

JAVA 프로그램 실습

경북대학교
소프트웨어융합과
교수 배희호

제어의 이동

- JAVA는 Program 제어를 이동시키기 위해 break, continue, return 문을 제공
- 이 문장들은 Program의 수행 순서를 변화시키는 역할을 함
- 가장 많이 사용 되어 온 제어 이동 명령어는 goto문
 - goto문은 Programmer가 원하는 문장으로 직접 실행을 이동하는데 사용
 - JAVA에서는 지원되지 않음

break문과 continue문

■ break 문

- 반복문(for, while, do ~ while)과 선택 제어문중 switch ~ case문과 사용
- 조건에 관계없이 break문을 만나면 break문을 기준으로 하여 가장 가까운 루프에서 한 겹만 탈출

■ continue문

- 반복문(for, while, do ~ while)과 사용
- 반복문에서 continue 문을 만나면 그 이후의 반복문들의 실행을 생략(중단)하고, 그 다음의 반복 실행을 계속하고자 할 목적으로 사용

break문과 continue문

```
for (expr1; expr2; expr3) {
```

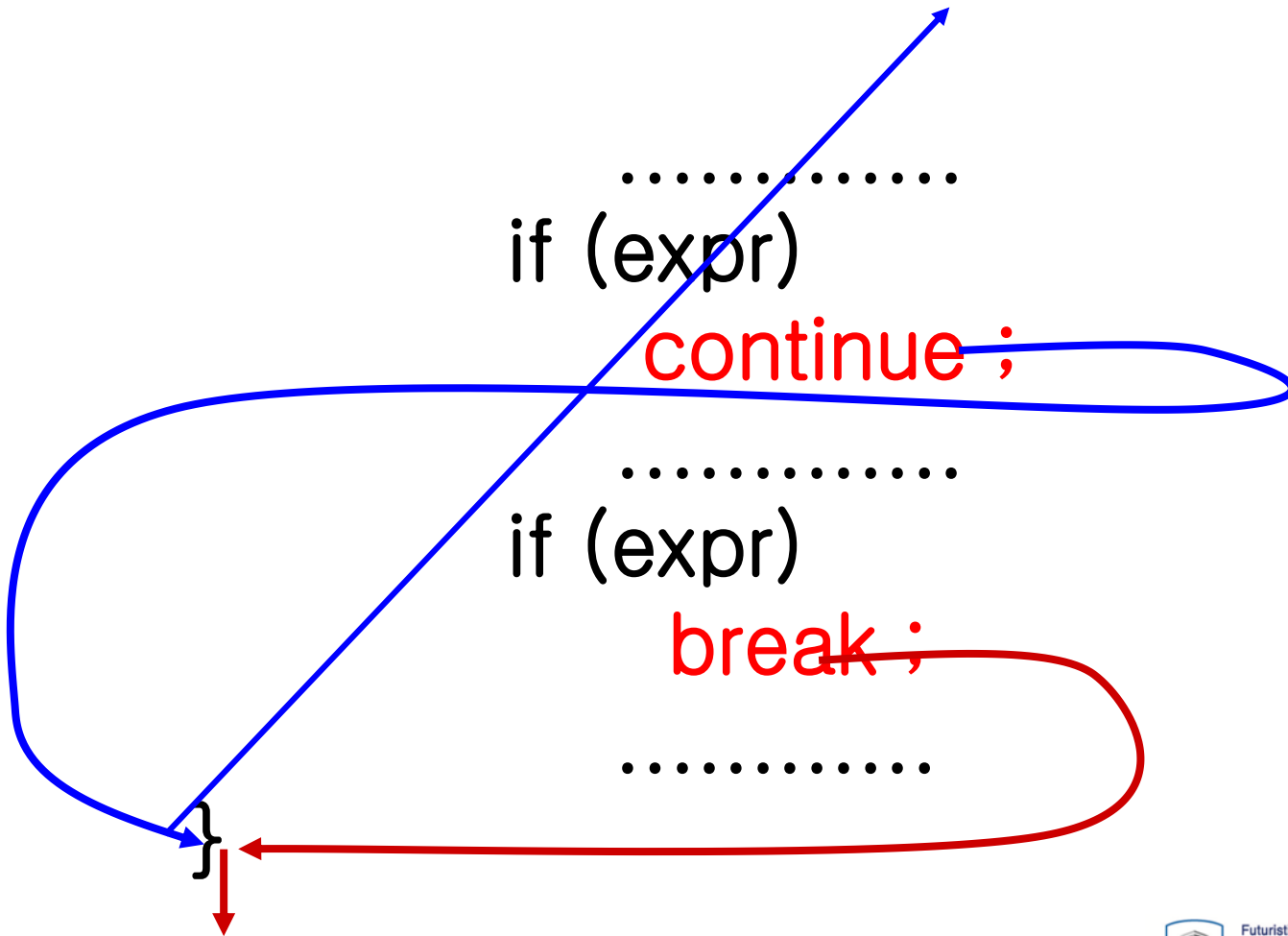
```
.....  
if (expr)
```

```
    continue ;
```

```
.....  
if (expr)
```

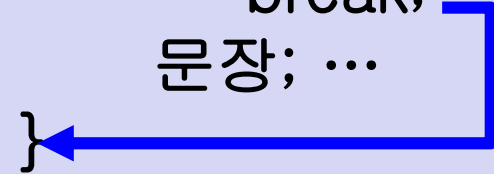
```
    break ;
```

```
.....  
}
```

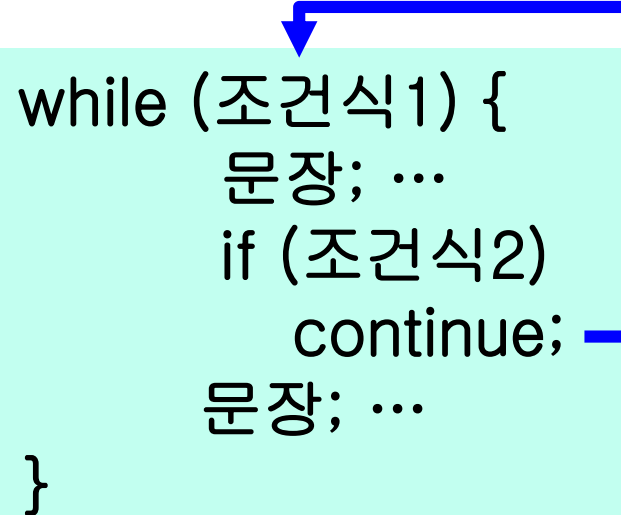


break 문과 continue 문

```
while (조건식1){  
    문장; ...  
    if (조건식2)  
        break; ... 무한루프에서는 반드시 사용  
    문장; ...  
}
```

A blue arrow originates from the 'break;' statement and points to the closing curly brace of the while loop, indicating that the loop is terminated.

```
while (조건식1) {  
    문장; ...  
    if (조건식2)  
        continue;  
    문장; ...  
}
```

A blue arrow originates from the 'continue;' statement and points to the opening curly brace of the while loop, indicating that the loop restarts from the beginning, skipping the rest of the current iteration.

switch문의 break문

- break가 있는 경우와 없는 경우의 프로그램 흐름
(number의 값이 2인 경우)

```
switch (number) {  
    case 1 : statement 1;  
             break;  
    case 2 : statement 2;  
             break;  
    case 3 : statement 3;  
             break;  
    default : statement 4;  
}  
statement 5;
```

A flowchart illustrating the execution of a switch statement with break statements. Red arrows show the flow: from the start of the switch block to case 1, then to case 2, and finally to case 3. From case 3, the flow goes to the default case and then to statement 5, bypassing the rest of the switch block.

```
switch (number){  
    case 1 : statement 1;  
    case 2 : statement 2;  
    case 3 : statement 3;  
    default : statement 4;  
}  
statement 5;
```

A flowchart illustrating the execution of a switch statement without break statements. Red arrows show the flow: from the start of the switch block to case 1, then to case 2, and finally to case 3. From case 3, the flow goes to the default case and then to statement 5, bypassing the rest of the switch block.

라벨로 분기

- JAVA에서는 반복문 앞에 라벨을 붙여서 그 라벨을 break, continue문 뒤에 붙여 중첩된 반복문을 한꺼번에 벗어날 수 있음
 - continue 라벨;
 - 특정 라벨의 다음 반복으로 분기
 - 중첩 반복(nested loop)에서 바깥의 반복문으로 빠져 나갈 때 주로 사용
 - break 라벨;
 - 라벨이 붙은 반복문을 벗어남
 - 중첩 반복문을 한 번에 벗어날 때 주로 사용

break문과 continue문 예제 1

■ 다음 프로그램의 실행 결과는 ?

```
public static void main(String[] args) {  
    for (int count = 1; count <= 10; count++) {  
        System.out.printf(" count = %d\n", count);  
        if (count == 5)  
            break;  
        else if ((count % 2) == 1)  
            continue;  
        System.out.printf(" %d * %d = %d\n", count, count, count * count);  
    }  
}
```

```
count = 1  
count = 2  
2 * 2 = 4  
count = 3  
count = 4  
4 * 4 = 16  
count = 5
```


break문과 continue문 예제 2

■ 다음 프로그램의 실행 결과는 ?

```
public static void main(String[] args) {  
    for (int loop = 0; loop <= 10; loop++) {  
        if (loop == 3)  
            break;  
        for (int count = 0; count < 5; count++) {  
            if (count == 2 || count == 4)  
                continue;  
            System.out.printf("loop = %d count = %d\n", loop, count);  
        }  
    }  
}
```

```
loop = 0 count = 0  
loop = 0 count = 1  
loop = 0 count = 3  
loop = 1 count = 0  
loop = 1 count = 1  
loop = 1 count = 3  
loop = 2 count = 0  
loop = 2 count = 1  
loop = 2 count = 3
```

break문과 continue문 예제 3

- 다음 프로그램의 실행 결과는 ?

```
public static void main(String[] args) {  
    for(int i = 1; i < 10; i++) {  
        for(int j = 1; j < i; j++) {  
            if (j > 6)  
                break;  
            System.out.print(" * ");  
        }  
        System.out.println();  
    }  
}
```

// 내포된 반복문만 벗어난다

```
*  
* *  
* * *  
* * * *  
* * * * *  
* * * * * *  
* * * * * *  
* * * * * *
```

break문과 continue문 예제 4

1. 다음 JAVA 코드의 출력 결과는 ?

```
int count = 12;
while (count > 0) {
    count -= 2;
    if (count == 6)
        break;
    System.out.println(count);
}
```

10
8
6

2. 1번 문제에서 break를 continue로 변경하면 어떻게 되는가?

break문과 continue문 예제 5

■ 다음 프로그램의 실행 결과는 ?

```
public static void main(String[] args) {  
    int i = 0;  
    Label:  
    while (i <= 10) {  
        for (int j = 0; j < 10; j++) {  
            if (j < i)  
                break Label;  
            if (j > i)  
                break;  
            if (j % 2 == 0)  
                continue;  
        }  
        System.out.println("A");  
        i++;  
    }  
    System.out.println("B");  
}
```

A
B

break문과 continue문 예제 5 설명

```
int i = 0;
label:
while (i <= 10){
    for (int j = 0; j < 10; j++){
        if (j < i)
            break label;
        if (j > i)
            break;
        if (j % 2 == 0)
            continue;
    }
    System.out.println("A");
    i++;
}
System.out.println("B");
```

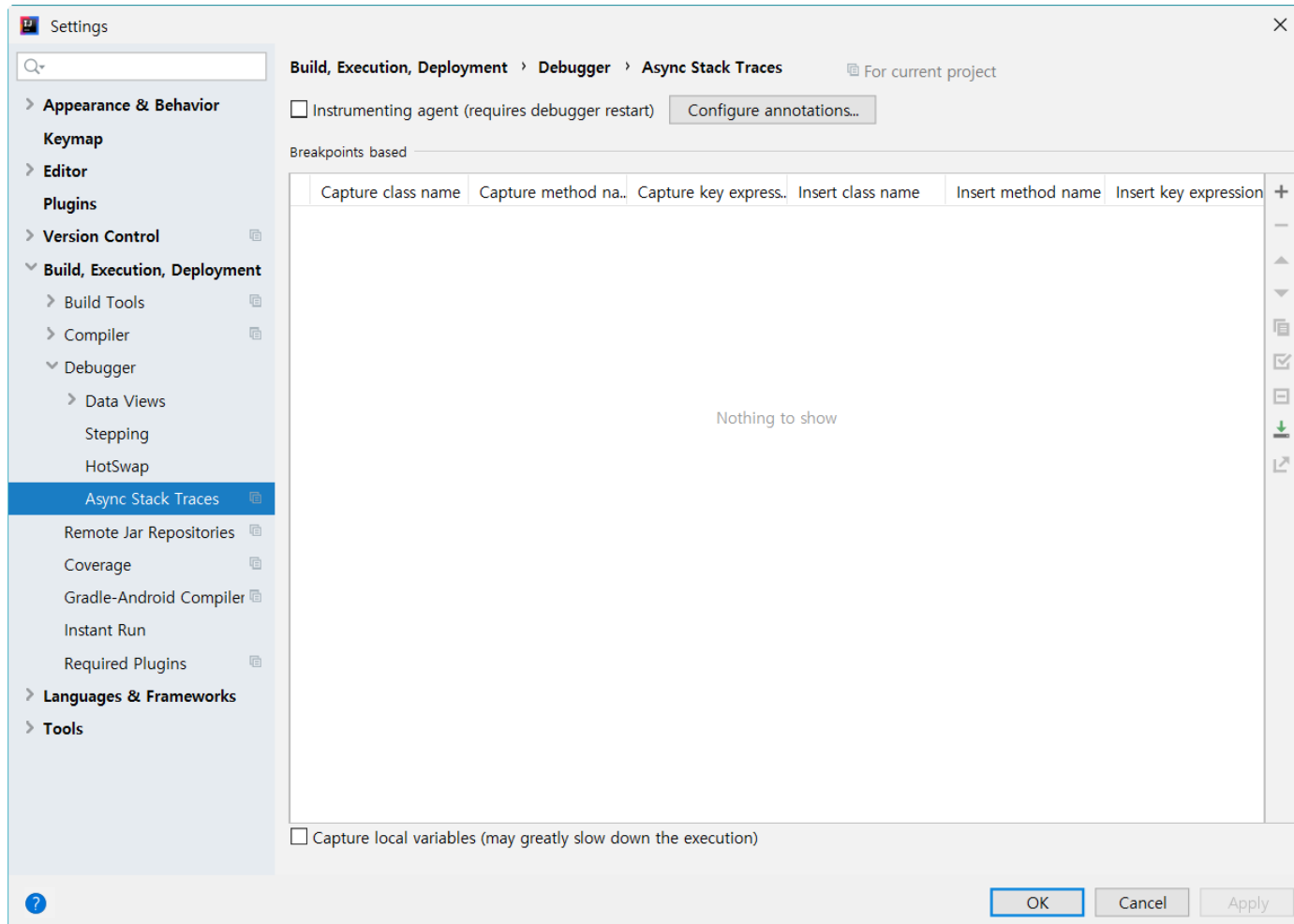
분기문의 조건으로 이동

현재 분기문 밖으로 이동

이름표 붙은 분기문 밖으로 이동

Trace 사용하는 법

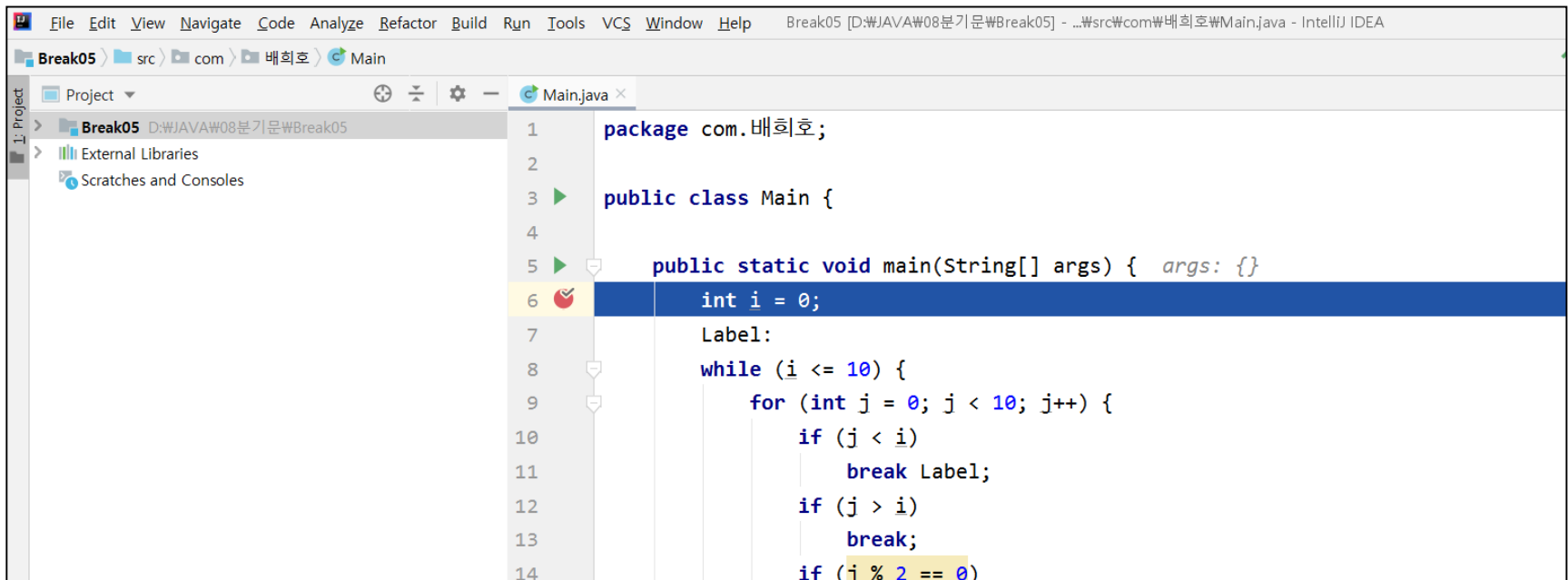
■ 오류 시 setting



Trace 사용하는 법

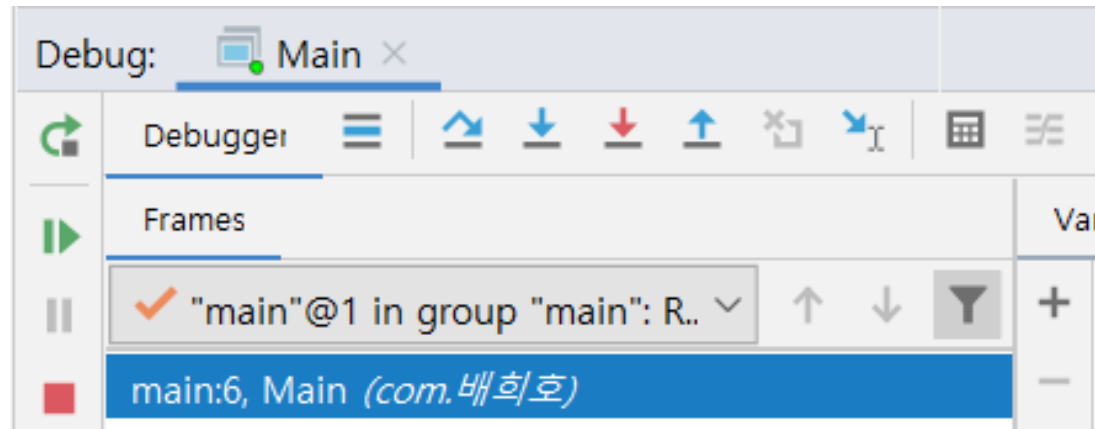
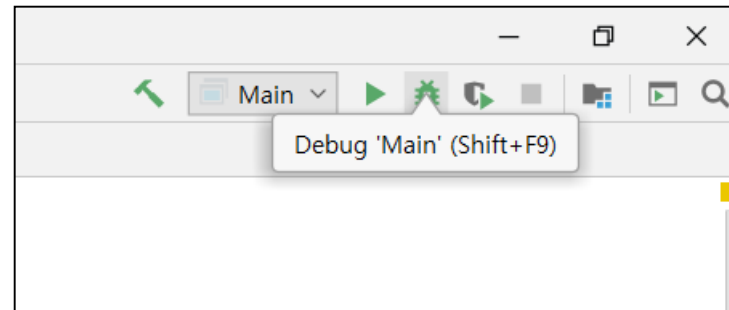
■ Trace 사용하는 법

- 라인 넘버와 코드 사이의 여백을 클릭하면 아래처럼 그 Line에 Break Point가 표시 됨



Trace 사용하는 법

- Trace 사용하는 법
 - Debug 실행



Trace 사용하는 법

■ Debug 실행 창

The screenshot displays the IntelliJ IDEA IDE with a Java project named 'Break05'. The main editor shows the source code of 'Main.java' with the following content:

```
1 package com.배희호;
2
3 public class Main {
4
5     public static void main(String[] args) { args: {}
6         int i = 0;
7         Label:
8         while (i <= 10) {
9             for (int j = 0; j < 10; j++) {
10                 if (j < i)
11                     break Label;
12                 if (j > i)
13                     break;
14                 if (j % 2 == 0)
```

The debugger window at the bottom is active, showing the 'Main' thread. The 'Frames' tab is selected, displaying the current stack frame: 'main:6, Main (com.배희호)'. The 'Variables' tab shows the variable 'args' with the value '(String[0]@808)'. The 'Debugger' toolbar includes buttons for running, stepping, and other debugging actions.

break문과 continue문 예제 6

- while문과 break문을 사용하여 1부터 100까지의 합(5050)을 구하는 프로그램을 작성하여라

```
public static void main(String[] args) {  
    int count = 0, sum = 0;  
  
    while (true) {  
        sum += count;  
        count++;  
        if (count > 100)  
            break;  
    }  
    System.out.println("Sum = " + sum);  
}
```

break문과 continue문 예제 7

- 1부터 100까지 숫자 중 짝수의 합(2550)만 출력하는 프로그램을 for문과 break, continue문을 함께 사용하여 작성하라

```
public static void main(String[] args) {  
    int even = 0;  
    for (int count = 1; ; count++) {  
        if (count % 2 != 0)  
            continue;  
        if (count > 100)  
            break;  
        even += count;  
    }  
    System.out.printf("1부터 100까지 짝수의 합 = %d\n", even);  
}
```

break문과 continue문 예제 8

- 1에서 100까지 정수에서 3의 배수는 제외한 정수의 합 (3367)을 구하는 프로그램을 for문, continue, break문을 사용하여 작성하여라

```
public static void main(String[] args) {  
    int sum = 0;  
  
    for (int count = 1; true; count++) {  
        if (count % 3 == 0)  
            continue;  
        sum += count;  
        if (count >= 100)  
            break;  
    }  
    System.out.printf("1 + 2 + 4 + ... + 100 = %d\n", sum);  
}
```

break문과 continue문 예제 9

- 1에서 100사이의 정수 중에 5와 7의 공배수를 출력하는 프로그램을 for문, continue, break문을 사용하여 작성하여라

```
public static void main(String[] args) {  
    for (int count = 1; ; count++) {  
        if (count % 5 == 0 && count % 7 == 0) {  
            System.out.println("5와 7의 공배수 : " + count);  
            continue;  
        }  
        if (count >= 100)  
            break;  
    }  
}
```

break문과 continue문 예제 10

- 1부터 N까지의 수 중 3의 배수를 제외한 모든 수를 더하는 프로그램을 작성하여라. (3367)

```
public static void main(String[] args) throws IOException {  
    Scanner input = new Scanner(System.in);  
    int number, count;  
    int sum = 0;  
  
    while (true) {  
        System.out.printf("Wn 어디까지 더할까요 ? ");  
        number = input.nextInt();  
        if (number > 0 && number <= 100)  
            break;  
        else {  
            System.err.printf(" ERROR : 입력 오류 !!!");  
            System.in.read();  
        }  
    }  
}
```

break문과 continue문 예제 10

```
for (count = 1; ; count++) {  
    if (count % 3 == 0)  
        continue;  
    if (count > 100)  
        break;  
    System.out.printf("%3d +", count);  
    sum += count;  
}  
System.out.printf("\n\n = %d\n", sum);  
}
```

break문과 continue문 예제 11

- 앞의 프로그램에서 추가로 5의 배수도 제외한 모든 수를 더하도록 프로그램을 수정하여라 (2632)

```
if (count % 3 == 0)  
    continue;
```

```
if (count % 3 == 0 || count % 5 == 0)  
    continue;
```


break문과 continue문 예제 12

- 앞의 프로그램에서 5의 배수와 3의 배수를 제외하지만 5와 3의 공배수인 15의 배수는 제외하지 않고 더하는 프로그램으로 수정하여라 (2947)

```
if (count % 3 == 0 || count % 5 == 0)
    continue;
```



```
if (count % 3 == 0 || count % 5 == 0)
    if (!(count % 3 == 0 && count % 5 == 0))
        continue;
```

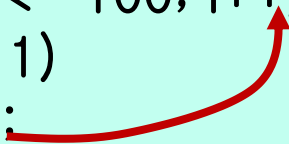
도입 예제

- for와 continue문을 사용하여 1부터 100까지 짝수의 합을 구해보자.

1부터 100까지 짝수의 합은 2,550

도입 예제

```
public static void main (String[] args) {  
    int sum = 0;  
  
    for (int i = 1; i <= 100; i++) {  
        if (i % 2 == 1)  
            continue;  
        else  
            sum += i;  
    }  
  
    System.out.printf("1부터 100까지 짝수의 합은 %,d\n", sum);  
}
```



입력된 숫자 개수 세기

- while문과 break문을 사용하여 -1이 입력될 때까지 입력된 숫자의 개수를 출력하시오.

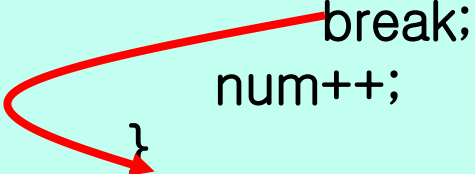
10
8
9
5
-1

마지막 입력을 뜻함

입력된 숫자 개수는 4

입력된 숫자 개수 세기

```
import java.util.Scanner;
public class BreakExample {
    public static void main (String[] args) {
        Scanner in = new Scanner(System.in);
        int num = 0;
        while (true) {
            if (in.nextInt() == -1)
                break;
            num++;
        }
        System.out.println("입력된 숫자 개수는 " + num);
    }
}
```



Prime number

- 어떤 양의 정수를 입력 받아 그것이 소수인지 검사하는 프로그램을 작성하여라.

소수(prime number)

1과 자신의 수만으로 나누어지는 수

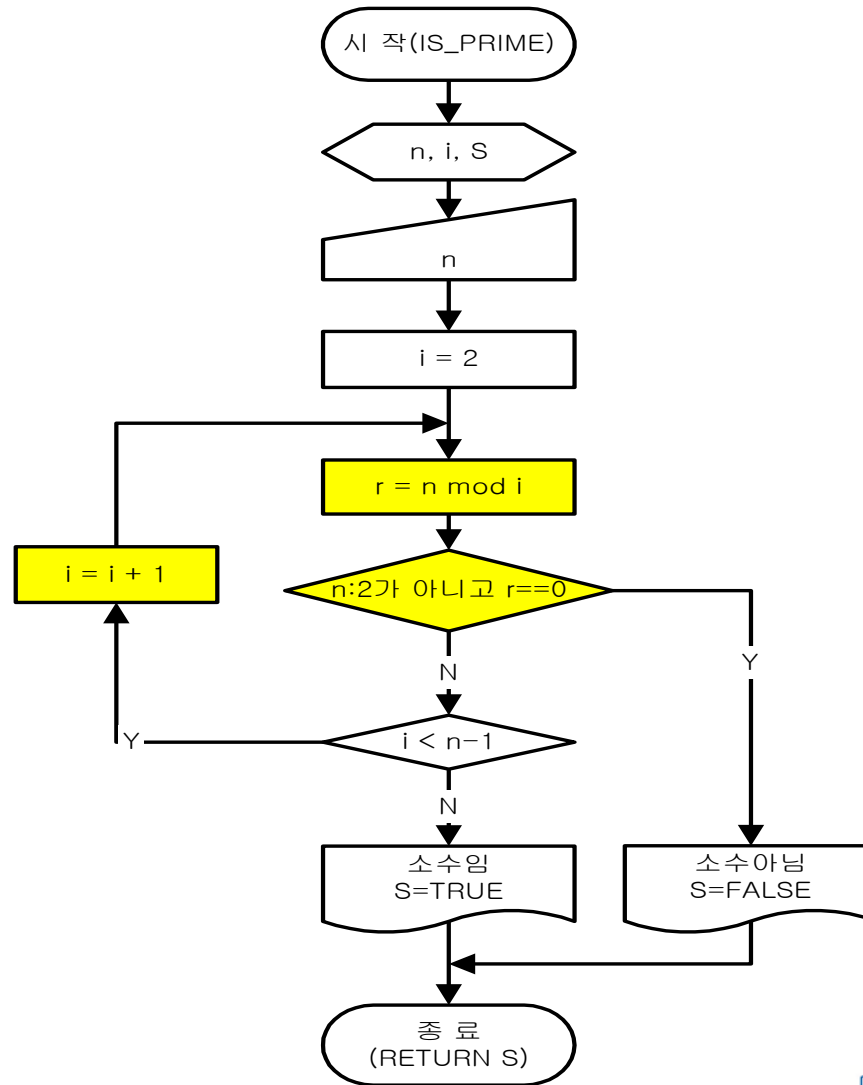
1은 소수도 합성수도 아니다.

1부터 100사이의 소수

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

©doopedia.co.kr

Prime number



Prime number

```
public static void main(String[] args) throws IOException {  
    Scanner keyboard = new Scanner(System.in);  
    int last;  
    int count = 0;  
    int prime;  
  
    while (true) {  
        System.out.printf("\n 10이상의 양의 자연수 입력 : ");  
        last = keyboard.nextInt( );  
        if (last != 1 && last > 0)  
            break;  
        else {  
            System.out.printf("\n ERROR : 입력 오류 !!!");  
            System.in.read( );  
        }  
    }  
}
```


Prime number

```
for (int number = 2; number <= last; number++) {  
    for (prime = 2; count < number; prime++) {  
        if (number % prime == 0)  
            break;  
    }  
    if (prime == number) {  
        ++count;  
        System.out.printf(" %3d,", number);  
    }  
}  
System.out.printf("\n\nTotal : %d\n", count);  
}
```

Armstrong number

- 1 ~ 1,000사이에 있는 수에 대하여 각 자리 수에 세제곱의 합이 그 자신의 수와 같아지는 수(암스트롱 수)를 구하여라.
- 예) $371 = 3^3 (= 27) + 7^3 (= 343) + 1^3 (= 1)$

1
153
370
371
407

Armstrong number

```
public static void main(String[] args) {  
    final int RANGE = 1000;  
    int number, remainder;  
    int total;  
  
    for (int i = 1; i <= RANGE; i++) {  
        total = 0;  
        number = i;  
        while (true) {  
            if (number == 0)  
                break;  
            remainder = number % 10;  
            total += remainder * remainder * remainder;  
            number /= 10;  
        }  
        if (i == total)  
            System.out.printf("%8d\n", total);  
    }  
}
```

Numerical Center

- 1000까지의 수 중에서 숫자 중심수를 구하여라.
- Numerical Center(숫자 중심)
 - 임의의 어떤 수를 중심으로 하여 그 수의 앞까지 합계와 그 수 뒤의 합계가 같은 경우 그 수를 숫자 중심이라고 한다.

Numerical Center

■ 문제 분석

■ 자연수에서 6의 경우

6을 중심으로 앞의 숫자의 합계

$$1 + 2 + 3 + 4 + 5 = 15 \text{이고,}$$

뒤의 숫자를 더해 가다 보면

$$7 + 8 = 15 \text{로 일치하는 경우}$$

우리는 6을 숫자 중심이라고 한다.

Numerical Center

```
public static void main(String[] args) {  
    final int RANGE = 1000;  
    int before, after;  
  
    for (int i = 2; i <= RANGE; i++) {  
        after = before = 0;  
        for (int j = 1; j <= i - 1; j++)  
            before += j;  
        for (int j = i + 1; j <= RANGE; j++) {  
            after += j;  
            if (after > before)  
                break;  
            else if (after == before) {  
                System.out.printf(" 숫자중심 = %3d,", i);  
                System.out.printf(" TOTAL = %,6d\n", before);  
            }  
        }  
    }  
}
```

짝수/홀수 판별하기

```
public static void main(String[] args) {  
    Scanner keyboard = new Scanner(System.in);  
    int test = 0;  
    boolean flag;  
    String result = "";  
  
    try {  
        System.out.print(" 정수 입력 : ");  
        test = keyboard.nextInt();  
        flag = true;  
    } catch (InputMismatchException e) {  
        flag = false;  
        result = " 입력 오류 입니다.";  
    }  
}
```

짝수/홀수 판별하기

```
if (flag) {  
    switch (test % 2) {  
        case 0:  
            result = String.format(" %d는(은) 짝수 입니다.", test);  
            break;  
        case 1:  
            result = String.format(" %d는(은) 홀수 입니다.", test);  
        }  
    }  
  
    System.out.println(result);  
}
```


수열 더하기

- 1부터 100까지의 합에서 5의 배수를 제외한 합계는 ?

수열 더하기

```
public static void main(String[] args) {  
    final int BAESU = 5;  
    int temp = 0;  
    int sum = 0;  
  
    for (int count = 1; count <= 100; count++) {  
        if ((count % BAESU) == 0) {  
            temp += count;  
            continue;  
        } else  
            sum += count;  
    }  
    System.out.printf("%d 1 과 100 사이에서 %d의 배수의 합 = %d\n",  
                      BAESU, temp);  
    System.out.printf(" 1 과 100 사이에서 %d의 배수를 제외한 합 = %d\n",  
                      BAESU, sum);  
}
```

수열 더하기[심화]

- 1에서 100사이의 값을 입력 받아 3의 배수와 5의 배수는 제외하고, 15의 배수는 제외하지 않으면서 합을 구해보자.

수열 더하기[심화]

```
public static void main(String[] args) throws IOException {
    Scanner keyboard = new Scanner(System.in);
    int number;
    int count;
    long sum = 0L;

    while (true) {
        System.out.print("\n 어디까지 더할까요 ? ");
        number = keyboard.nextInt();
        if (number > 0 && number <= 100)
            break;
        else {
            System.out.printf("ERROR : 입력 오류 !!!");
            System.in.read();
        }
    }
}
```

수열 더하기[심화]

```
for (count = 1; count <= number; count++) {  
    if (count % 3 == 0 || count % 5 == 0) {  
        if (!(count % 3 == 0 && count % 5 == 0))  
            continue;  
    }  
    if (count > 100)  
        break;  
    System.out.printf("%,3d +", count);  
    sum += count;  
}  
System.out.printf("%bWb = %,dWn", sum);  
}
```

나이 계산

- 현재 어머니의 나이는 41세이고, 아들의 나이는 11세이다.
어머니 나이가 아들 나이의 3배가 되는 때는 몇 년 후인가?

현재 어머니 나이 41세, 아들 나이 11 살 입니다
4년 후 어머니 나이 45세, 아들 나이 15 살 입니다

나이 계산

```
public static void main(String[] args) {  
    int mother = 41;  
    int son = 11;  
  
    System.out.printf("현재 어머니 나이 %d세, 아들 나이 %d 살 입니다\n",  
                      mother, son);  
  
    int year = 1;  
    while (true) {  
        mother++;  
        son++;  
        if (mother == son * 3)  
            break;  
        year++;  
    }  
    System.out.printf("%d년 후 어머니 나이 %d세, 아들 나이 %d 살 입니다\n",  
                      year, mother, son);  
}
```

저축하기

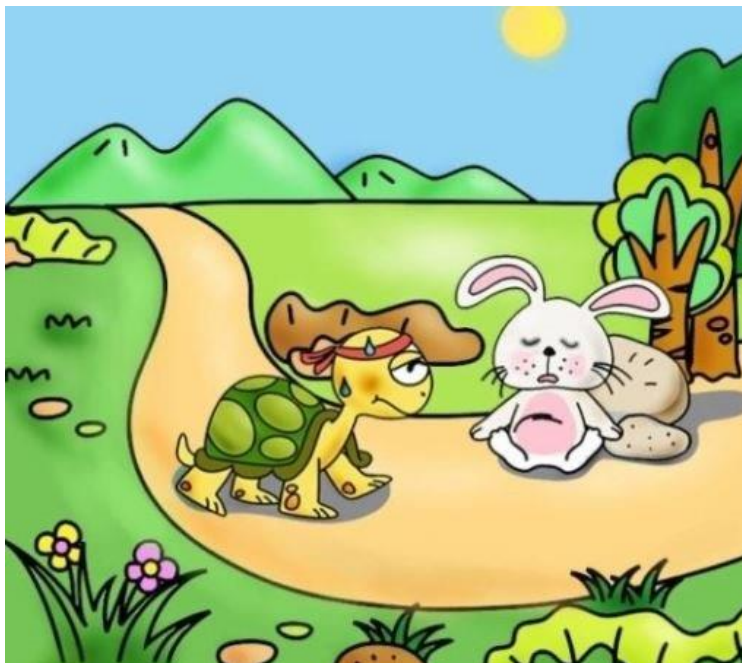
- 어떤 사람이 1억원을 은행에 년 13%의 복리 이자로 예금하면 몇 년 후에 원리금이 초기 원금의 2배가 넘게되는 지 계산하여라.
- 복리 이자 = 원금 * (1 + 이율)^{기간}
- 단리 이자 = 원금 * (1 + 이율 * 기간)

저축하기

```
public static void main(String[] args) {  
    final int MONEY = 100000000;  
    final float RATE = 13.0f / 100;  
    int deposit = MONEY;  
  
    int year = 0;  
    do {  
        deposit = (int) (deposit * (1 + RATE));  
        year++;  
        if (deposit >= MONEY * 2)  
            break;  
        System.out.printf("%2d년 후 원리금 = %,d 원\n", year, deposit);  
    } while (true);  
    System.out.printf("%2d년 후 원리금 = %,d 원\n", year, deposit);  
}
```

토끼와 거북이

- 토끼와 거북이가 달리기를 하는데 토끼의 속도는 시속 60Km/h로 달리고, 거북이는 시속 35Km/h로 달린다
- 그리고 거북이는 토끼보다 450m 앞에서 출발한다
- 토끼가 거북이를 앞지르는 것은 출발 후 몇 초 후부터 인가 ?



토끼와 거북이

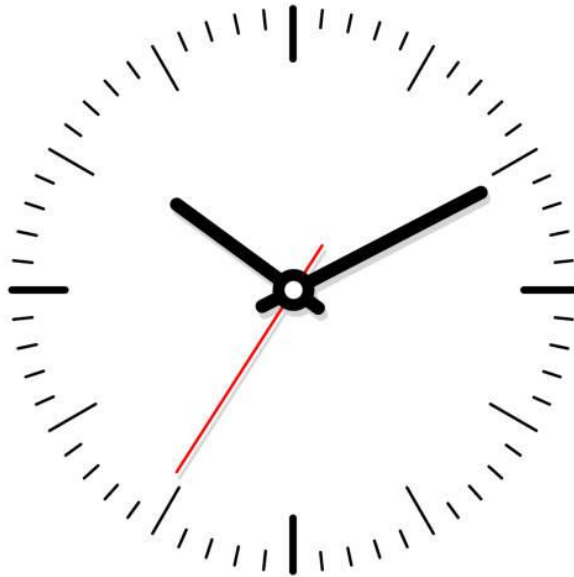
```
public static void main(String[] args) {  
    final int MINUTE = 60;  
    final int SECOND = 60;  
    final int KILO = 1000;  
    final float RABIT = 60.0f * KILO / (MINUTE * SECOND) ;  
    final float TURTLE = 35.0f * KILO / (MINUTE * SECOND);  
    float rdis = 0.0f, tdis = 450.0f;    /* 거리 */  
    int sec;  
  
    for (sec = 1; true; ++sec) {  
        rdis += RABIT;  
        tdis += TURTLE;  
        System.out.printf(" %3d초, 토끼 = %,8.2f m, 거북이 = %,8.2f m\n",  
                           sec, rdis, tdis);  
  
        if (rdis > tdis)  
            break;  
    }  
}
```

토끼와 거북이

```
System.out.printf("\n second = %d 초", sec);  
System.out.printf("\n 토끼가 달린 거리 = %,.2f m", rdis);  
System.out.printf("\n 거북이가 달린 거리는 = %,.2f m \n", tdis);  
}
```

시계 바늘

- 아날로그 시계에서 시침, 분침이 일치하는 시간을 알아보는 Program을 작성하여라.



시계 바늘

- 분침은 60분 동안 한 바퀴(360도)를 회전하므로, 분침이 1분 동안 회전하는 각은 6도임
- 시침은 한 시간(60분) 동안 30도를 움직이므로(시계 눈금은 12개이고 360도를 12로 나누면 30도) 시침은 1분 동안 0.5도($30\text{도} \div 60\text{분}$) 회전
- 초침은 1분에 360도 회전

시계 바늘

- 0시 정각과 12시 정각에 시침과 분침과 초침이 일치
- 그 사이에 0시 정각과 12시 정각을 포함하면, 12번 시침과 분침이 일치
- 12개의 지점으로 나누어지는 11개의 구간은 그 길이가 같다. 왜냐하면, 시계의 숫자와 눈금을 모두 지우면, 하나의 시침, 분침의 일치 지점과 그 다음 번의 시침, 분침 일치 지점 간의 상대적 위치는 모두 동등하기 때문임
- 따라서, 시침만을 생각했을 때, 시침이 0시 정각부터 12시 정각까지 오는 한 바퀴를 11로 동등하게 나누는 지점들이 시침과 분침이 일치하는 지점
- 즉, 0시 정각 이후 첫번째 일치 지점의 시각은 $0\text{시 정각} + 12\text{시간}/11$
- 그 다음은 $0\text{시 정각} + 2 \times (12\text{시간}/11)$

시계 바늘

```
public static void main(String[] args) {  
    double temp;  
    int minute, second;  
  
    for (int time = 0; time < 12; time++) {  
        temp = 60.0 / 11.0 * time;  
        minute = (int) temp;  
        if (minute >= 60)  
            continue;  
        second = (int) ((temp - minute) * 60);  
        System.out.printf("%2d 시 %2d 분 %2d 초", time, minute, second);  
    }  
}
```


아기의 생일

- 2017년 5월 6일은 일요일이다. 이날 태어난 아기의 10번째 생일은 무슨 요일인가?