

3장. 제어문(조건문, 반복문)



if / loop



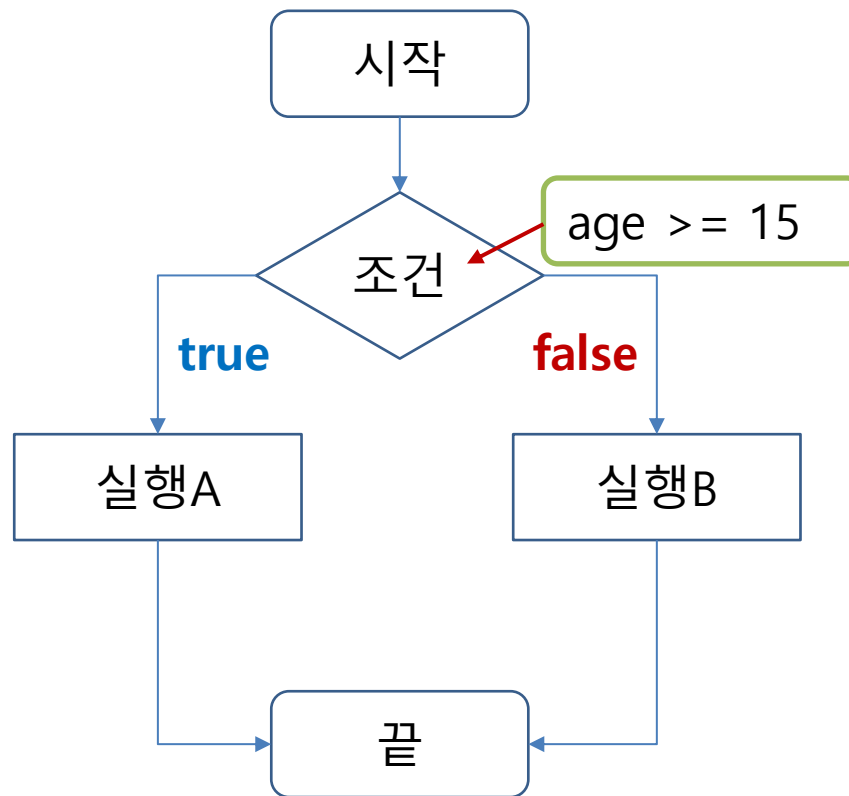
조건문(if문)

❖ 조건문

- 주어진 조건에 따라 다른 수행문이 실행되도록 한 프로그래밍 구문
- if문, switch 문이 대표적이다.

▪ if문

```
if(조건식){  
    수행문;  
}  
...
```



조건문(if문)

- if문

```
if(조건식){  
    조건식이 true 이면 실행  
}
```

- if-else 문

```
if(조건식){  
    조건식이 true 이면 실행  
}  
else{  
    조건식이 false 이면 실행  
}
```



조건문(if문)

```
public class LimitSpeed {  
    public static void main(String[] args) {  
        // 제한 속도  
        int limitSpeed = 55;  
  
        if(limitSpeed >= 50) {  
            System.out.println("제한 속도 위반!! 과태료 10만원 부과 대상");  
        }else {  
            System.out.println("안전 속도 준수!!");  
        }  
        System.out.println("시속 " + limitSpeed + "km입니다.");  
    }  
}
```



조건문(if문)

■ if ~ else if ~ else문

```
if(조건식){  
    조건식이 true이면 실행  
}  
else if(조건식2){  
    조건식2가 true이면 실행  
}  
else{  
    조건식1,2가 모두 false이면  
    실행  
}
```

중,고등학생입니다.
입장료는 2500원입니다.

대 상	입장료
취학전 아동	1,000원
초등학생	2,000원
중.고등학생	2,500원
일반인	3,000원



조건문(if문)

```
public class AdmissionFee {  
  
    public static void main(String[] args) {  
        int age = 9;  
        int fee;    //입장료(요금)  
  
        if(age < 8) {  
            fee = 1000;  
            System.out.println("취학 전 아동입니다.");  
        }  
        else if(age < 14) {  
            fee = 2000;  
            System.out.println("초등학생입니다.");  
        }  
        else if(age < 20) {  
            fee = 2500;  
            System.out.println("중, 고등학생입니다.");  
        }  
        else {  
            fee = 3000;  
            System.out.println("일반인입니다.");  
        }  
        //System.out.println("입장료는 " + fee + "원입니다.");  
        System.out.printf("입장료는 %,d원입니다.", fee);  
    }  
}
```



조건문(if문)

■ 조건문과 조건 연산자

- 간단한 if ~ else 조건문은 조건 연산자로 구현할 수 있음
- 두 수 중 큰 값을 출력하는 프로그램

```
if(a > b) {  
    max = a;  
}  
else {  
    max = b;  
}
```



```
max = (a > b) ? a : b;
```

If~else 문

조건 연산자



조건문(if문)

■ 조건문과 조건 연산자

```
// 두 수 중 큰수 비교
Scanner sc = new Scanner(System.in);
System.out.print("첫번째 수 입력 : ");
int n1 = sc.nextInt();
System.out.print("두번째 수 입력 : ");
int n2 = sc.nextInt();

int max;
max = n1 > n2 ? n1 : n2; //조건 연산자 구문

/*
if(n1 > n2) {
    max = n1;
}
else {
    max = n2;
}
*/

System.out.println("두 수 중 큰 수는 " + max + "입니다.");
```



if문 – 연습 예제

입장객 수에 따른 좌석 줄 수를 계산하는 프로그램을 작성하세요.
(파일이름:seat.java)

👉 실행 결과

```
입장객 수 입력:  
23  
좌석 열의 수 입력:  
5  
5개의 줄이 필요합니다.
```



윤년을 계산하는 프로그램


조건 : 윤년은 4년마다 오며 100년 단위는 윤년이 아니나, 400년 단위로 윤년이다.
(파일 이름 : LeapYear.java)

```
Scanner sc = new Scanner(System.in);

System.out.println("년도를 입력하세요: ");
int year = sc.nextInt();

if(year % 4 == 0 && year % 400 == 0 || year % 100 != 0) {
    System.out.println(year + "년은 윤년입니다.");
}
else {
    System.out.println(year + "년은 윤년이 아닙니다.");
}

sc.close();
```

윤년 (閏年) 

[명사] [천문] 윤달이나 윤일이 든 해. 지구가 태양을 한 번 공전하는 데에 365일 5시간 48분 46초 걸리므로 태양력에서는 그 나머지 시간을 모아 4년마다 한 번 2월을 하루 늘리고, 태음력에서는 1년을 354일로 정하므로 계절과 역...



조건문(SWITCH – CASE)

▪ switch ~ case문

조건식의 결과가 정수 또는 문자열의 값이고 그 값에 따라 수행문이 결정될때

```
//순위에 따른 메달 색깔 출력하기
int rank = 2;
char medalColor;

switch(rank) {
case 1:
    medalColor = 'G';
    break;
case 2:
    medalColor = 'S';
    break;
case 3:
    medalColor = 'B';
    break;
default:
    medalColor = 'A';
    break;
}
System.out.println("메달 색깔은 " + medalColor + "입니다.");
```



조건문(SWITCH – CASE)

▪ switch ~ case문에 문자열 사용하기

```
public class medalColor {  
    public static void main(String[] args) {  
        String medalColor = "Gold";  
        switch(medalColor){  
            case "Gold":  
                System.out.println("금메달 입니다.");  
                break;  
            case "Silver":  
                System.out.println("은메달 입니다.");  
                break;  
            case "Bronze":  
                System.out.println("동메달 입니다.");  
                break;  
            default:  
                System.out.println("메달이 없습니다.");  
                break;  
        }  
    }  
}
```



조건문(SWITCH – CASE)

▪ case문 동시에 사용하기

```
// case문 동시 사용
int month = 6;
int day = 0;

switch(month) {
case 1: case 3: case 5: case 7: case 8: case 10: case 12:
    day = 31;
    break;
case 4: case 6: case 9: case 11:
    day = 30;
    break;
case 2:
    day = 28;
    break;
default:
    System.out.println("지원되지 않는 기능입니다.");
    return; //즉시 종료
}
System.out.println(month + "월은 " + day + "일까지 있습니다.");
```



실습 예제

- operator 값이 +, -, *, / 인 경우에 사칙연산을 수행하는 프로그램
if문과 switch문을 사용해 작성하기

```
int num1 = 10;  
int num2 = 2;  
char operator = '+';
```

☞ 실행 결과

결과는 12입니다.



실습 예제

```
int num1 = 10;
int num2 = 2;
char operator = '+';
int result = 0;

if(operator=='+') {
    result = num1 + num2;
}else if(operator=='-') {
    result = num1 - num2;
}else if(operator=='*') {
    result = num1 * num2;
}else if(operator=='/') {
    result = num1 / num2;
}else {
    System.out.println("연산자 오류입니다.");
    return;
}

System.out.println("결과 값은 " + result + "입니다.");
```

```
int num1 = 10;
int num2 = 2;
char operator = '+';
int result = 0;

switch(operator) {
    case '+':
        result = num1 + num2;
        break;
    case '-':
        result = num1 - num2;
        break;
    case '*':
        result = num1 * num2;
        break;
    case '/':
        result = num1 / num2;
        break;
    default:
        System.out.println("연산자 오류입니다.");
        return;
}

System.out.println("결과 값은 " + result + "입니다.");
```



반복문(while문)

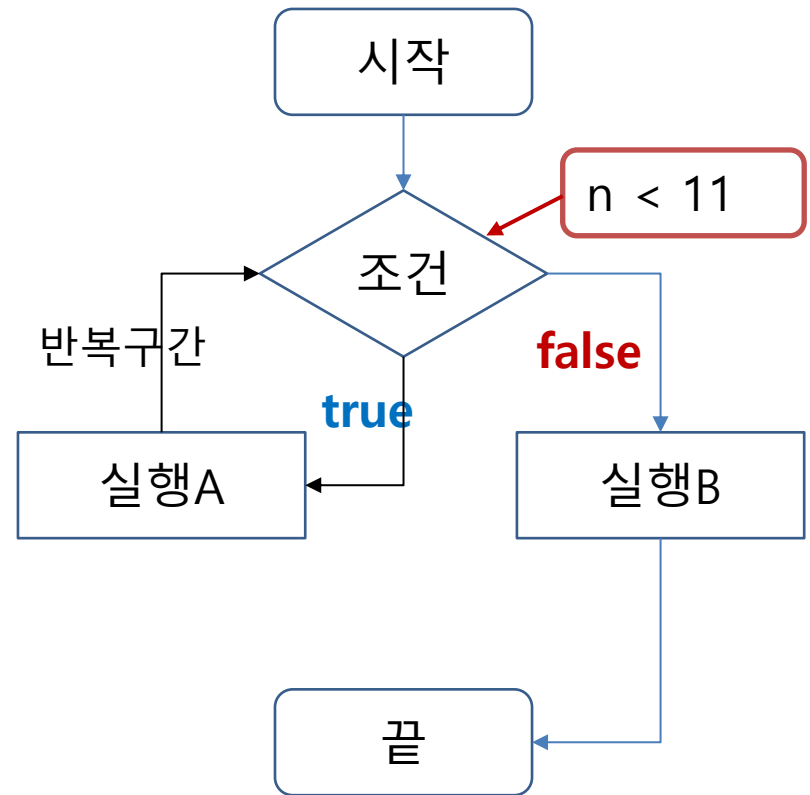
● 반복문

- 주어진 조건이 만족할 때까지 수행문을 반복적으로 수행함
- while, for 문이 있음

● while 문

- 조건식이 참인 동안 반복 수행

```
while(조건식){  
    수행문1;  
}  
  
수행문2;  
  
...
```



반복문(while문)

- "Hello"를 10번 출력하는 프로그램

```
int n = 1; //초기값

while(n <= 10) { //종료값
    System.out.println("Hello" + n);
    n++; //증감값
}
```

```
Hello~ 1
Hello~ 2
Hello~ 3
Hello~ 4
Hello~ 5
Hello~ 6
Hello~ 7
Hello~ 8
Hello~ 9
Hello~ 10
```



반복문(while문)

- 1부터 10까지 합계를 계산하는 프로그램

```
int num = 1;  
num += 2;  
num += 3;  
...  
num += 9;  
num += 10;  
  
System.out.println("합계  
는" + num + "입니다.");
```

vs

```
int n = 1;  
int sum = 0;  
  
while(n <= 10) {  
    sum += n;  
    n++;  
}  
  
System.out.println("합계  
는" + sum + "입니다.");
```



반복문(while문)

```
public class While1to10 {  
    public static void main(String[] args) {  
        //1~4까지 더하기  
        int num = 1;  
        num += 2;  
        num += 3;  
        num += 4;  
        System.out.println("num = " + num);  
  
        //1부터 10까지의 합계  
        int n = 0;    //반복 변수  
        int sum = 0;  //합계  
  
        while(n < 10) {  
            n++;  
            sum += n; //sum = sum + n  
            System.out.println("n = " + n + ", sum = " + sum);  
        }  
  
        System.out.println("1부터 10까지의 합은 " + sum + "입니다.");  
    }  
}
```



무한반복 - break 문

■ 무한 반복문 - break 문

반복문에서 break 문을 만나면 더 이상 반복을 수행하지 않고 반복문 빠져 나옴

```
while(true){  
    if(조건식){  
        break;  
    }  
}
```

```
public class WhileIfBreak {  
  
    public static void main(String[] args) {  
        //반복 조건문  
        int i = 0;  
        while(true) {  
            i++;  
            System.out.println("Hello " + i);  
            if(i == 10)  
                break;  
        }  
        System.out.println("=====");  
  
        // 1부터 10까지 더하기  
        int n = 0;  
        int sumV = 0;  
        while(true) {  
            n++;  
            sumV += n;  
            if(n == 10)  
                break;  
        }  
        System.out.println("합계 : " + sumV);  
    }  
}
```



while문 – 실습 예제

조건 : 'y' 키를 누르면 "계속 반복", 'n'키를 누르면 "반복 중단",
이외의 키를 누르면 "지원하지 않는 키" 프로그램을 작성하세요.

문자열이 같은 지를 비교하는 함수(메서드)는 equals() 임

👉 실행 결과

```
계속 반복할까요?(y/n) : y  
계속 반복합니다.  
계속 반복할까요?(y/n) : k  
지원하지 않는 키입니다.  
계속 반복할까요?(y/n) : n  
반복을 중단합니다.  
프로그램을 종료합니다.
```



while문 – 실습 예제

```
public class KeyRepeat {  
    public static void main(String[] args) {  
        //y - "계속 반복합니다", n - "반복을 중단합니다."  
        //그 이외의 키는 "지원하지 않는 키입니다."  
        Scanner sc = new Scanner(System.in);  
  
        while(true) {  
            System.out.print("계속 반복할까요?(y/n) : ");  
  
            String key = sc.nextLine();  
  
            if(key.equals("y") || key.equals("Y")) { //equals() 문자열 비교함수  
                System.out.println("계속 반복합니다.");  
            }  
            else if(key.equals("n") || key.equals("N")) {  
                System.out.println("반복을 중단합니다.");  
                break;  
            }  
            else {  
                System.out.println("지원하지 않는 키입니다.");  
            }  
        }  
        System.out.println("프로그램을 종료합니다.");  
        sc.close();  
    }  
}
```



커피 자동판매기

■ 실습 예제 – 커피자판기 프로그램

돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 4개입니다.
돈을 넣어주세요: 600
거스름돈 100원을 돌려주고 커피가 나옵니다.
남은 커피의 양은 3개입니다.
돈을 넣어주세요: 400
돈을 돌려주고 커피는 나오지 않습니다.
남은 커피의 양은 3개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 2개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 1개입니다.
돈을 넣어주세요: 500
커피가 나옵니다.
남은 커피의 양은 0개입니다.
커피가 다 떨어졌습니다. 판매를 중지 합니다.

- 커피의 총 개수 : 5
- 커피 값은 500원 -> "커피가 나옵니다" 출력
하고, 커피 1개 감소
- 500원 보다 크면 -> "거스름돈이 나옵니다." 출력
하고, 커피 1개 감소
- 500원 보다 작으면 -> "커피가 나오지 않습니다."
- 커피가 모두 소진되면 "판매를 중단합니다" 출력하
고, 프로그램 종료



커피 자동판매기

```
public class CoffeeMachine {
    public static void main(String[] args) {
        // 커피 자동판매기
        Scanner sc = new Scanner(System.in);

        int coffee = 5;    //커피 총 수량
        while(true) {
            System.out.print("돈을 넣어주세요: ");
            int money = sc.nextInt();
            if(money == 500) {
                System.out.println("커피가 나옵니다.");
                coffee -= 1;
            }
            else if(money > 500) {
                System.out.println("거스름돈 " + (money - 500) + "원을 돌려주고 커피가 나옵니다.");
                coffee -= 1;
            }
            else {
                System.out.println("돈을 돌려주고 커피는 나오지 않습니다.");
            }
            System.out.println("남은 커피의 양은 " + coffee + "개입니다.");

            if(coffee == 0) {
                System.out.println("커피가 다 떨어졌습니다. 판매를 중지 합니다.");
                break;
            }
        }
        sc.close();
    }
}
```



은행 업무 프로그램

아래의 실행 결과대로 은행 업무를 구현하는 프로그램을 작성하세요.
(switch~case문으로 각각 구현해 보세요)

1. 예금 | 2. 출금 | 3. 잔고 | 4. 종료

선택> 1

예금액> 10000

1. 예금 | 2. 출금 | 3. 잔고 | 4. 종료

선택> 2

출금액> 3000

1. 예금 | 2. 출금 | 3. 잔고 | 4. 종료

선택> 3

잔고> 7000

1. 예금 | 2. 출금 | 3. 잔고 | 4. 종료

선택> 4

프로그램 종료



은행 업무 프로그램

■ BankSwitch.java

```
public class BankingSwitch {  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        boolean run = true;  
        int balance = 0;  
  
        while(run) {  
            System.out.println("=====");  
            System.out.println("1.예금 | 2.출금 | 3.잔고 | 4.종료");  
            System.out.println("=====");  
            System.out.print("선택>");  
  
            int selNum = sc.nextInt();  
            switch(selNum) {  
                case 1:  
                    System.out.print("예금액>");  
                    balance += sc.nextInt();  
                    break;  
                case 2:  
                    System.out.print("출금액>");  
                    balance -= sc.nextInt();  
                    break;  
                case 3:  
                    System.out.println("잔고>" + balance);  
                    break;  
                case 4:  
                    run = false;  
                    break;  
                default:  
                    System.out.println("메뉴를 잘못 누르셨습니다. 다시 입력해 주세요");  
                    break;  
            }  
        }  
        System.out.println("프로그램 종료.");  
        sc.close();  
    }  
}
```



은행 업무 프로그램

- 파일 이름 : Bank.java

```
Scanner sc = new Scanner(System.in);
int balance = 0;
boolean run = true;

System.out.println("-----");
System.out.println("1.입금 | 2.출금 | 3.잔고 | 4.종료");
System.out.println("-----");
while(run) {
    System.out.print("선택> ");
    int selNum = sc.nextInt();
    if(selNum==1) {
        System.out.println("입금액> ");
        balance += sc.nextInt();
    }
    else if(selNum==2) {
        System.out.println("출금액> ");
        balance -= sc.nextInt();
    }
    else if(selNum==3) {
        System.out.println("잔고> ");
        System.out.println(balance);
    }
    else if(selNum==4) {
        run = false;
    }
}
sc.close();
System.out.println("프로그램 종료!!");
```



개선사항

- ① 잔액이 부족한 경우 처리
- ② 수행후 "정상 처리"와 돈에 천 단위 구분기호 넣기



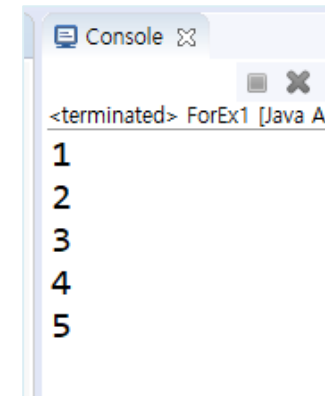
반복문(for문)

▪ for 문

- 주로 조건이 횟수인 경우에 사용
- 초기화식, 조건식, 증감식을 한꺼번에 작성

```
for(초기화식; 조건식; 증감식){  
    수행문;  
}
```

```
int n;  
    ① → ② ← ④  
for(n = 1; n <= 5; n++) {  
    ③ ↗  
    System.out.println(n);  
}
```



The screenshot shows a console window titled "Console" with a close button. The output text is "<terminated> ForEx1 [Java Ap" followed by the numbers 1, 2, 3, 4, and 5, each on a new line.



반복문(for문)

▪ for문

"Hello"를 10번 반복하기

```
int n = 1; //초기값  
  
for(n=1; n <= 10; n++) {  
    System.out.println("hello");  
}
```

1부터 10까지 합계

```
int n;  
int sum;  
for(n=1, sum = 0; n <=10; n++) {  
    sum += n;  
}  
System.out.println(sum);
```



문자 세트 반복

- 문자 세트(유니코드)
 - 알파벳과 한글 자음, 모음 출력하기

```
char ch;
for(ch=65; ch<123; ch++) {
    System.out.print(ch + " ");
}

System.out.println();

for(ch=12593; ch<12686; ch++) {
    System.out.print(ch + " ");
}
```

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\] ^ _ ` a b c d e f g h i j k l m n o p q r s t u
ㄱ ㅋ ㆁ ㄴ ㄷ ㄹ ㅁ ㅂ ㅅ ㅇ ㅈ ㅊ ㅌ ㅍ ㅎ ㅣ ㅡ ㅑ ㅓ ㅕ ㅗ ㅛ ㅜ ㅠ



■ 구구단 프로그램

6x1=6
6x2=12
6x3=18
6x4=24
6x5=30
6x6=36
6x7=42
6x8=48
6x9=54

```
//구구단
int dan = 6;
int i;
for(i=1; i<10; i++) {
    System.out.println(dan + "x" + i + "=" + (dan*i));
}
```



반복문(중첩 for문)

■ 중첩된 반복문(Nested Loop)

반복문 내부에 또 반복문이 사용됨

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

```
int i, j;  
for(i=1; i<=5; i++) {  
    for(j=1; j<=5; j++) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

```
가가가가가  
가가가가가  
가가가가가  
가가가가가  
가가가가가
```

	열1	열2	열3	열4	열5
행1					
행2					
행3					
행4					
행5					



반복문(중첩 for문)

■ 삼각형 모양의 별 찍기1

hint) 열(column)이 변하는 것에 주목한다.

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

```
int i, j;  
for(i=1; i<=5; i++) {  
    for(j=1; j<i+1; j++) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

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

```
for(i = 1; i <= 5; i++) {  
    for(j = 5; j >= i; j--) {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```



반복문(중첩 for문)

■ 삼각형 모양의 별찍기2

hint) 공백과 별로 나눠서 생각

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

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

```
for(i = 1; i <= 5; i++) {
    for(j = 1; j <= 5-i; j++) {
        System.out.print(" "); //공백문자
    }
    for(j = 1; j <= i; j++) {
        System.out.print("*");
    }
    System.out.println(); //행 바꿈
}
```

```
for(i = 1; i <= 5; i++) {
    for(j = 1; j < i; j++) {
        System.out.print(" "); //공백문자
    }
    for(j = 5; j >= i; j--) {
        System.out.print("*");
    }
    System.out.println(); //행 바꿈
}
```



반복문(for문)

■ 중첩된 반복문(Nested Loop)

■ 구구단의 예

```
for(dan = 2; dan <= 9; dan++) {  
    for(times = 1; times <= 9; times++) {  
        System.out.println(dan + "x" + times + "=" + dan * times);  
    }  
    System.out.println();  
}
```

각 단에서 1~9를 곱하는 내부 반복문



break문 예제

- 실습 예제 – 반복조건문(break 사용)
 - 1부터 더했을때 그 합이 100이 넘는 자연수는?

```
int n = 1;
int sum = 0;
while(true) {
    sum += n;
    if(sum > 100)
        break;
    n++;
}
System.out.println("n=" + n);
System.out.println("sum=" + sum);
```

VS

```
int n = 1;
int sum = 0;
for(n=0; ; n++) {
    sum += n;
    if(sum > 100)
        break;
}
System.out.println("n=" + n);
System.out.println("sum=" + sum);
```

```
n=14
sum=105
```



break문 예제

구구단을 단보다 곱하는 수가 작거나 같은 경우까지만 출력하는 프로그램을 작성하세요(GugudanTest.java)

👉 실행 결과

```
2x1=2  
2x2=4  
  
3x1=3  
3x2=6  
3x3=9  
  
4x1=4  
4x2=8  
4x3=12  
4x4=16
```



continue문

▪ continue 문

- 반복문과 함께 쓰이며, 반복문 내부 continue 문을 만나면 이후 반복되는 부분을 수행하지 않고 조건식이나 증감식을 수행함
- 예제 - 1부터 10까지의 자연수 중 4나 8을 제외한 수를 출력하기

```
for(int i=1; i<=10; i++) {  
    if(i==4 || i==8)  
        continue;  
    System.out.println(i);  
}
```

1
2
3
5
6
7
9
10

```
// 홀수만 출력  
for(int i=1; i<11; i++) {  
    if(i%2==0)  
        continue;  
    System.out.println(i);  
}
```



continue문

- 실습 예제

continue문을 사용하여 구구단을 짝수 단만 출력하는 프로그램을 만드세요.

```
for(int i=2; i<10; i++) {  
    if(i % 2 == 1)  
        continue;  
    for(int j=1; j<10; j++) {  
        System.out.println(i + "x" + j + "=" + i*j);  
    }  
    System.out.println();  
}
```



반복문(for문)

■ 중첩된 반복문(Nested Loop)

```
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
```

```
int i, j;
for(i=0; i<5; i++) {
    for(j=1; j<=5; j++) {
        System.out.print(i*5+j + " ");
        //i*열의 끝수+j
    }
    System.out.println();
}
System.out.println();
```



자리배치도 프로그램 만들기

입장객 수에 따라 좌석을 배치하는 프로그램을 작성하세요.

(파일이름: Seats.java)

```
입장객 수 입력: 44
좌석 열의 수: 5
좌석1 좌석2 좌석3 좌석4 좌석5
좌석6 좌석7 좌석8 좌석9 좌석10
좌석11 좌석12 좌석13 좌석14 좌석15
좌석16 좌석17 좌석18 좌석19 좌석20
좌석21 좌석22 좌석23 좌석24 좌석25
좌석26 좌석27 좌석28 좌석29 좌석30
좌석31 좌석32 좌석33 좌석34 좌석35
좌석36 좌석37 좌석38 좌석39 좌석40
좌석41 좌석42 좌석43 좌석44
```



자리배치도 프로그램 만들기

```
import java.util.Scanner;

public class Seats {

    public static void main(String[] args) {
        //좌석 줄 수 계산
        //입장객 수, 좌석 열의 개수
        Scanner sc = new Scanner(System.in);
        System.out.print("입장객 수 : ");
        int customNum = sc.nextInt();
        System.out.print("좌석 열 수 : ");
        int colNum = sc.nextInt();

        int rowNum; //줄(행) 수

        if(customNum % colNum == 0) {
            rowNum = customNum / colNum;
        }else { //나머지 인원이 있는 경우 1줄 추가
            rowNum = customNum / colNum + 1
        }
    }
}
```

```
        //좌석 배치 반복
        for(int i = 0; i < rowNum; i++) { //줄
            for(int j = 1; j <= colNum; j++) { //열
                int seatNum = i * colNum + j; //좌석 번호
                if(seatNum > customNum)
                    break;
                System.out.print("좌석" + seatNum + " ");
            }
            System.out.println();
        }
        sc.close();
    }
}
```

