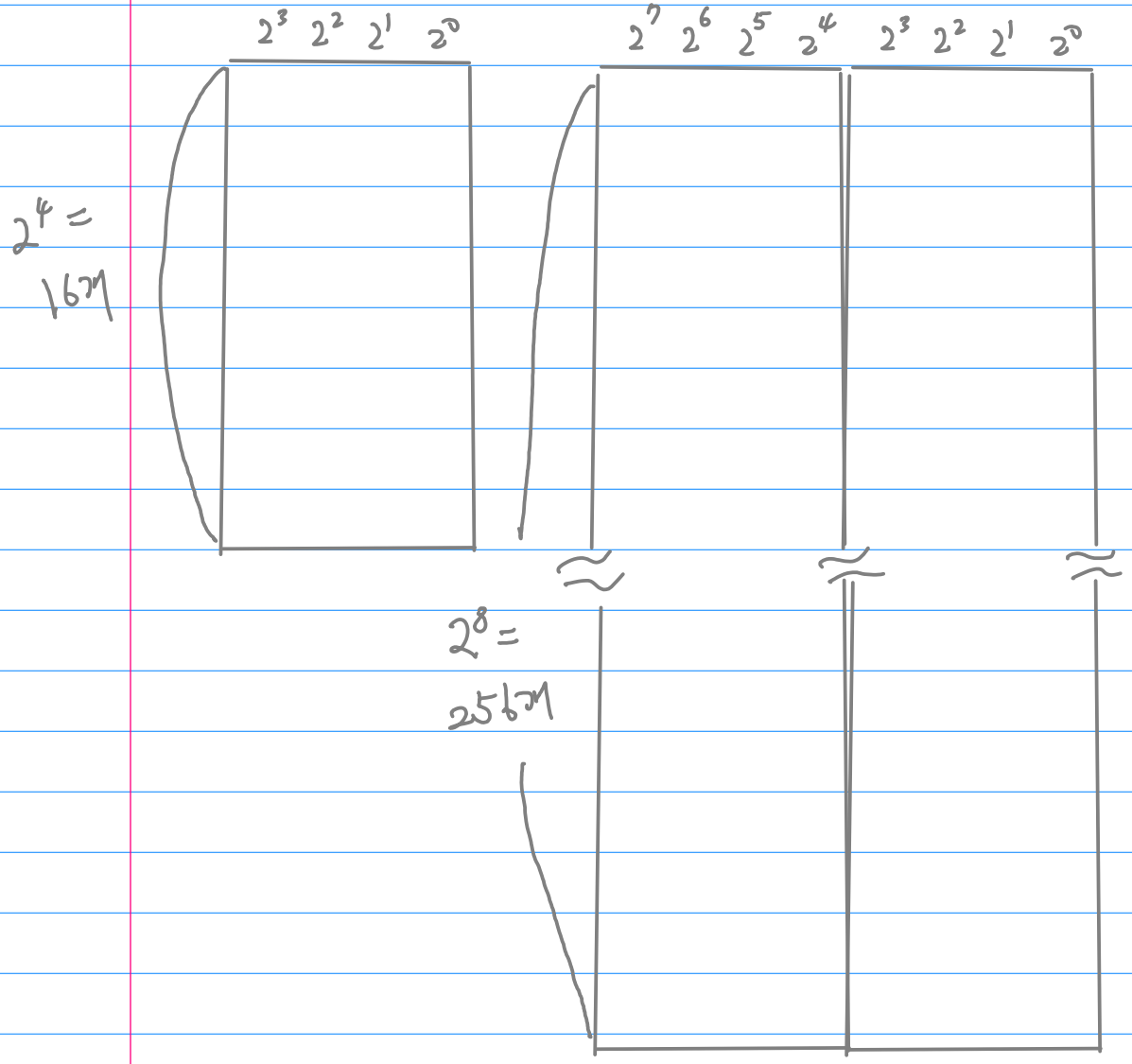
Byte primitive type
Exception Handling
File IO

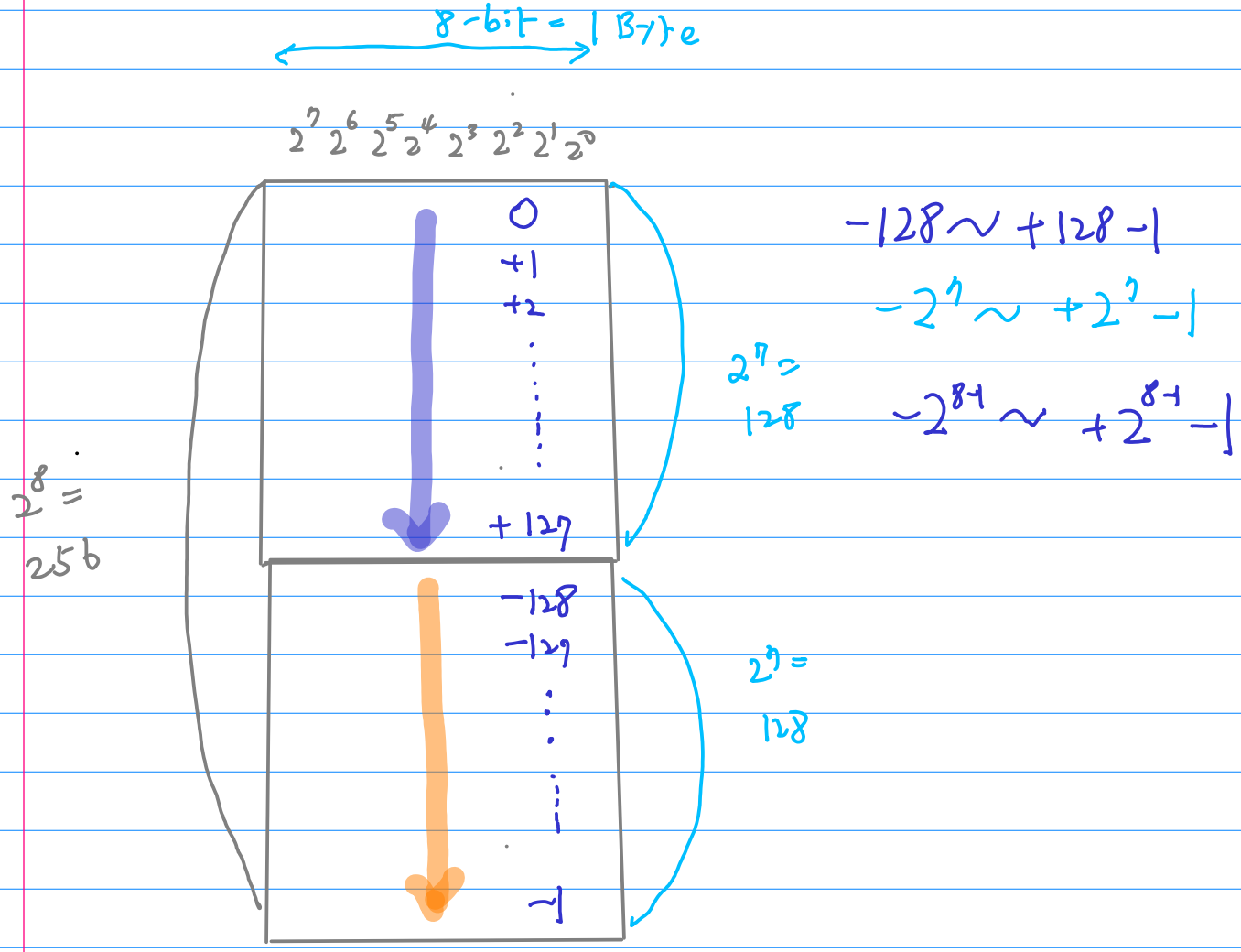
Byte primitive type 8-bit

integer $-128 \sim +127$

$-2^7 \sim +2^7 - 1$

$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$





* Byte 1-Byte (8-bit) $-2^{8-1} \sim +2^{8-1}-1$

* Short 2-Byte (16-bit) $-2^{16-1} \sim +2^{16-1}-1$

* int 4-Byte (32-bit) $-2^{32-1} \sim +2^{32-1}-1$

Primitive Type	Size	Minimum Value	Maximum Value	Wrapper Type
char	16-bit	Unicode 0	Unicode 216-1	Character
byte	8-bit	-128 -2^{8-1}	+127 $+2^{8-1}-1$	Byte
short	16-bit	-215 (-32,768) -2^{16-1}	+215-1 (32,767) $+2^{16-1}-1$	Short
int	32-bit	-231 (-2,147,483,648) -2^{32-1}	+231-1 (2,147,483,647) $+2^{32-1}-1$	Integer
long	64-bit	-263 (-9,223,372,036,854,775,808) -2^{64-1}	+263-1 (9,223,372,036,854,775,807) $+2^{64-1}-1$	Long
float	32-bit	Approx range 1.4e-045 to 3.4e+038		Float
double	64-bit	Approx range 4.9e-324 to 1.8e+308		Double
boolean	1-bit	true or false		Boolean

the same method name

different argument types

```
public static void pr(int x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}

public static void pr(double x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}

public static void pr(boolean x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}

public static void pr(char x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}

public static void pr(char x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}

public static void pr(byte x[]) {
    int i;
    for (i=0; i<x.length; ++i)
        System.out.println("x[" + i + "]= " + x[i]);
}
```

Array with Initialization

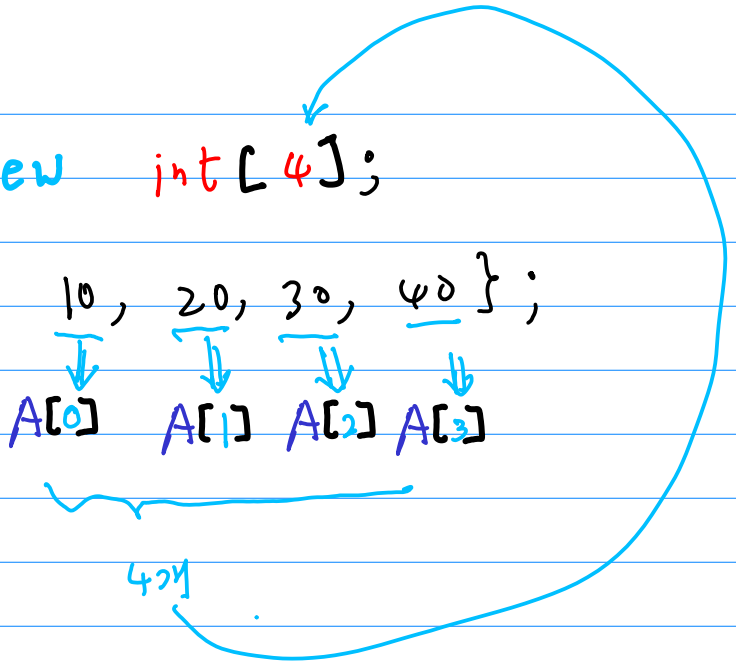
```
public static void main(String[] args) {
    // TODO Auto-generated method stub

    int     A[] = { 10,     20,     30,     40   };
    double  B[] = { 0.1,    0.2,    0.3,    0.4  };
    boolean C[] = { true,   true,   false,  true };
    char    D[] = { 'a',    'b',    'c',    'd'  };
    byte    E[] = { -128,   -127,   126,    127  };
    // byte     F[] = { -129,   -127,   126,    127  };
    // byte     G[] = { -128,   -127,   126,    128  };

    pr(A);
    pr(B);
    pr(C);
    pr(D);
    pr(E);



}
```

```
int [] A = new int [4];
```

```
int A[] = { 10, 20, 30, 40 };
```

① `int [] A = new int[4];`

② `int A[] = { 10, 20, 30, 40 };`



`int A[] = { 10, 20, 30, 40 };`

① main method 내에서 예외처리

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

    try {
        FileReader in2 = new FileReader("TTTT.txt");
        Scanner    s   = new Scanner( in2 );

        int x;

        while (s.hasNext()) {
            x = s.nextInt();
            System.out.println( x );
        }

    } catch (IOException e ) {
        System.out.println("TTTT.txt does not exists");
    }
}
```

이곳에서 Exception이 발생할 수 있음

발생한 Exception중 IOException type의 Exception을

이곳에서 처리함

```
TTTT.txt does not exists
```

② main method를 call하는 다른 method가 예외처리를 하도록 함.

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

        FileReader in2 = new FileReader("TTTT.txt");
        Scanner    s   = new Scanner( in2 );

        int x;

        while (s.hasNext()) {
            x = s.nextInt();
            System.out.println( x );
        }
}
```

main 함수를 call 하는 함수에게 main 함수 내에서 IOException 예외가 일어날 수 있음을 알려줌.

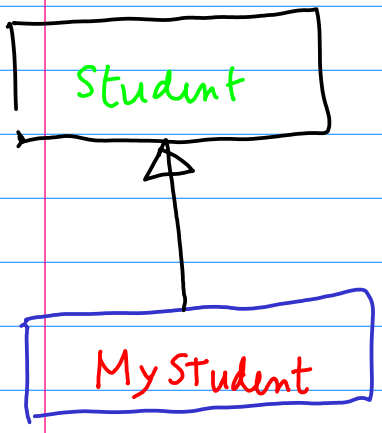
```
Exception in thread "main" java.io.FileNotFoundException: TTTT.txt (No such
        at java.io.FileInputStream.open(Native Method)
        at java.io.FileInputStream.<init>(FileInputStream.java:146)
        at java.io.FileInputStream.<init>(FileInputStream.java:101)
        at java.io.FileReader.<init>(FileReader.java:58)
        at ExceptionTest.main(ExceptionTest.java:34)
```

Super class , Sub class

Code view

```
Class Students {
    ...
}
```

```
Class MyStudent extends Student {
    ...
}
```



Super class

Sub class

Subclass **extends** Super class

MyStudent **extends** Student

Sub class의 생성자 함수에서
Super class의 생성자 함수 부르기

Super class

```
Class Students {

    Student ( ) { ... }
    Student ( int x, int y, int z) { ... }

}
```

Sub class

```
Class MyStudent extends Student {

    MyStudent ( ) {
        Super ( );
    }

    MyStudent ( int x, int y, int z) {
        Super ( int x, int y, int z);
    }
```

```java
class MyStudent extends Student {
    int StID;
    String Name;

    //MyStudent() { setKor(0); setEng(0); setMath(0);}
    //MyStudent(int x, int y , int z) { setKor(x); setEn
    MyStudent() { super();}
    MyStudent(String s, int i, int x, int y , int z) {
        super(x, y, z);
        setName(s);
        setStID(i);
    }
```

Kor Eng Math

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

S[0] = new MyStudent("Park",  20150001, 99, 45, 50);
S[1] = new MyStudent("Kim",   20150002, 88, 55, 80);
S[2] = new MyStudent("Lee",   20150003, 77, 65, 90);
S[3] = new MyStudent("Baker", 20150004, 66, 75, 80);
S[4] = new MyStudent("John",  20150005, 55, 85, 90);
```