조건문과 반복문

컴퓨터공학전공
박요한

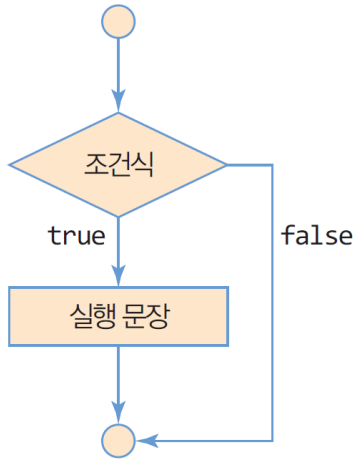
수업 내용

- 조건문
  - ✓ if
  - ✓ switch
  - ✓ case
- 반복문
  - ✓ for
  - ✓ while

단순 if문

- 단순 if 문
  - ✓if의 괄호 안에 조건식(논리형 변수나 논리 연산)
    - 실행문장이 단일 문장인 경우 둘러싸는 {, } 생략 가능

```
if (조건식) {
   ...실행 문장...  // 조건식이 참인 경우
}
```




```
if(n%2 == 0) {
   System.out.println(n + "은 짝수입니다.");
}
```

예제 2-10 : if문 사용하기

시험 점수가 80점이 이상이면 합격 판별을 하는 프로그램을 작성하시오.

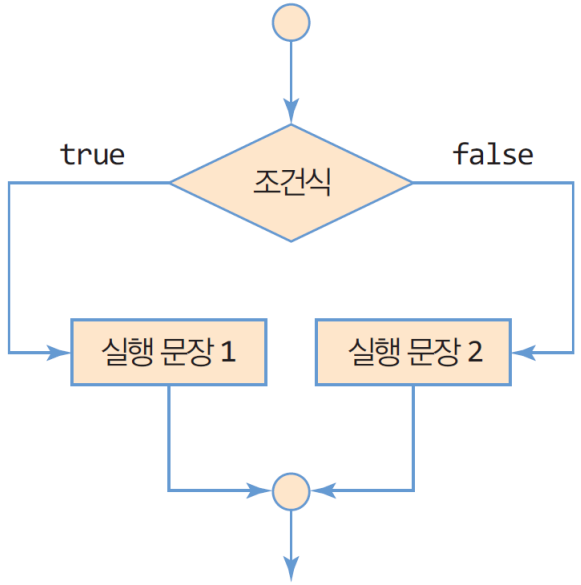
```
import java.util.Scanner;

public class SuccessOrFail {
   public static void main (String[] args) {
      Scanner scanner = new Scanner(System.in);

      System.out.print("점수를 입력하시오: ");
      int score = scanner.nextInt();
      if (score >= 80)
         System.out.println("축하합니다! 합격입니다.");

      scanner.close();
   }
}
```

```
점수를 입력하시오: 95
축하합니다! 합격입니다.
```

조건문 - if-else

- if-else 문
  - ✓조건식이 true면 실행문장1 실행 후 if-else문을 벗어남
  - ✓false인 경우에 실행문장2 실행후, if-else문을 벗어남

```
if (조건식) {
   ...실행 문장 1...
}
else {
   ...실행 문장 2...
}
```



예제 2-11 : if-else 사용하기

입력된 수가 3의 배수인지 판별하는 프로그램을 작성하시오.

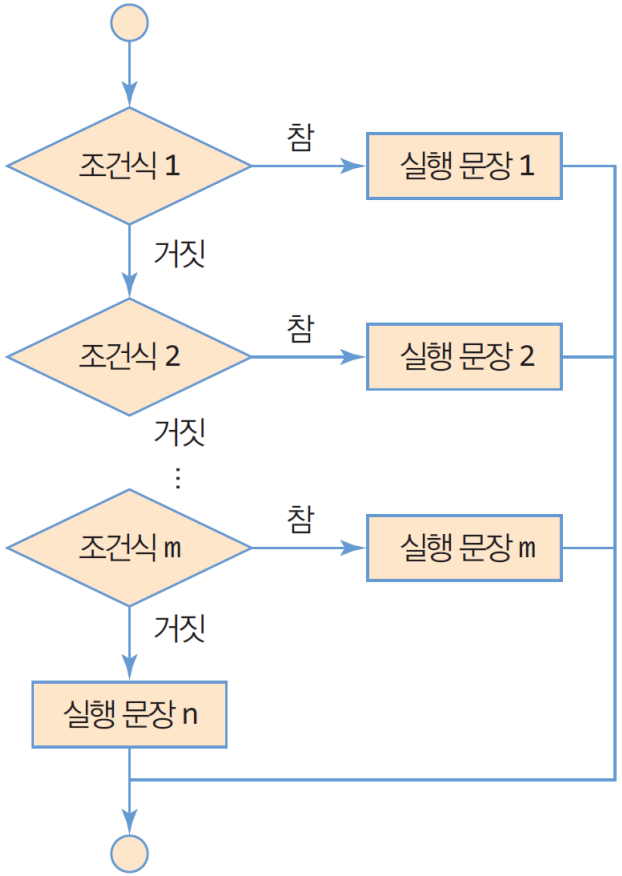
```
import java.util.Scanner;

public class MultipleOfThree {
   public static void main (String[] args) {
      Scanner in = new Scanner(System.in);

      System.out.print("수를 입력하시오: ");
      int number = in.nextInt();

      if (number % 3 == 0)
         System.out.println("3의 배수입니다.");
      else
         System.out.println("3의 배수가 아닙니다.");

      scanner.close();
   }
}
```

```
수를 입력하시오: 129
3의 배수입니다.
```

다중 if-else 문

- 다중 if-else 문
  - ✓ if-else가 연속되는 모양
    - 조건문

```
if (조건식 1) {
   실행 문장 1; // 조건식 1이 참인 경우
}
else if (조건식 2) {
   실행 문장 2; // 조건식 2가 참인 경우
}
else if (조건식 m) {
   ..... // 조건식 m이 참인 경우
}
else {
   실행 문장 n; // 앞의 모든 조건이 거짓인 경우
}
```



예제 2-12 : 다중 if-else로 학점 매기기

다중 if-else문을 이용하여 입력받은 성적에 대해 학점을 부여하는 프로그램을 작성해보자.

```
import java.util.Scanner;
public class Grading {
    public static void main(String[] args) {
        char grade;
        Scanner scanner = new Scanner(System.in);

        System.out.print("점수를 입력하세요(0~100): ");
        int score = scanner.nextInt(); // 점수 읽기
        if(score >= 90) // score가 90 이상
            grade = 'A';
        else if(score >= 80) // score가 80 이상 90 미만
            grade = 'B';
        else if(score >= 70) // score가 70 이상 80 미만
            grade =  ' C ' ;
        else if(score >= 60) // score가 60 이상 70 미만
            grade = 'D';
        else // score가 60 이만
            grade = 'F';
        System.out.println("학점은 "+ grade + "입니다.");

        scanner.close();
    }
}
```

```
점수를 입력하세요(0~100): 89
학점은 B입니다.
```

예제 2-13 : 중첩 if-else 문 사례

점수와 학년을 입력 받아 60점 이상이면 합격, 미만이면 불합격을 출력한다. 4학년의 경우 70점 이상이어야 합격이다.

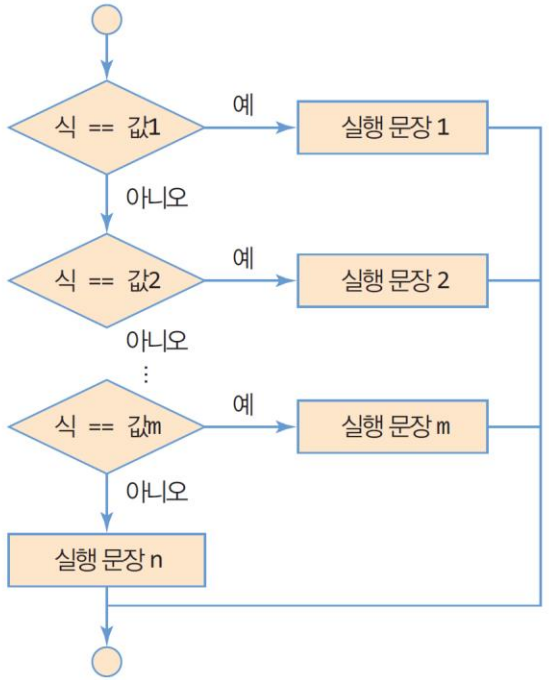
```
import java.util.Scanner;
public class NestedIf {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("점수를 입력하세요(0~100): ");
        int score = scanner.nextInt();

        System.out.print("학년을 입력하세요(1~4): ");
        int year = scanner.nextInt();

        if(score >= 60) { // 60점 이상
            if(year != 4)
                System.out.println("합격!"); // 4학년 아니면 합격
            else if(score >= 70)
                System.out.println("합격!"); // 4학년이 70점 이상이면 합격
        else
            System.out.println("불합격!"); // 4학년이 70점 미만이면 불합격
        }
        else // 60점 미만 불합격
            System.out.println("불합격!");

        scanner.close();
    }
}
```

```
점수를 입력하세요(0~100): 65
학년을 입력하세요(1~4): 4
불합격!
```

switch문

- switch문은 식과 case 문의 값과 비교
  - ✓case의 비교 값과 일치하면 해당 case의 실행문장 수행
    - break를 만나면 switch문을 벗어남
  - ✓case의 비교 값과 일치하는 것이 없으면 default 문 실행
- default문은 생략 가능

```
switch (식) {
   case 값1:
      실행 문장 1;
      break;
   case 값2:
      실행 문장 2;
      break;
   ...
   case 값m:
      실행 문장 m;
      break;
   default:
      실행 문장 n;
}
```



예제 2-14 switch 문으로 학점 매기기

```
import java.util.Scanner;
public class Grading {
    public static void main(String[] args) {
        char grade;
        Scanner scanner = new Scanner(System.in);

        System.out.print("점수를 입력하세요(0~100): ");
        int score = scanner.nextInt(); // 점수 읽기
        if(score >= 90) // score가 90 이상
            grade = 'A';
        else if(score >= 80) // score가 80 이상 90 미만
            grade = 'B';
        else if(score >= 70) // score가 70 이상 80 미만
            grade = ' C ' ;
        else if(score >= 60) // score가 60 이상 70 미만
            grade = 'D';
        else // score가 60 이만
            grade = 'F';
        System.out.println("학점은 "+ grade + "입니다.");

        scanner.close();
    }
}
```

```
import java.util.Scanner;
public class GradingSwitch {
    public static void main (String[] args) {
        Scanner scanner = new Scanner(System.in);

        char grade;
        System.out.print("점수를 입력하세요(0~100): ");
        int score = scanner.nextInt();
        switch (score/10) {
            case 10: // score = 100
            case 9: // score는 90~99
                grade = 'A';
                break;
            case 8: // score는 80~89
                grade = 'B';
                break;
            case 7: // score는 70~79
                grade = 'C';
                break;
            case 6: // score는 60~69
                grade = 'D';
                break;
            default: // score는 59 이하
                grade = 'F';
        }
        System.out.println("학점은 "+grade+"입니다");
        scanner.close();
    }
}
```

switch문에서 벗어나기

- switch문 내의 break문
  - ✓ break문 만나면 switch문 벗어남
  - ✓ case 문에 break문이 없다면, 다음 case문으로 실행 계속
    - 언젠가 break를 만날 때까지 계속 내려 가면서 실행

```
char grade='A';
switch (grade) {
   case 'A':
      System.out.println("90 ~ 100점입니다.");
      break;
   case 'B':
      System.out.println("80 ~ 89점입니다.");
      break;
   case 'C':
      System.out.println("70 ~ 79점입니다.");
      break;
}
```

```
90 ~ 100점입니다.
80 ~ 89점입니다.
```

case 문의 값

- case 문의 값
  - ✓ 문자, 정수, 문자열 리터럴만 허용
  - ✓ 실수 리터럴은 허용되지 않음

```
int b;
switch(b%2) {
   case 1 : ...; break;
   case 2 : ...; break;
}

char c;
switch(c) {
   case '+' : ...; break;
   case '-' : ...; break;
}

String s = "예";
switch(s) {
   case "예" : ...; break;
   case "아니요" : ...; break;
}
```

정수 리터럴 사용 가능

문자 리터럴 사용 가능

문자열 리터럴 사용 가능

오류

```
switch(a) {
   case a :          // 오류. 변수 사용 안됨
   case a > 3 :      // 오류. 수식 안됨
   case a == 1 :     // 오류. 수식 안됨
}
```

예제 2-15 : switch 문 활용

switch 문을 이용하여 커피 메뉴의 가격을 알려주는 프로그램을 작성하라.
에스프레소, 카푸치노, 카페라떼는 3500원이고, 아메리카노는 2000원이다.

```
import java.util.Scanner;
public class CoffeePrice {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("무슨 커피 드릴까요? ");
        String order = scanner.next();
        int price=0;
        switch (order) {
            case "에스프레소":
            case "카푸치노":
            case "카페라떼":
                price = 3500;
                break;
            case "아메리카노" :
                price = 2000;
                break;
            default:
                System.out.println("메뉴에 없습니다!");
        }
        if(price != 0)
            System.out.print(order + "는 " + price + "원입니다");
        scanner.close();
    }
}
```

```
무슨 커피 드릴까요? 에스프레소
에스프레소는 3500원입니다
```

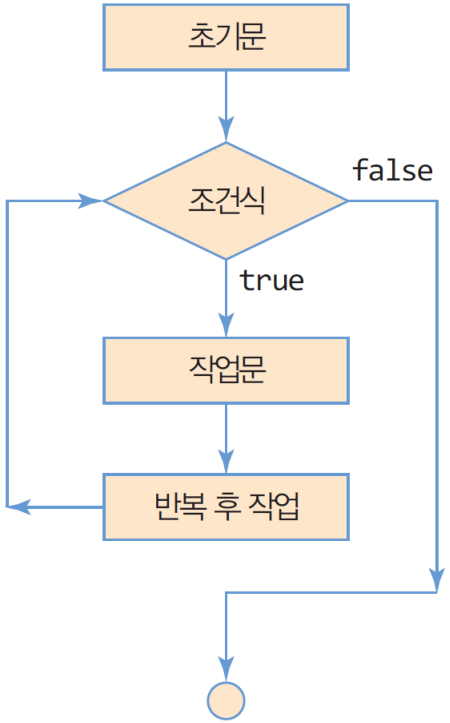
반복문의 특징

- 자바 반복문의 종류
  - ✓for 문
  - ✓while 문
  - ✓do while 문

for 문의 구성

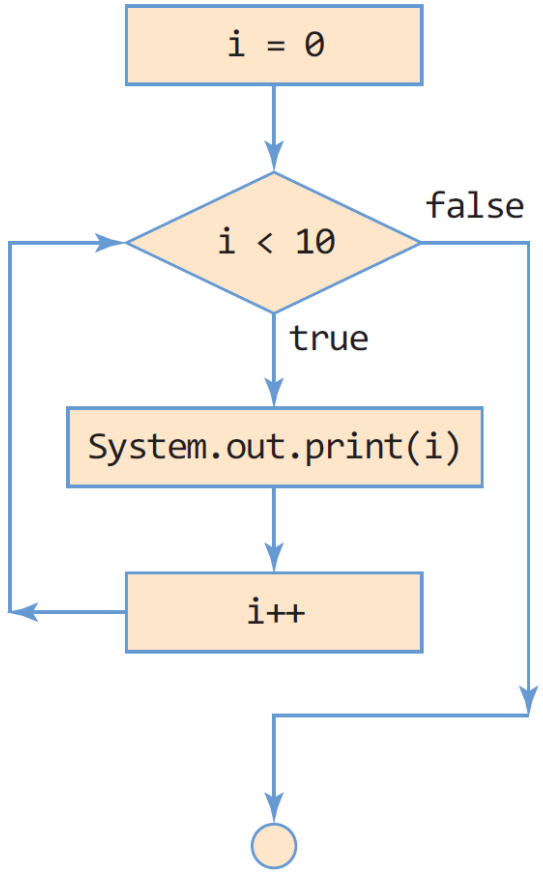
```
for (초기문; 조건식; 반복 후 작업) {
    ..작업문..
}
```

1 2 4 3




```
for(i=0; i<10; i++) { // i가 0~9까지 10번 반복
                System.out.print(i); // 0에서 9까지 출력
}
```

for 문의 실행 과정을 나타내는 순서도




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

```
0123456789
```

for문의 예시

- 0에서 9까지 정수 출력

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

```
int i;
for(i = 0; i < 10; i++)
   System.out.print(i);
```

- 반복문에 변수 선언 가능

```
for(int i = 0; i < 10; i++) // 변수 i는 for문을 벗어나서 사용할 수 없음
   System.out.print(i);
```

- 0에서 100까지의 합 구하기

```
int sum = 0;
for(int i = 0; i <= 100; i++)
   sum += i;
```

```
int i,  sum;
for(i = 0, sum=0; i <= 100; i++)
   sum += i;
```

```
int sum = 0;
for(int i = 100; i >= 0; i--)
   sum += i;
```

for문의 특이한 형태

```
for(초기작업; true; 반복후작업) { // 반복 조건이 true이면  무한 반복
.....
}
```

```
for(초기작업; ; 반복후작업) { // 반복조건이 비어 있으면 true로 간주, 무한 반복
.....
}
```

```
// 초기 작업과 반복후작업은 ','로 분리하여 여러 문장 나열 가능
for(i=0; i<10; i++, System.out.println(i)) {
.....
}
```

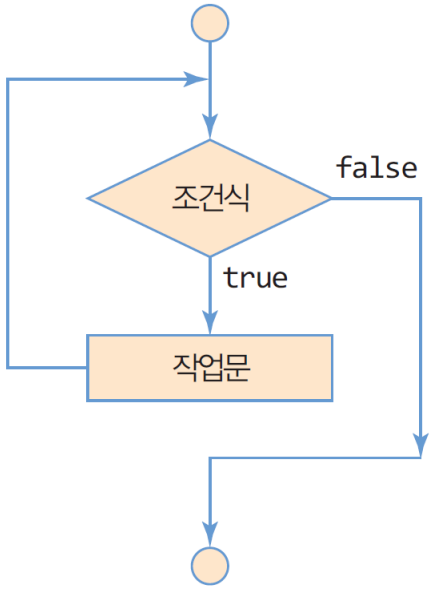
```
// for문 내에 변수 선언
for(int i=0; i<10; i++) { // 변수 i는 for문 내에서만 사용 가능
.....
}
```

while 문의 구성

```
while (조건식) {
    ..작업문..
}
```

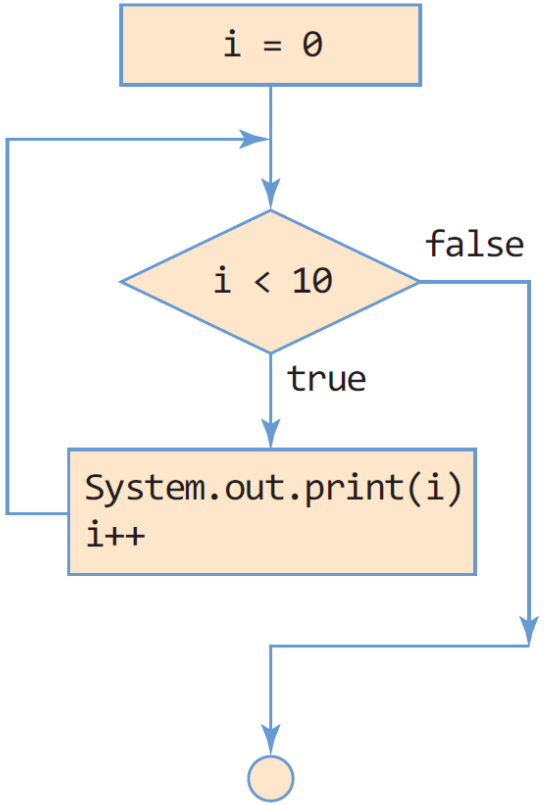
1

2



- 반복 조건이 true이면 반복, false이면 반복 종료
- 반복 조건이 없으면 컴파일 오류
- 처음부터 반복조건을 통과한 후 작업문 수행

while문의 실행 과정을 나타내는 순서도




```
i = 0;
while(i<10) {
    System.out.print(i);
    i++;
}
```

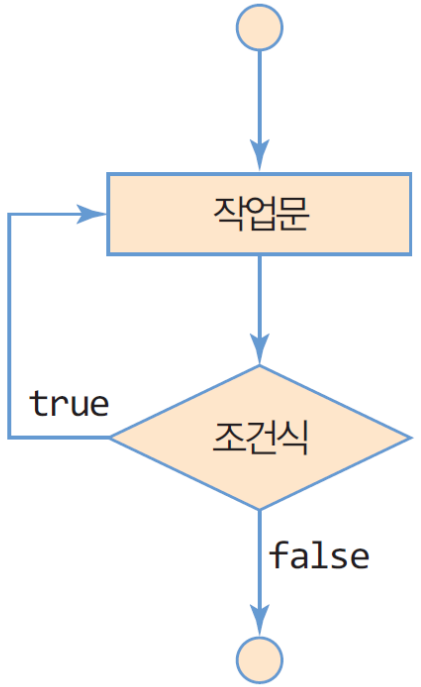
```
0123456789
```

do-while 문의 구성

```
do {
   ..작업문..
} while(조건식);
```

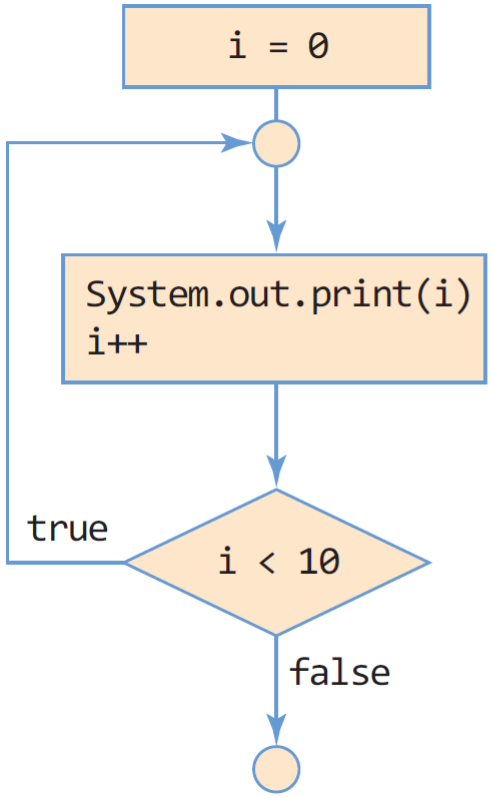
1

2



- 무조건 최소 한번 작업문 실행

do-while문의 실행 과정을 나타내는 순서도




```
i = 0;
do {
    System.out.print(i);
    i++;
} while(i<10);
```

```
0123456789
```

중첩 반복

- 중첩 반복
  - ✓반복문이 다른 반복문을 내포하는 구조
  - ✓이론적으로는 몇 번이고 중첩 반복 가능
  - ✓너무 많은 중첩 반복은 프로그램 구조를 복잡하게 하므로 2중 또는 3중 반복이 적당

```
for(int i=0; i<100; i++) { // 100개의 학교 성적을 모두 더한다.
        .....
        for(int j=0; j<10000; j++) { // 10000명의 학생 성적을 모두 더한다.
              .....
              .....
           }
     .....
}
```

10000명의 학생이 있는 100개 대학의 모든 학생 성적의 합을 구할 때,
for 문을 이용한 이중 중첩 구조

예제 3-4 : 2중 중첩을 이용한 구구단

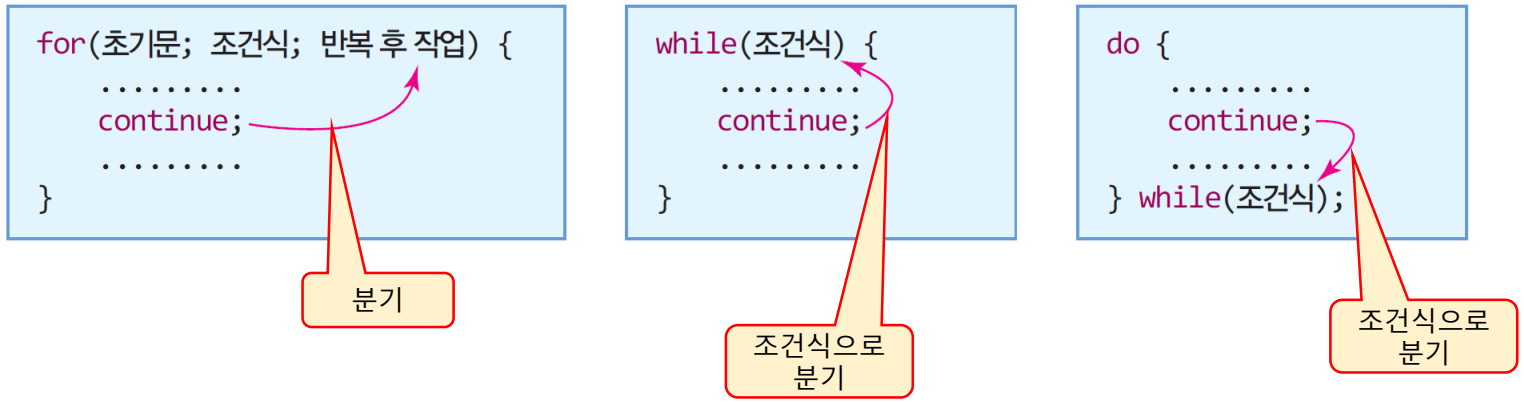
2중 중첩 for문을 사용하여 구구단을 출력하는 프로그램을 작성하시오. 한 줄에 한 단씩 출력한다.

```
public class NestedLoop {
   public static void main(String[] args) {
      for(int i=1; i<10; i++) { // 1단에서 9단
         for(int j=1; j<10; j++) { // 각 단의 구구셈 출력
            System.out.print(i + "*" + j + "=" + i*j); // 구구셈 출력
            System.out.print('₩t'); // 하나씩 탭으로 띄기
         }
         System.out.println(); // 한 단이 끝나면 다음 줄로 커서 이동
      }
   }
}
```

```
1*1=1   1*2=2   1*3=3   1*4=4   1*5=5   1*6=6   1*7=7   1*8=8   1*9=9
2*1=2   2*2=4   2*3=6   2*4=8   2*5=10  2*6=12  2*7=14  2*8=16  2*9=18
3*1=3   3*2=6   3*3=9   3*4=12  3*5=15  3*6=18  3*7=21  3*8=24  3*9=27
4*1=4   4*2=8   4*3=12  4*4=16  4*5=20  4*6=24  4*7=28  4*8=32  4*9=36
5*1=5   5*2=10  5*3=15  5*4=20  5*5=25  5*6=30  5*7=35  5*8=40  5*9=45
6*1=6   6*2=12  6*3=18  6*4=24  6*5=30  6*6=36  6*7=42  6*8=48  6*9=54
7*1=7   7*2=14  7*3=21  7*4=28  7*5=35  7*6=42  7*7=49  7*8=56  7*9=63
8*1=8   8*2=16  8*3=24  8*4=32  8*5=40  8*6=48  8*7=56  8*8=64  8*9=72
9*1=9   9*2=18  9*3=27  9*4=36  9*5=45  9*6=54  9*7=63  9*8=72  9*9=81
```

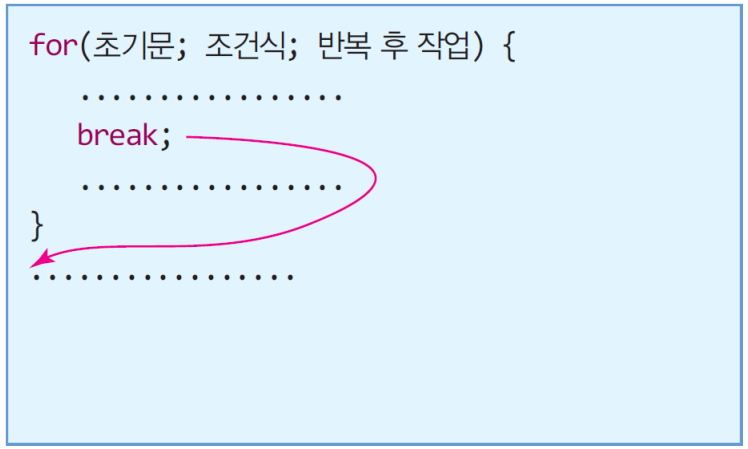
continue문

- continue 문
  - ✓반복문을 빠져 나가지 않으면서 다음 반복으로 진행

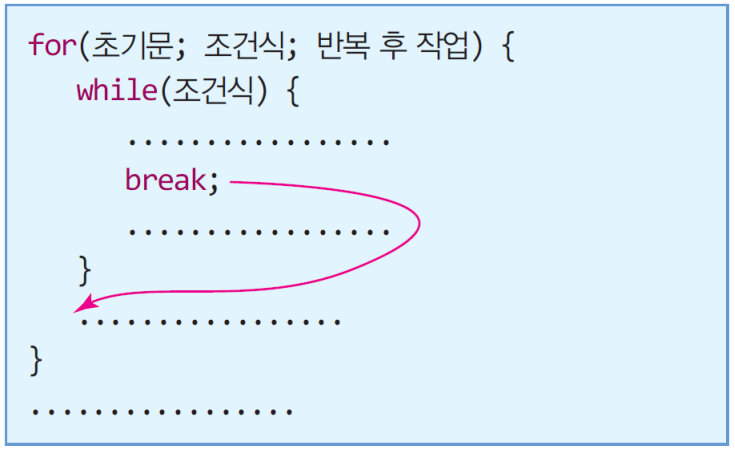


break문

- break 문
  - ✓반복문 하나를 완전히 빠져 나갈 때 사용
    - 하나의 반복문만 벗어남
    - 중첩 반복의 경우 안쪽 반복문의 break 문이 실행되면 안쪽 반복문만 벗어남



(a) 현재 반복문 벗어나기



(b) 중첩 반복에서 안쪽 반복문만 벗어나는 경우