

능동적 사고 방식의

java

강사 박주병

Park Ju Byeong

Park Ju Byeong



Part04 반복문.

01 for

02 while

03 do while, 중첩 반복문

04 실습 문제

디버그

- 프로그램을 한 단계씩 실행하며 실행 상태를 추적하여 오류를 찾아내는 방법

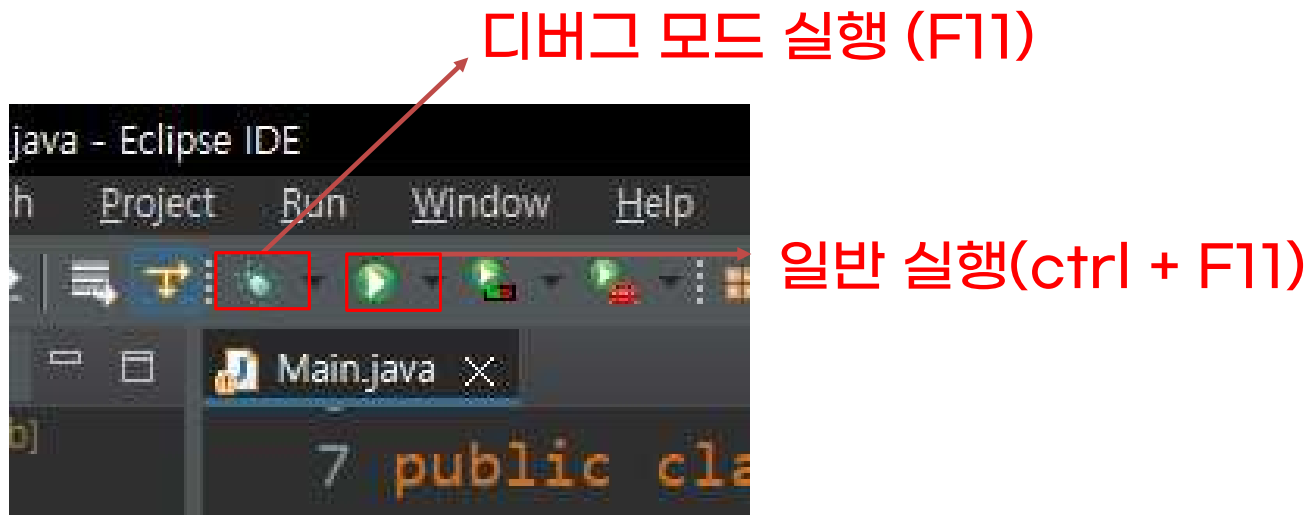
```
1 package joo.강의4;
2
3 import java.util.Scanner;
6
7 public class Main {
8
9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11
12         int a= 10;
13         int b=20;
14
15         a=30;
16         b=50;
17
18     }
19
20 }
```

실행 대기중

12번째 라인은 실행이 되었기에 a변수가 생성되었고 10이 들어 있다.

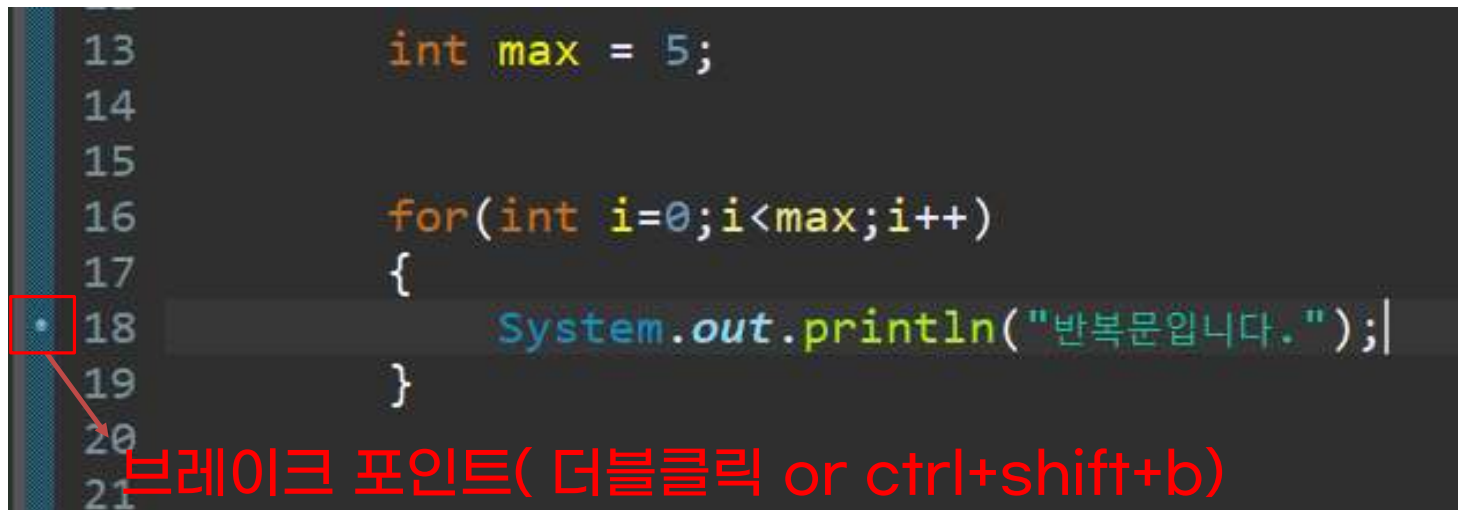
Name	Value
no method return	
args	String[0] (id=20)
a	10

디버그 실행방법



브레이크 포인트

- 디버그 모드로 실행되었을시 브레이크포인트(중단점)을 만나면 실행이 일시중지 된다.

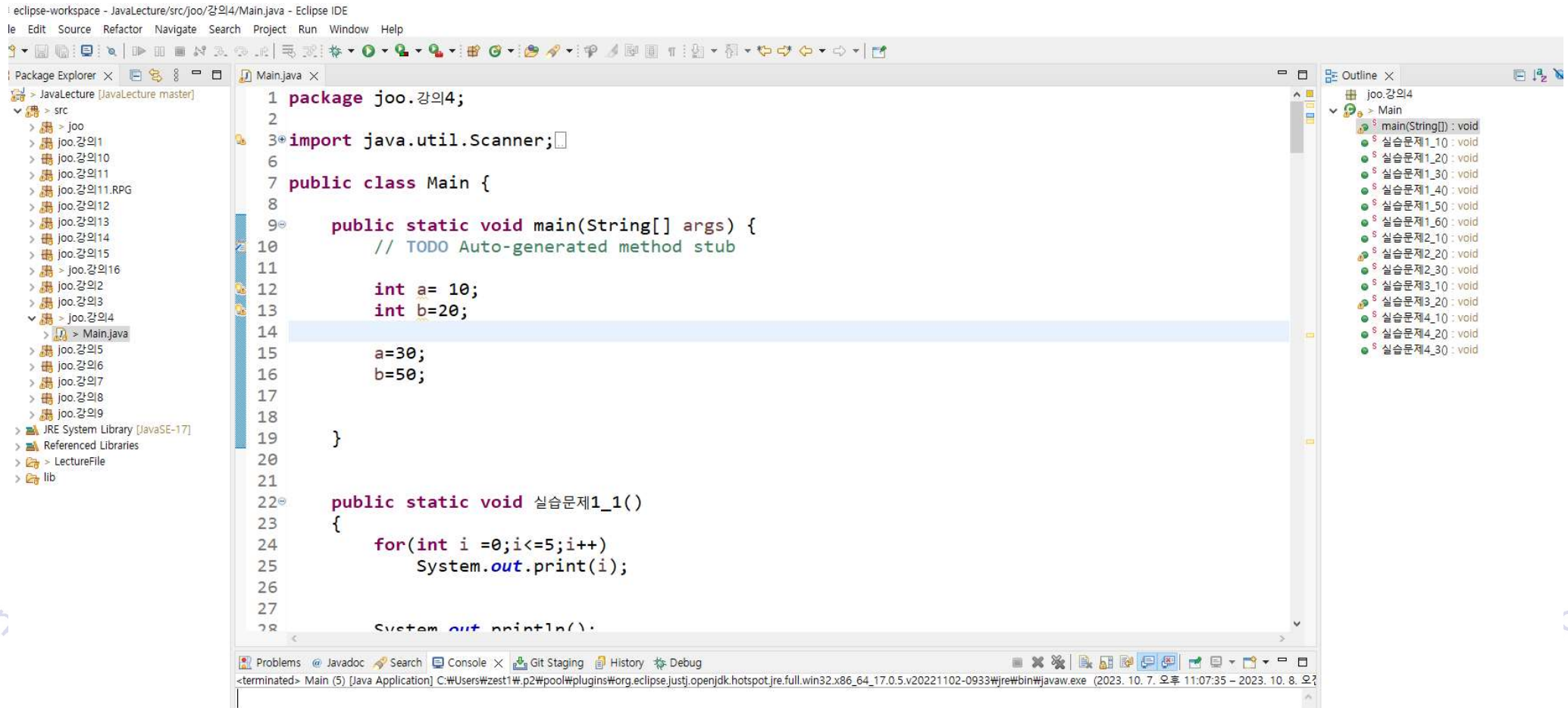


```
13      int max = 5;
14
15
16      for(int i=0;i<max;i++)
17      {
18          System.out.println("반복문입니다.");
19      }
20
21
```

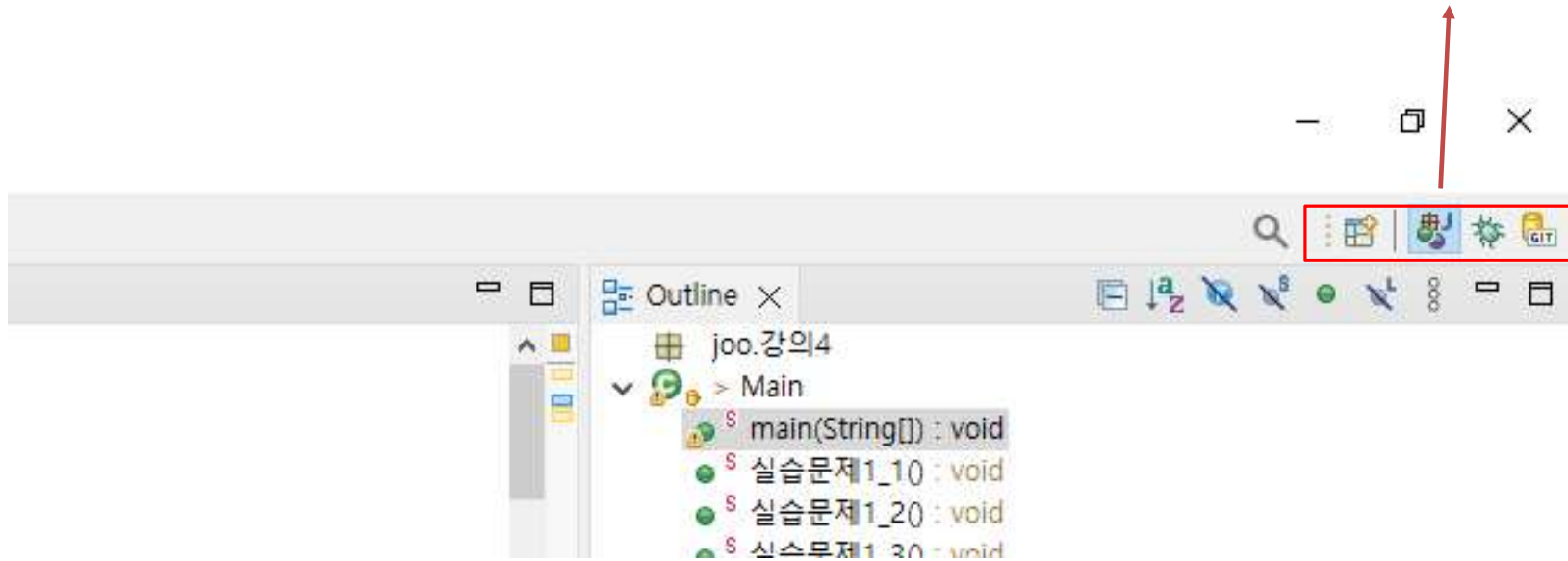
브레이크 포인트(더블클릭 or ctrl+shift+b)

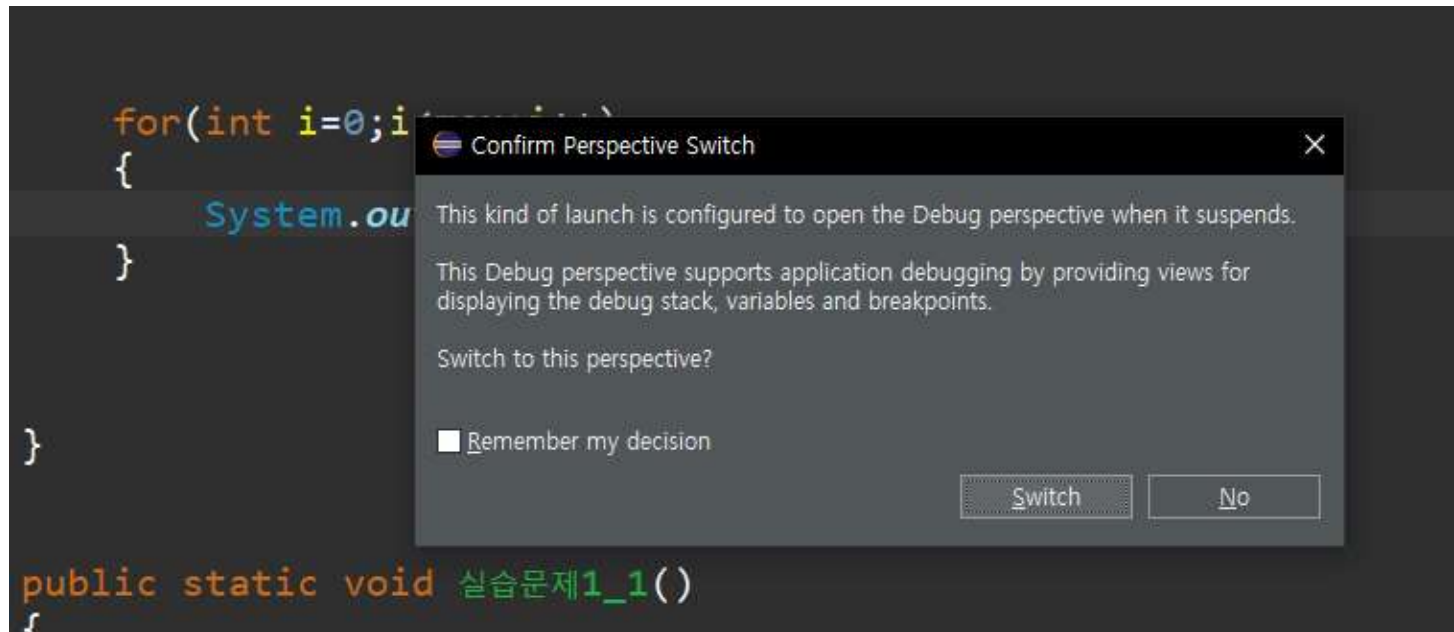
이클립스 퍼스펙티브

- 이클립스의 화면 레이아웃 구성



퍼스펙티브 전환버튼





디버그 모드로 프로그램이 일시중지 될때 디버그용 퍼스펙티브로 전환된다.

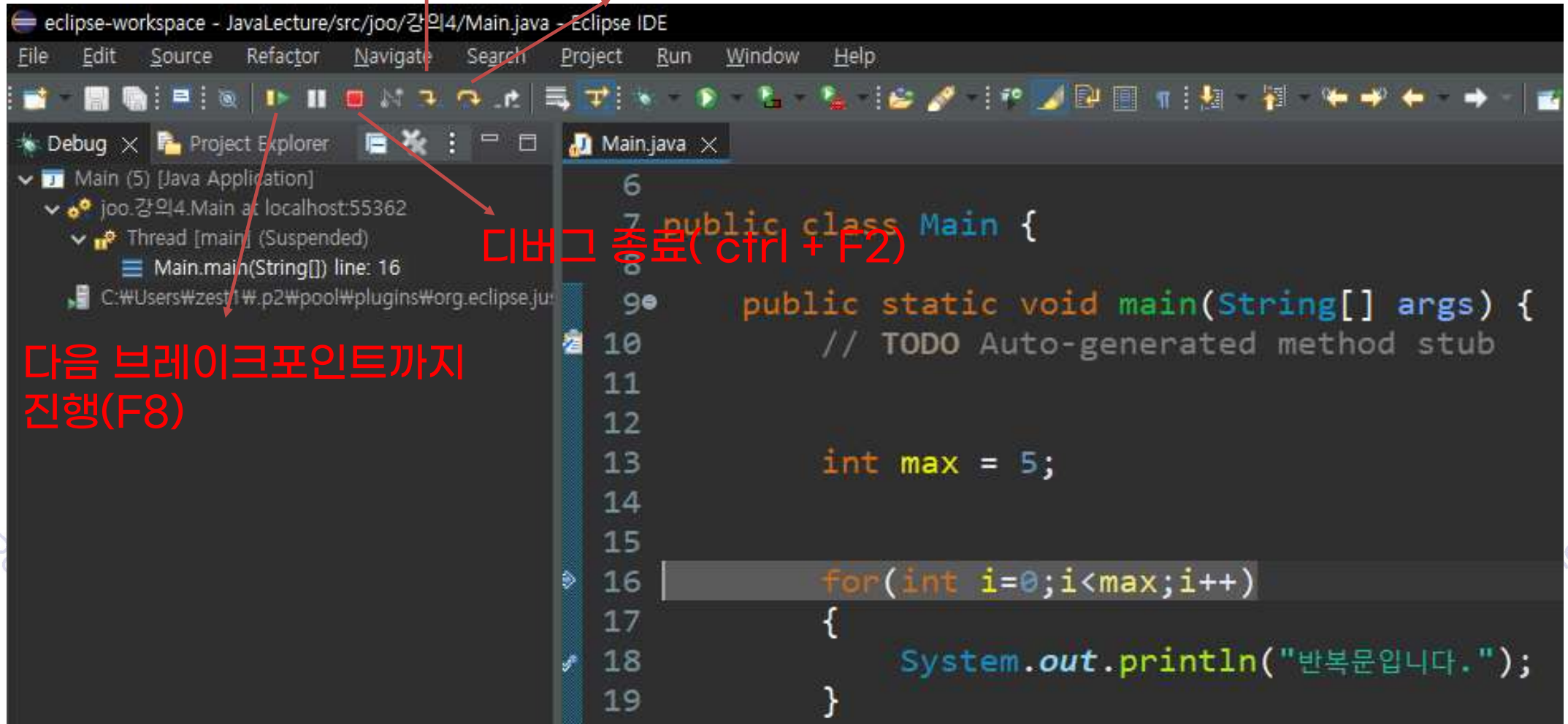
디버그 진행

메서드 내부로 진입(F5)

다음줄로 한칸 진행(F6)

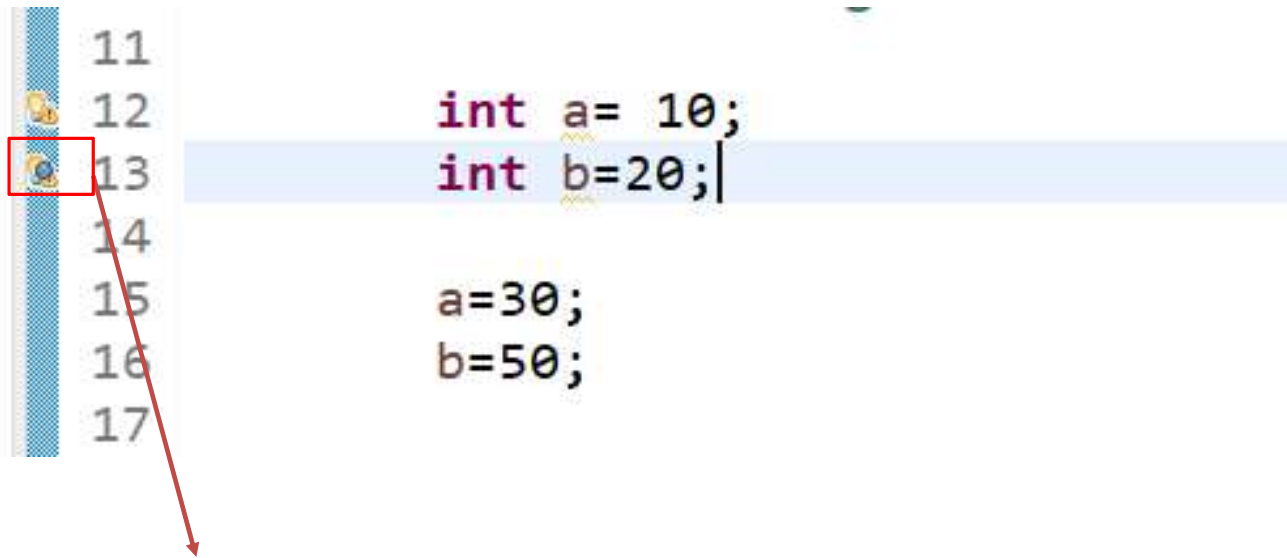
디버그 종료(ctrl + F2)

다음 브레이크포인트까지
진행(F8)

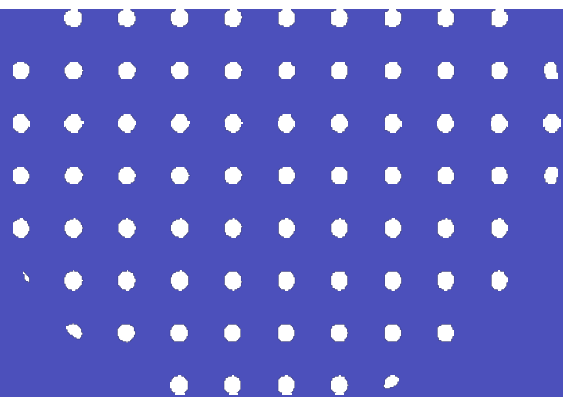


디버그 연습

- 브레이크 포인트를 지정하여 디버그 모드로 진입한후 한줄씩 진행하면서 변수 값의 변화를 살펴보자.



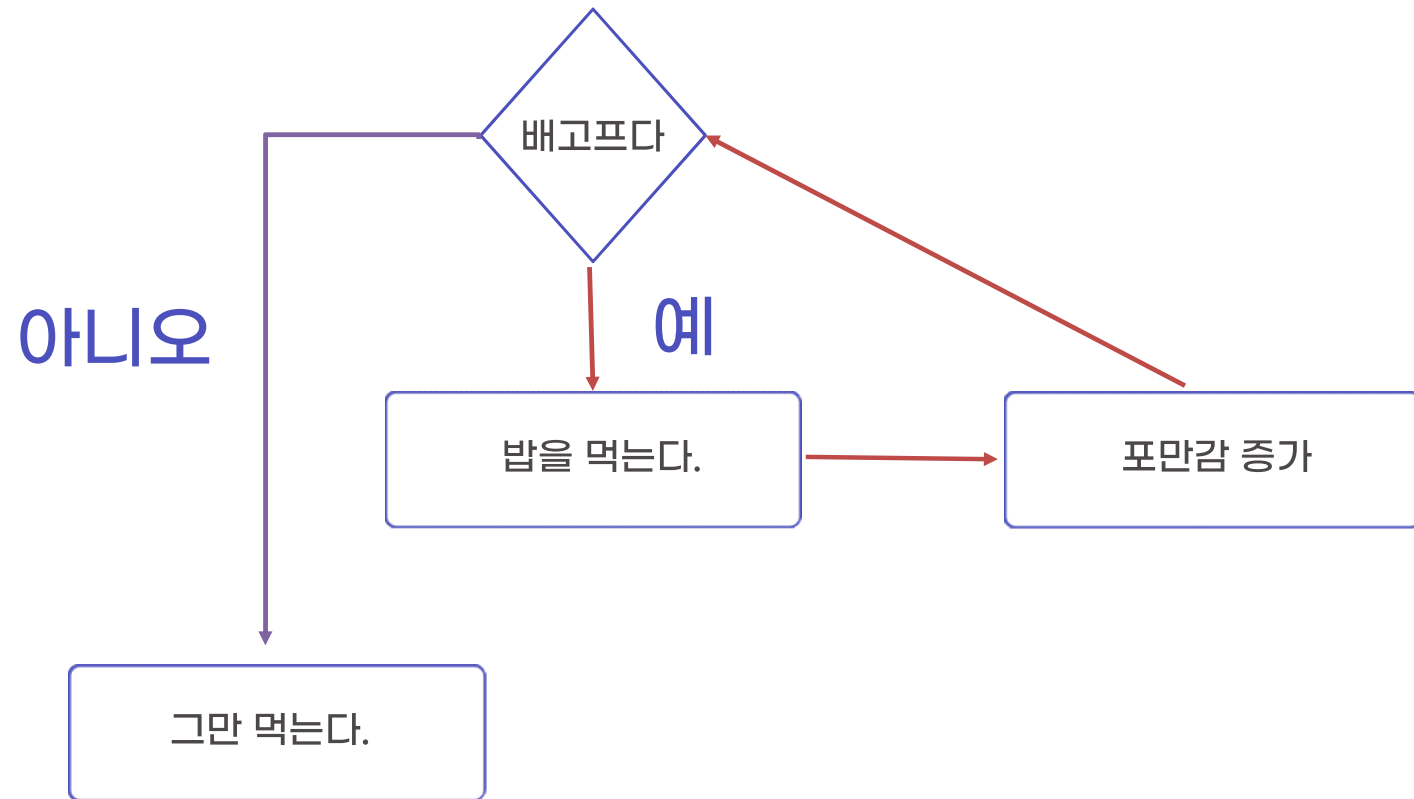
브레이크 포인트



01 —
for



조건식이 참(true)이면 계속 반복하여 실행한다



for문

For(초기화 ; 조건식;증감식)

```
for (int i =0 ; i<3;i++)  
{  
    System.out.println("study JAVA");  
}
```

변수 선언 및 초기화
(최초 한번 실행)

조건식(true이면
실행한다.)

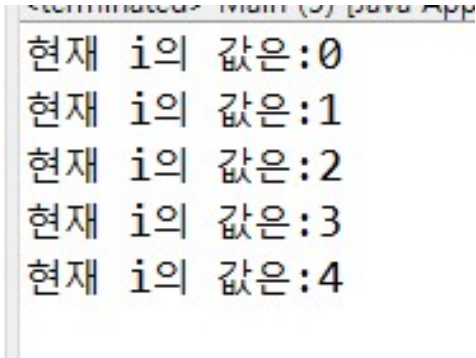
For문의 코드를 모두 실행하면 마지막에 실행된다.

실행순서: 초기화 -> 조건식 -> for문 내부실행 -> 증감식

for문 연습

- 아래의 코드를 작성하여 실행해보자.

```
for(int i =0;i<5;i++)  
{  
    System.out.println("현재 i의 값은:"+i);  
}
```



Terminated: main (0) [Java App]

```
현재 i의 값은:0  
현재 i의 값은:1  
현재 i의 값은:2  
현재 i의 값은:3  
현재 i의 값은:4
```

for문 초기화

```
for(int i =0 ; i<3;i++)  
{  
    System.out.println("study  
}
```

일반적인 변수 선언과 초기화와 동일하다.

```
for(int i =0, j=13 ; i<3;i++)  
{  
    System.out.println("s  
}
```

여러 개의 변수를 선언하고 초기화도 가능하다
(권장하지 않음, 반복을 결정하는 변수만 선언하고 그 외에 필요하다면
for문 외부에서 따로 선언하는게 일반적)

```
int num= 5;  
for(num=1 ; num<3;num++)  
{  
    System.out.println("study JAVA");  
}
```

For문 외부에서 선언된 변수를 활용해도 됨

```
int num= 5;  
for(; num<3;num++)
```

조건식과 증감식이 외부에서 선언된 변수를
활용한다면 없어도 된다.

for문 조건식

```
for(int i=0; i<3; i++)  
{
```

조건식이 true이면 실행
false이면 종료된다.

```
for(int i=0; i>0; i++)  
{
```

False가 될 수 없는 구조이면
무한루프에 빠진다.

```
for(int i=0; true; i++)
```

→ 실행가능하며 무한루프이다.

```
for(int i=0; ; i++)  
{
```

→ 실행가능하며 무한루프이다.

```
int max = 5;
```

```
for(int i=0; i<max; i++)  
{
```

→ 변수가 올수 있다.

for문 증감식

```
for(int i=0; i<3; i++)  
{
```

→ For문이 실행되고 마지막에 실행되며 i값을 1증가 시킨다.

```
for(int i=5; i>3; i--)  
{
```

→ 감소도 가능하다.

```
for(int i=0; i<5; i=i+1)  
{
```

→ 다른 형태의 증감식도 가능하다.

```
for(int i=0; i<5; System.out.println("될까?"))  
{
```

→ 사실상 어떤 코드든 들어갈 수 있으나 의미상 for문의 실행여부와 관계되는 값의 변화를 넣는것이 올바르다.

```
for(int i=0; i<5; ;)  
{
```

→ 생략가능하며 for문 내부에서 증감을 해도 된다.

```
for(;;)  
{
```

→ 무한루프!

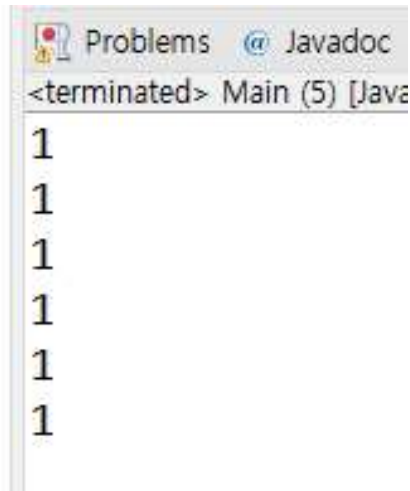
1부터 시작하면 편할텐데 왜 예제들이 0부터 시작할까?

```
for(int i =0;i<5;i++)  
{  
    System.out.println("현재 i의 값은:"+i);  
}
```

반복문 내부에서 변수 생성

```
for(int i = 0; i <= 5; i++)  
{  
    int b = 0;  
    System.out.println(++b);  
}
```

중괄호 내부에서 만들어진 변수는 반복문이 한 사이클이 끝나면 삭제되며 다시 시작될때 새롭게 만들어진다.



Problems Javadoc
<terminated> Main (5) [Java
1
1
1
1
1
1

```
int b=0;
```

```
for(int i =0;i<=5;i++)  
{
```

```
    System.out.println(++b);  
}
```

반복문 외부에서 만들어져서
반복문 한사이클이 끝나도
사라지지 않는다.

즉 반복될수록 값이 누적된다.

<terminated> via

1
2
3
4
5
6

중첩 반복문

```
for(int i =0;i<=10;i++)  
{  
    for(int j =0;j<=10;j++)  
    {  
        System.out.print('*');  
    }  
    System.out.println();  
}
```

```
<terminated> main [Jav  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

중첩 반복문

```
for(int i =0;i<=10;i++)  
{  
    for(int i =0;i<=10;i++)  
    {  
        System.out.print('*');  
    }  
    System.out.println();  
}
```

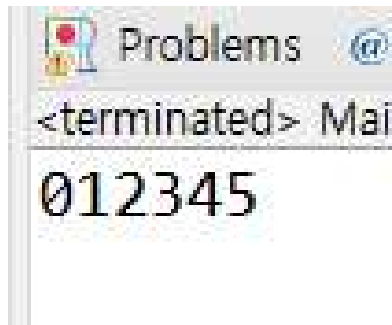
이미 선언된 변수이다.

1-1 실습문제 (normal)

For 반복문을 이용하여 아래와 같이 출력해보자

- System.out.print() 를 사용하면 줄이 넘어가지 않는다.

```
System.out.print(i);
```



1-1 문제풀이 (normal)

1-2 실습문제 (normal)

숫자 1 부터 10까지의 합계를 출력하시오

```
int sum = 0;
```

```
1부터0까지의 합:0  
1부터1까지의 합:1  
1부터2까지의 합:3  
1부터3까지의 합:6  
1부터4까지의 합:10  
1부터5까지의 합:15  
1부터6까지의 합:21  
1부터7까지의 합:28  
1부터8까지의 합:36  
1부터9까지의 합:45  
1부터10까지의 합:55
```

1-2 문제풀이 (normal)

byeong

Park Ju eong

1-3 실습문제 (hard)

0부터 10까지 증가 하는 값과 10부터 0까지 감소하는 값을 출력하시오

- 중첩 반복문은 필요 없음

```
<terminate
0 10
1 9
2 8
3 7
4 6
5 5
6 4
7 3
8 2
9 1
10 0
```

1-3 문제풀이 (hard)

byeong

Park Ju-ong

1-4 실습문제 (hard)

아래의 그림과 같이 삼각형 모양의 별을 출력하시오

- 중첩 반복문을 사용해야 한다.
- System.out.println(); 을 사용하면 줄을 넘길수 있다.
- System.out.print("*"); 사용시 줄을 넘기지 않고 출력한다.

```
<terminated> m
*
**
***
****
*****
```

1-4 문제풀이 (hard)

1

Byeong

Park Ju
ong

1-5 실습문제 (hard)

아래의 그림과 같이 구구단을 출력하시오(9단까지)

```
<terminated> main [Jav  
2 x 1 = 2  
2 x 2 = 4  
2 x 3 = 6  
2 x 4 = 8  
2 x 5 = 10  
2 x 6 = 12  
2 x 7 = 14  
2 x 8 = 16  
2 x 9 = 18  
2 x 10 = 20
```

1-5 문제풀이 (hard)

Park Ju Byeong

Park Ju Byeong

1-6 실습문제 (expert)

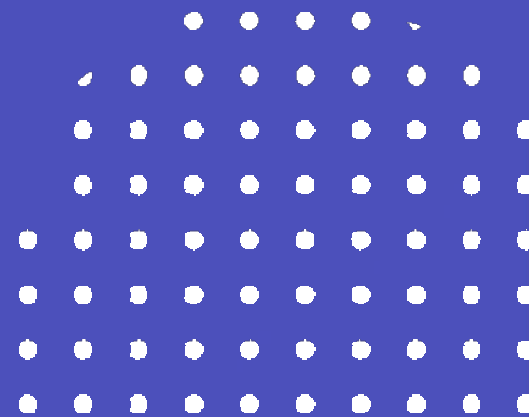
아래의 모양을 출력하시오



1-6 문제풀이 (expert)

—• 02

while



while

```
int i = 0;
```

```
while (i < 5)  
{  
    i++;  
}
```

→ boolean타입이 들어가야하며
true이면 계속 실행한다.

```
while()  
{  
  
}
```

조건식 생략 불가능

```
while(true)  
{  
  
}
```

무한루프

for문이 있는데 왜 while문이 필요할까?

for문 과의 차이점

```
int i=0; → 초기화
while(i<5) → 조건식
{
    System.out.println(i);
    i++; → 증감식
}
```

```
for(int i=0; i<5; i++)
{
    System.out.println(i);
}
```



for

횟수가 정해져 있을때 사용한다

```
for(int i = 1 ;i<=5; i++)  
{  
    System.out.println(i+" 번째 패스워드를 틀렸습니다."+ (5-i)+"번 남았습니다.");  
}
```

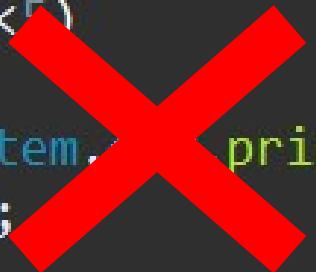
while

특정 조건을 만족할때까지일때 사용한다

```
int life = 100;  
  
while(life >0) // 캐릭터의 체력이 있을때만  
{  
  
}
```



```
int i=0;
while(i<5)
{
    System.out.println(i);
    i++;
}
```



→ 이럴거면 for문 쓰자

횟수를 정해놓고 for문처럼
사용은 가능하다

2-1 실습문제 (normal)

While 문을 이용해 1부터 10까지 출력하시오

```
<C:\Program Files\Java\jdk-1.8.0_101\bin> java  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

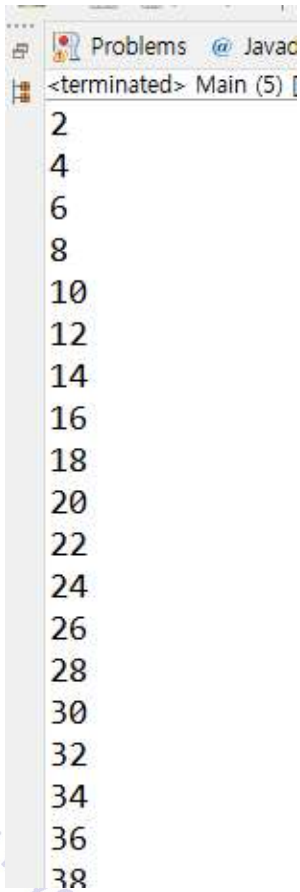
2-1 문제풀이 (normal)

Park Ju Byeong

Park Ju Byeong

2-2 실습문제 (normal)

while문을 이용해 1부터 100 숫자중 짝수만 출력하시오



```
<terminated> Main (5) [
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
```

2-2 문제풀이 (normal)

Park Ju Byeong

Park Ju Byeong

2-3 실습문제 (hard)

1+2+3+4... 합계를 누적하여 몇까지 올라가야 누적합계가 150이상 되는지를 출력하시오

```
<terminated> main [Java Application] C:\Users\W
누적 합계:28 현재 숫자 : 7
누적 합계:36 현재 숫자 : 8
누적 합계:45 현재 숫자 : 9
누적 합계:55 현재 숫자 : 10
누적 합계:66 현재 숫자 : 11
누적 합계:78 현재 숫자 : 12
누적 합계:91 현재 숫자 : 13
누적 합계:105 현재 숫자 : 14
누적 합계:120 현재 숫자 : 15
누적 합계:136 현재 숫자 : 16
누적 합계:153 현재 숫자 : 17
```

2-3 문제풀이 (hard)

Park Ju Byeong

Park Ju Byeong

2-4 실습문제 (hard)

사용자로부터 숫자를 입력 받아 while을 이용해 각 자리의 합을 구하시오

예) 12345 -> 1+2+3+4+5 =15

```
숫자를 입력하세요:  
12345  
현재 합계 : 5  
현재 합계 : 9  
현재 합계 : 12  
현재 합계 : 14  
현재 합계 : 15  
각 자리의 합계:15
```

```
Scanner scan = new Scanner(System.in);  
  
System.out.println("숫자를 입력하세요:");  
int number = scan.nextInt();  
  
int sum = 0;  
while(number > 0){  
    //현재 합계, 현재 숫자 자리의 숫자를 구하고 더함  
    sum += number % 10;  
  
    System.out.println("현재 합계 : " + sum);  
  
    //다음에 숫자 자리의 숫자를 구함  
    number /= 10;  
}  
  
System.out.println("각 자리의 합계:" + sum);
```


2-4 문제풀이 (hard)

Park Ju By

Park Ju Byeong



03

do while
break
continue:::



do ~ while문

```
int i=0;
```

```
do  
{
```

```
    System.out.println(i);
```

```
    i++;
```

```
}while(i<5);
```

→ 최초 1번은 무조건 실행된다.

→ 세미콜론을 붙여 줘야 한다.

break

```
int i=0;
while(true)
{
    if(i==5)
        break; // 반복문 탈출!

    i++;
}
```

Break문을 만나면 그 뒤는 더 이상
실행되지 않고 반복문이 **종료**된다.

continue

```
for(int i =0;i<10;i++)  
{  
    if(i%2==0)  
        continue;  
    System.out.println(i);  
}
```

Continue문을 만나면 반복문 내부의
끝지점으로 간다.(해당 회차만 종료된다)

1
3
5
7
9

04

실습문제

3-1 실습문제 (normal)

10부터 1까지 숫자를 반복해서 출력하시오



```
Problems Javadoc S
<terminated> Main (5) [Java App
10
9
8
7
6
5
4
3
2
1
```

3-1 문제풀이 (normal)

Park Ju Byeong

Park Ju Byeong

3-2 실습문제 (normal)

do while문을 이용하여 숫자 맞추기 게임을 만들어보자

- 컴퓨터는 1~100사이의 랜덤한 숫자를 저장하고 유저는 숫자를 입력받아 값을 맞추도록하자
- 정답을 맞출시 몇번만에 성공한것인지도 출력해보자

```
<terminated> main [Java Application] C:\Users\wze  
15  
15 보다 작습니다.  
1~100 사이의 숫자를 입력하세요:  
7  
7 보다 작습니다.  
1~100 사이의 숫자를 입력하세요:  
3  
3 보다 작습니다.  
1~100 사이의 숫자를 입력하세요:  
2  
정답입니다. 6 번 시도하였습니다.
```

```
int count=0;  
Scanner scanner = new Scanner(System.in); //입력을 받기 위한 스캐너 객체를  
  
int computer =(int)(Math.random()*100)+1;  
int user;  
  
do {  
    System.out.println("숫자 맞추기 게임 시작. 1~100 사이의 숫자를 입력하세요.");  
    user = scanner.nextInt();  
    if (computer < user) {  
        System.out.println("입력하신 숫자보다 작습니다.");  
    } else if (computer > user) {  
        System.out.println("입력하신 숫자보다 큼니다.");  
    } else {  
        System.out.println("정답입니다. " + count + "번 시도하였습니다.");  
    }  
    count++;  
} while (user != computer);
```

탐색 알고리즘

1. 순차탐색 : 모든 데이터를 전부 확인하면서 찾아낸다.
2. 이진탐색 : 정렬된 데이터에서 절반씩 줄여나가면서 찾아낸다.
3. 해시 탐색 : 데이터와 그 위치를 연결 지어 보관하고 해시알고리즘을 이용해 탐색

빅오표기법과 시간복잡도

1. 순차탐색 : 데이터가 100개면 최악의 경우 100번만에 찾는다. $O(N)$
2. 이진탐색 : 데이터가 100개면 최악의 경우 7번만에 찾는다. $O(\log n)$
3. 해시 탐색 : 데이터가 100개면 해시충돌이 없다면 1번, 충돌 한다면 $O(1)$, $O(n)$
최악의 경우 100번만에 찾는다.

3-2 문제풀이 (normal)

Park Ju Byeong

yeong

3-3 실습문제 (hard)

1~100 사이의 소수를 출력하여보자

소수: 1과 자기자신으로만 나누어 떨어지는수

- 중첩반복문을 사용해야 한다.



```
Problems Javadoc Declaration Search Console X Git Staging History  
<terminated> main [Java Application] C:\Users\Wzest1\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot  
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53
```

3-3 문제풀이 (hard)

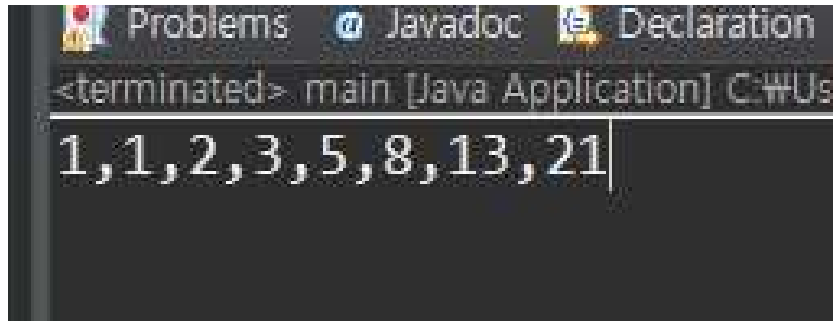
pc

y

3-4 실습문제 (expert)

1과 1부터 시작 하는 피보나치수열로 12번째 숫자가 무엇인지 출력하여보자
피보나치 수열 : 앞의 두 수를 더하여 다음의 수를 만들어 내는 수열이다.

ex) 1, 1, 2, 3, 5, 8, 13



```
Problems Javadoc Declaration  
<terminated> main [Java Application] C:\#Us  
1,1,2,3,5,8,13,21
```

3-4 문제풀이 (expert)

fibonacci

3-5 실습문제 (expert)

아래의 그림과 같이 구구단을 출력하시오

- 일정한 간격만큼 띄우고 싶으면 \t 를 문자열에 포함시키면 된다.

ex) “2*1=2 \t 3*1=3”

```
2*1=2    3*1=3    4*1=4
2*2=4    3*2=6    4*2=8
2*3=6    3*3=9    4*3=12

5*1=5    6*1=6    7*1=7
5*2=10   6*2=12   7*2=14
5*3=15   6*3=18   7*3=21

8*1=8    9*1=9
8*2=16   9*2=18
8*3=24   9*3=27
```

3-5 문제풀이 (expert)



THANK YOU



강사 박주병