

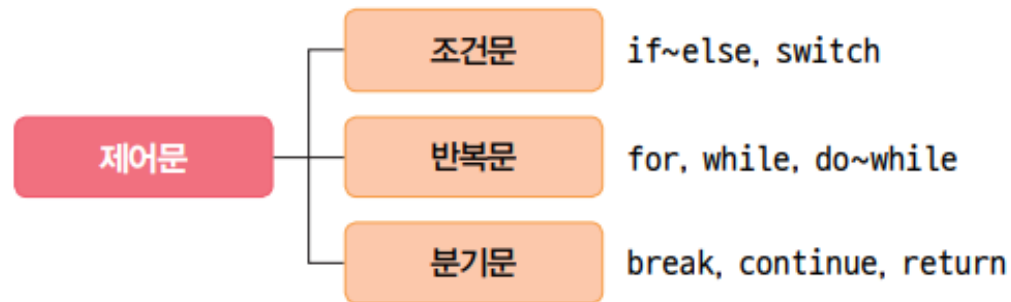
Section 01

제어문

1. 제어문

■ 제어문

- 프로그램을 구성하는 코드가 실행될 때 코드의 처리 순서를 변경할 수 있는 문장
- 명령의 흐름 처리 방법을 지정할 때 사용함
 - **조건문**: 조건에 따라 처리하는 명령을 달리하는 문장
 - **반복문**: 일정한 조건이 충족될 때까지 동일한 명령을 반복적으로 처리하는 문장
 - **분기문**: 강제로 명령의 처리 순서를 바꾸는 문장



[그림 4-1] 제어문의 유형

Section 02

조건문

2. 조건문

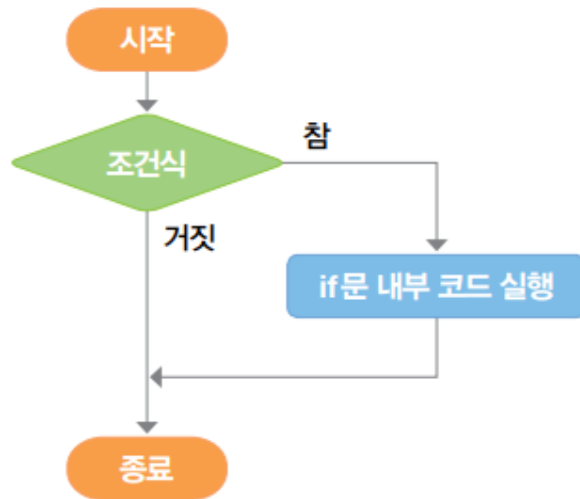
■ if문

- 가장 간단한 조건문

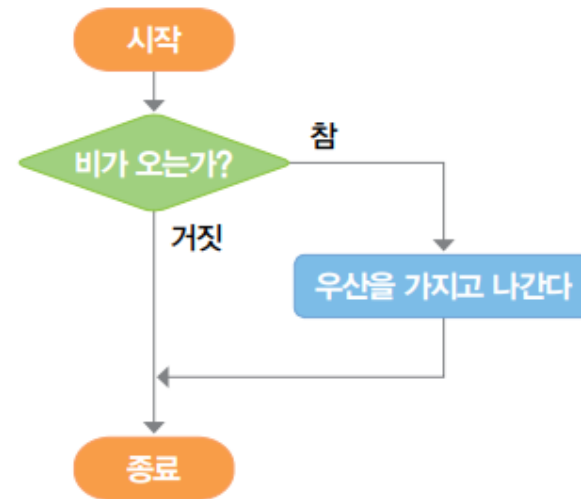
→ 특정 코드 또는 코드 블록 실행 여부를 결정하는 데 사용함

```
if (조건식) {  
    // 조건식이 참이면 실행되는 코드  
}
```

결과가 참 또는 거짓인 연산식이나 불리언 변수



(a) if문



(b) if문의 예(우산 지참 여부)

[그림 4-3] if문의 순서도

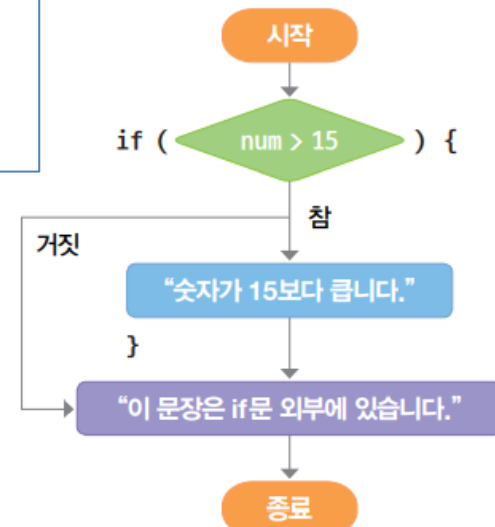
2. 조건문

if문 사용 예시

```
public class Example01 {  
    public static void main(String[] args) {  
        int num = 10;  
  
        if (num > 15) {  
            System.out.println("숫자가 15보다 큼니다.");  
        }  
  
        System.out.println("이 문장은 if문 외부에 있습니다.");  
    }  
}
```

실행 결과

이 문장은 if문 외부에 있습니다.



[그림 4-4] Example01.java 프로그램의 if문 처리 과정

2. 조건문

예제 4-1 투표가 가능한 나이 판별하기

```
01 import java.util.Scanner;
02
03 public class If01 {
04     public static void main(String[] args) {
05         System.out.println("당신의 나이를 입력하세요.");
06
07         Scanner s = new Scanner(System.in);
08         int age = s.nextInt();
09
10         if ( age >= 18) {
11             System.out.println("당신의 나이는 18세 이상입니다.");
12             System.out.println("당신은 투표할 자격이 있습니다.");
13         }
14
15         System.out.println("이 문장은 if문 외부에 있습니다.");
16     }
17 }
```

실행 결과

당신의 나이를 입력하세요.

20

당신의 나이는 18세 이상입니다.

당신은 투표할 자격이 있습니다.

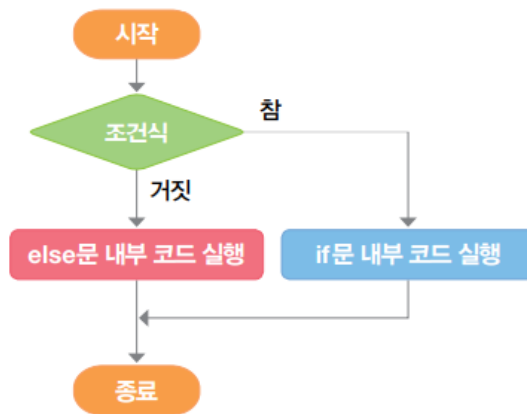
이 문장은 if문 외부에 있습니다.

2. 조건문

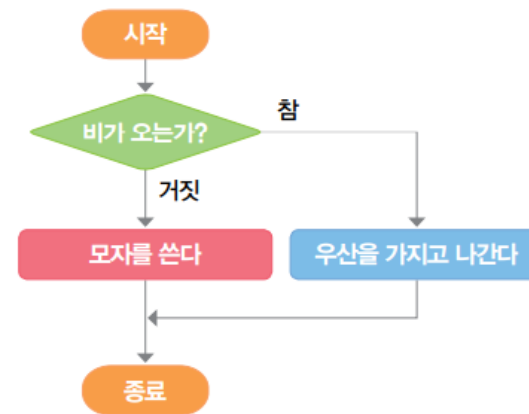
■ if~else문

- if문과 else문을 함께 사용하여 조건식의 결과에 따라 특정 코드 또는 코드 블록의 실행을 선택하는 조건문

```
if (조건식) {  
    // 조건식이 참이면 실행되는 코드  
}  
else {  
    // 조건식이 거짓이면 실행되는 코드  
}
```



(a) if~else문



(b) if~else문의 예(우산 지참 여부)

[그림 4-5] if~else문의 순서도

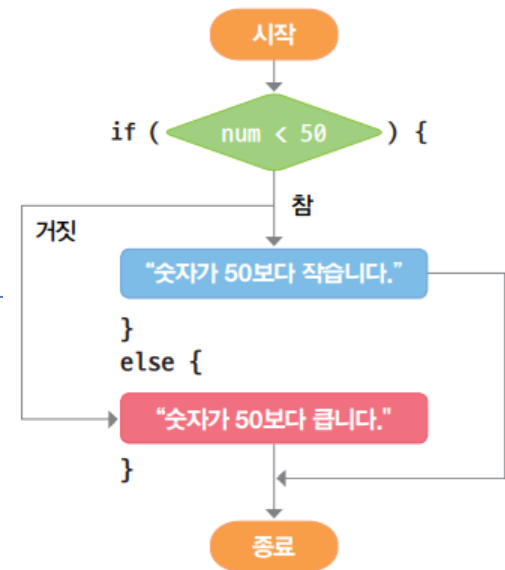
2. 조건문

if~else문 사용 예시

```
public class Example02 {  
    public static void main(String[] args) {  
        int num = 120;  
  
        if ( num < 50) {  
            System.out.println("숫자가 50보다 작습니다.");  
        }  
        else {  
            System.out.println("숫자가 50보다 큼니다.");  
        }  
    }  
}
```

실행 결과

숫자가 50보다 큼니다.



[그림 4-6] Example02.java 프로그램의 if~else문 처리 과정

2. 조건문

예제 4-2 짝수, 홀수 판별하기

```
01 import java.util.Scanner;
02
03 public class If02 {
04     public static void main(String[] args) {
05         System.out.println("숫자를 입력하세요.");
06
07         Scanner s = new Scanner(System.in);
08         int num = s.nextInt();
09
10         if ( num % 2 == 0 ) {
11             System.out.println(num + "은 짝수입니다.");
12         }
13         else {
14             System.out.println(num + "은 홀수입니다.");
15         }
16
17         System.out.println("이 문장은 if문 외부에 있습니다.");
18     }
19 }
```

실행 결과

숫자를 입력하세요.

6

6은 짝수입니다.

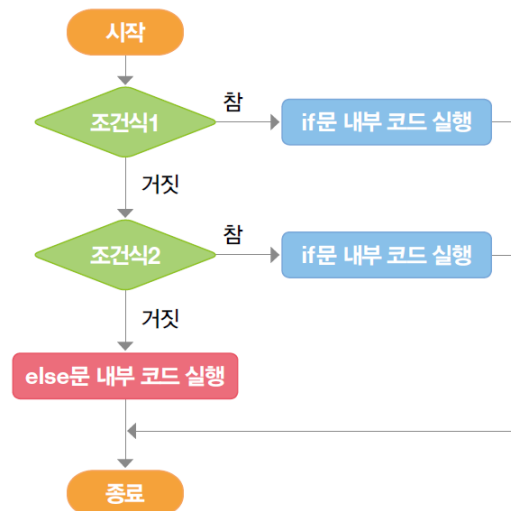
이 문장은 if문 외부에 있습니다.

2. 조건문

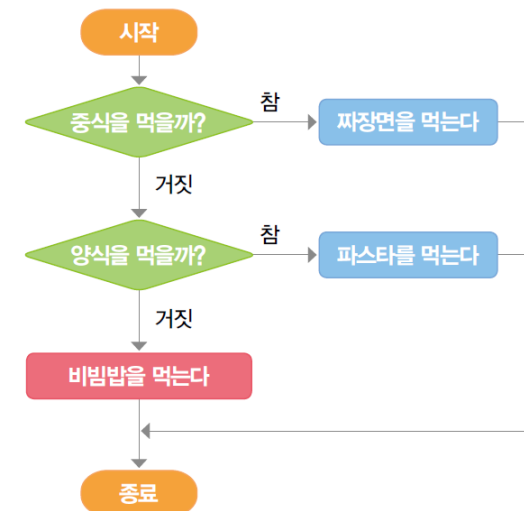
■ if~else if문

- if~else문과 유사하나 else문이 서로 다른 if문의 조건식과 쌍을 이루는 조건문임
- else if문의 수는 제한이 없음
- else if문 코드 블록의 마지막에 else문의 코드 블록을 추가할 수 있음

```
if (조건식1) {  
    // 조건식1이 참이면 실행되는 코드  
}  
else if (조건식2) {  
    // 조건식2가 참이면 실행되는 코드  
    ...  
}  
else if (조건식n) {  
    // 조건식n이 참이면 실행되는 코드  
}  
else {  
    // 조건식1~조건식n이 거짓이면 실행되는 코드  
}
```



(a) if~else if문



(b) if~else if문의 예(메뉴 선택)

[그림 4-7] if~else if문의 순서도

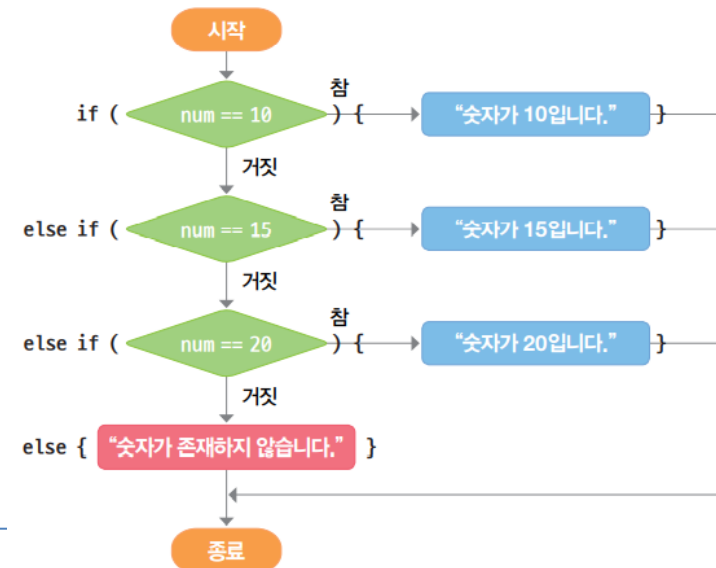
2. 조건문

if~else if문 사용 예시

```
public class Example03 {  
    public static void main(String[] args) {  
        int num = 20;  
  
        if (num == 10) {  
            System.out.println("숫자가 10입니다.");  
        }  
        else if (num == 15) {  
            System.out.println("숫자가 15입니다.");  
        }  
        else if (num == 20) {  
            System.out.println("숫자가 20입니다.");  
        }  
        else {  
            System.out.println("숫자가 존재하지 않습니다.");  
        }  
    }  
}
```

실행 결과

숫자가 20입니다.



[그림 4-8] Example03.java 프로그램의 if~else문 처리 과정

2. 조건문

예제 4-3 점수에 따라 학점 부여하기

```
01 import java.util.Scanner;
02
03 public class If03 {
04     public static void main(String[] args) {
05         System.out.println("점수를 입력하세요.");
06
07         Scanner s = new Scanner(System.in);
08         int grade = s.nextInt();
09
10         if (grade >= 90) {
11             System.out.println("A 학점");
12         }
13         else if (grade >= 80) {
14             System.out.println("B 학점");
15         }
```

2. 조건문

예제 4-3 점수에 따라 학점 부여하기

```
16     else if (grade >= 70) {  
17         System.out.println("C 학점");  
18     }  
19     else if (grade >= 60) {  
20         System.out.println("D 학점");  
21     }  
22     else {  
23         System.out.println("F 학점");  
24     }  
25 }  
26 }
```

실행 결과

점수를 입력하세요.

85

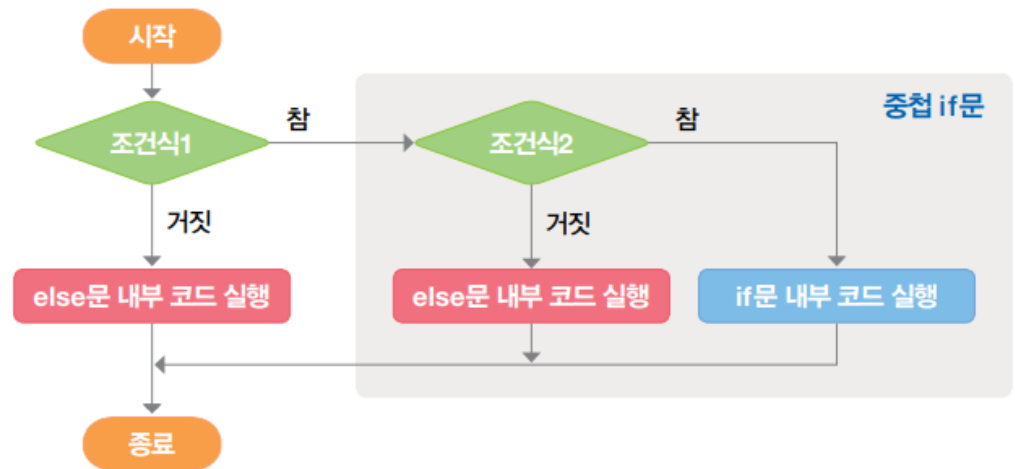
B 학점

2. 조건문

■ 중첩 if~else문

- if문 또는 else문 코드 블록 내에 다른 if문, if~else문, if~else if문 등이 포함된 조건문
- 외부 if문의 조건식이 참이면 내부 if문의 코드 블록을 실행함

```
if (조건식1) {  
    // 조건식1이 참이면 실행되는 코드  
    if (조건식2) {  
        // 조건식2가 참이면 실행되는 코드  
    }  
    else {  
        // 조건식2가 거짓이면 실행되는 코드  
    }  
}  
else {  
}
```



[그림 4-9] 중첩 if~else문의 순서도

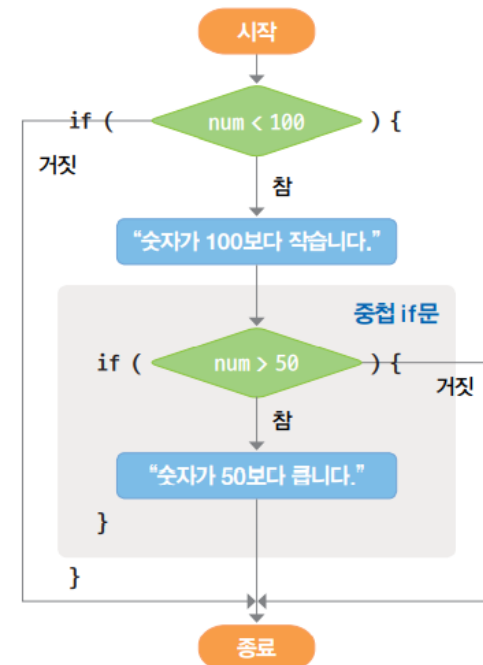
2. 조건문

중첩 if~else문 사용 예시

```
public class Example04 {  
    public static void main(String[] args) {  
        int num = 70;  
        if (num < 100) {  
            System.out.println("숫자가 100보다 작습니다.");  
            if (num > 50) {  
                System.out.println("숫자가 50보다 큼니다.");  
            }  
        }  
    }  
}
```

실행 결과

숫자가 100보다 작습니다.
숫자가 50보다 큼니다.



[그림 4-10] Example04.java 프로그램의 if~else문 처리 과정

2. 조건문

예제 4-4 첫 번째로 입력된 값이 최댓값인지 판별하기

```
01 import java.util.Scanner;
02
03 public class If04 {
04     public static void main(String[] args)
05         System.out.println("3개의 숫자를 입력하세요.");
06
07     Scanner s = new Scanner(System.in);
08     int x = s.nextInt();
09     int y = s.nextInt();
10     int z = s.nextInt();
11
12     if (x > y) {
13         if (x > z) {
14             System.out.println(x + "는 가장 큰 정수입니다.");
15         }
```


2. 조건문

예제 4-4 첫 번째로 입력된 값이 최댓값인지 판별하기

```
16     else {  
17         System.out.println(x + "는 가장 큰 정수가 아닙니다.");  
18     }  
19 }  
20 else {  
21     System.out.println(x + "는 가장 큰 정수가 아닙니다.");  
22 }  
23 }  
24 }
```

실행 결과

3개의 숫자를 입력하세요.

32 25 40

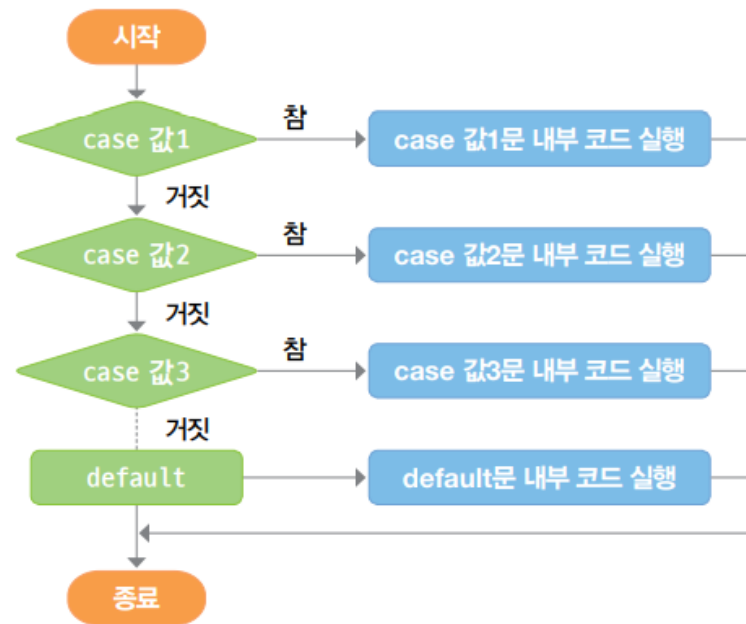
32는 가장 큰 정수가 아닙니다.

2. 조건문

■ switch문

- if~else if문처럼 많은 선택 사항이 있고 각 선택 사항에 대해 특정 코드 또는 코드 블록을 실행하는 조건문

```
switch (연산식 또는 변수) {  
    case 값1:  
        // 실행 코드  
        break;  
    case 값2:  
        // 실행 코드  
        break;  
    ...  
    case 값n:  
        // 실행 코드  
    default:  
        // 실행 코드  
}
```



[그림 4-11] switch문의 순서도

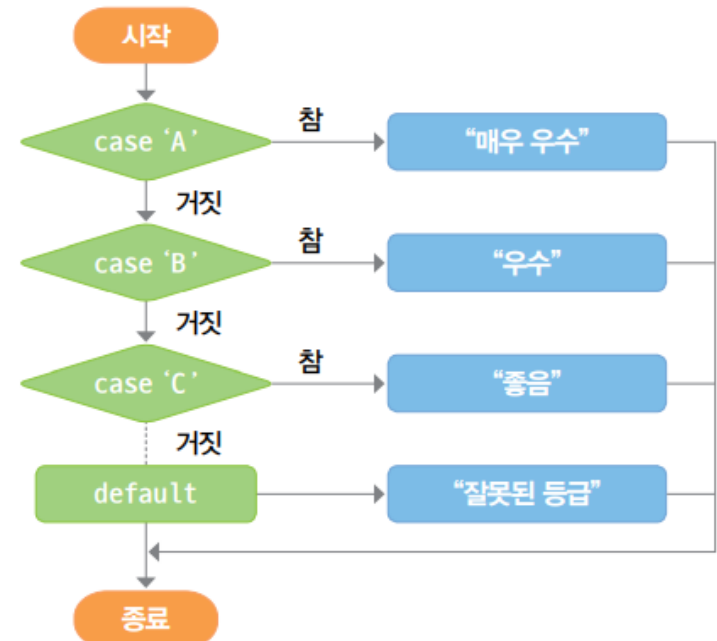
2. 조건문

switch문 사용 예시

```
public class Example06 {  
    public static void main(String[] args) {  
        char grade = 'B';  
        switch (grade) {  
            case 'A':  
                System.out.println("매우 우수");  
                break;  
            case 'B':  
                System.out.println("우수");  
                break;  
            case 'C':  
                System.out.println(" 좋음");  
                break;  
            case 'D':  
                System.out.println("좀 더 열심히");  
                break;  
            case 'F':  
                System.out.println("미흡");  
                break;  
            default:  
                System.out.println("잘못된 등급");  
        }  
    }  
}
```

실행 결과

우수



[그림 4-12] Example06.java 프로그램의 switch문 처리 과정

2. 조건문

예제 4-5 switch문을 활용하여 점수에 따라 학점 부여하기

```
01 import java.util.Scanner;
02
03 public class Switch01 {
04     public static void main(String[] args) {
05         System.out.println("점수를 입력하세요.");
06
07         Scanner s = new Scanner(System.in);
08         int num = s.nextInt();
09
10         switch (num / 10) {
11             case 10 :
12             case 9 :
13                 System.out.println("A 학점");
14                 break;
```

2. 조건문

예제 4-4 첫 번째로 입력된 값이 최댓값인지 판별하기

```
15     case 8:
16         System.out.println("B 학점");
17         break;
18     case 7:
19         System.out.println("C 학점");
20         break;
21     case 6:
22         System.out.println("D 학점");
23         break;
24     default:
25         System.out.println("F 학점");
26         break;
27 }
28 }
29 }
```

실행 결과

점수를 입력하세요.

100

A 학점

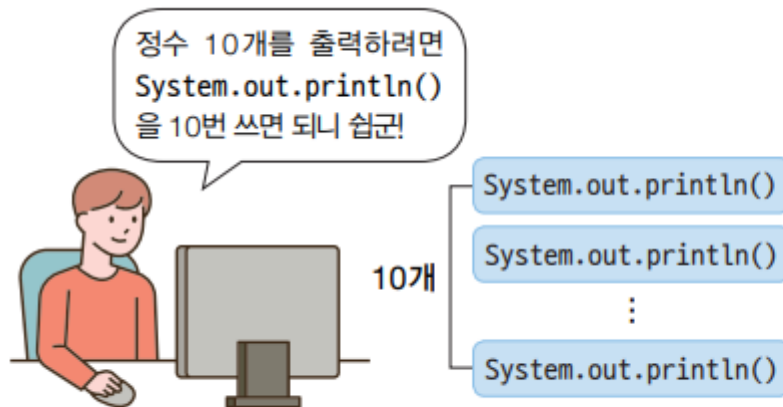
Section 03

반복문

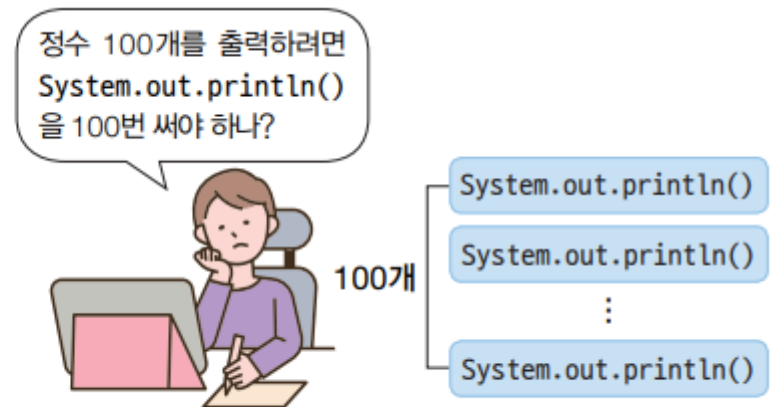
3. 반복문

■ 반복문의 개요

- 반복문
 - 일정 횟수만큼 어떤 코드를 반복하여 실행하는 제어문
 - 반복적으로 실행되는 동작을 루프(loop)라고 함
 - for문, while문, do~while문



(a) 같은 코드를 10번 작성하는 경우



(b) 같은 코드를 100번 작성하는 경우

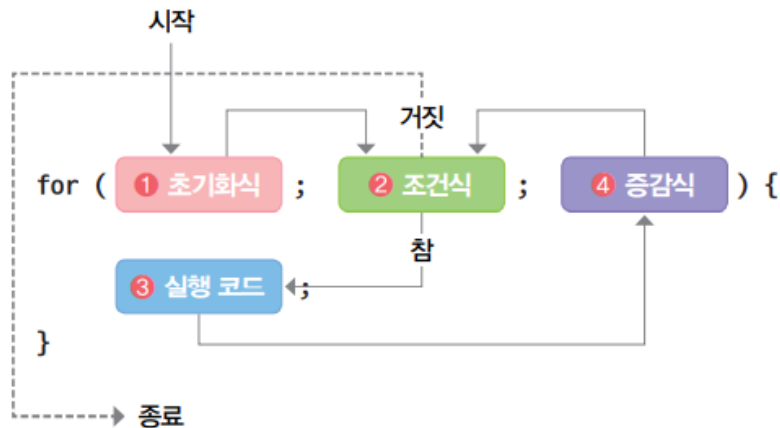
[그림 4-13] 반복문의 예

3. 반복문

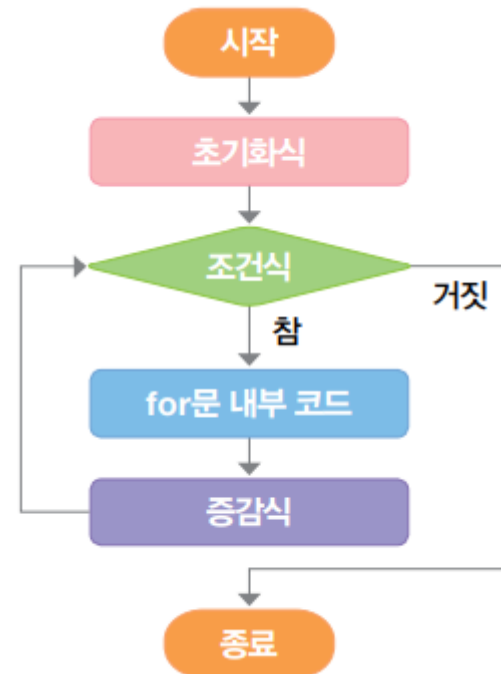
■ for문

- 특정 횟수만큼 실행해야 하는 루프를 효율적으로 쓸 수 있는 반복문
- 작업을 반복하는 횟수를 알고 있을 때 유용함

```
for (초기화식; 조건식; 증감식) {  
    // 실행 코드  
}
```



[그림 4-14] for문의 실행 흐름



[그림 4-15] for문의 순서도

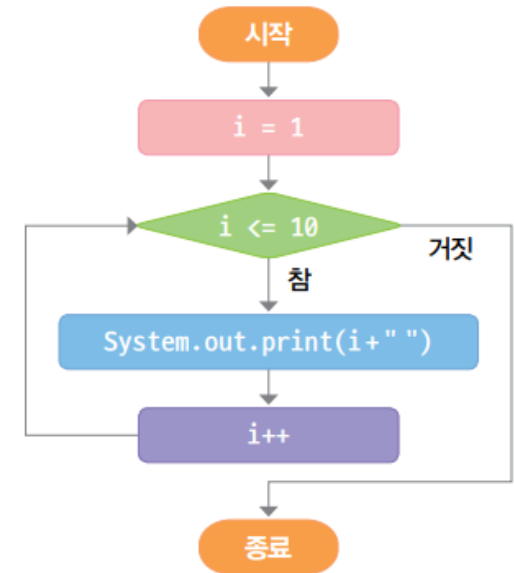
3. 반복문

for문 사용 예시

```
public class Example07 {  
    public static void main(String[] args) {  
        int i;  
  
        for (i = 1; i <= 10; i++) {  
            System.out.print(i + " ");  
        }  
    }  
}
```

실행 결과

1 2 3 4 5 6 7 8 9 10



[그림 4-16] Example07.java 프로그램의 for문 처리 과정

3. 반복문

예제 4-6 입력받은 5개 숫자의 합 구하기

```
01 import java.util.Scanner;
02
03 public class Loop01 {
04     public static void main(String[] args) {
05         int sum = 0;
06         System.out.println("5개의 숫자를 입력하세요.");
07
08         for (int n = 1; n <= 5; n++) {
09             Scanner s = new Scanner(System.in);
10             int num = s.nextInt();
11             sum += num;      // sum = sum + num과 같은 코드
12         }
13
14         System.out.println("합계 : " + sum);
15     }
16 }
```

실행 결과

5개의 숫자를 입력하세요.

20

10

40

30

10

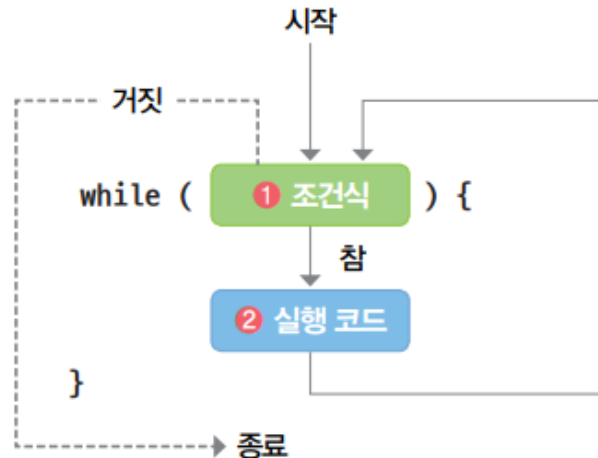
합계 : 110

3. 반복문

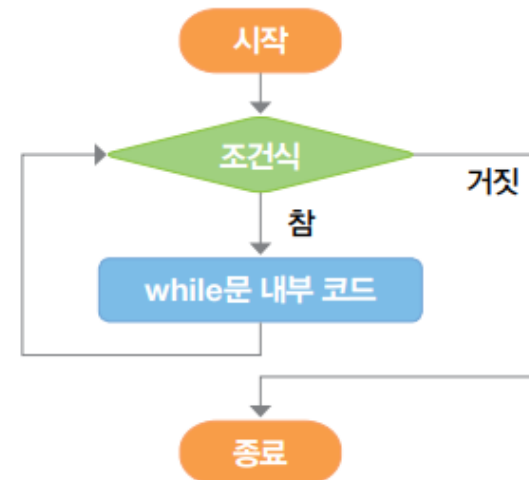
■ while문

- 조건식이 참이면 루프를 계속 반복하는 반복문
 - 조건식은 대개 비교 또는 논리 연산식
- while문은 반복 횟수가 정해져 있지 않을 때 사용함

```
while (조건식) {  
    // 실행 코드  
}
```



[그림 4-17] while문의 실행 흐름



[그림 4-18] while문의 순서도

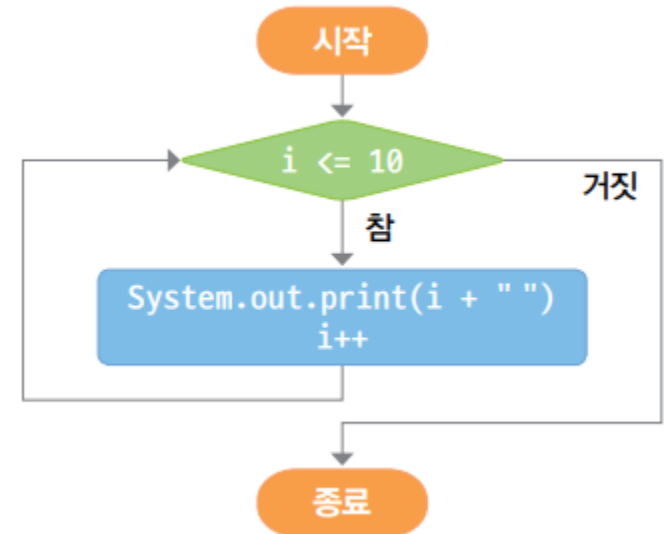
3. 반복문

while문 사용 예시

```
public class Example08 {  
    public static void main(String[] args) {  
        int i = 1;  
        while (i <= 10) {  
            System.out.print(i + " ");  
            i++;  
        }  
    }  
}
```

실행 결과

1 2 3 4 5 6 7 8 9 10



[그림 4-19] Example08.java 프로그램의 while문 처리 과정

3. 반복문

예제 4-7 짝수와 홀수 판별 지속 여부 확인하기

```
01 import java.util.Scanner;
02
03 public class Loop02 {
04     public static void main(String[] args) {
05         Scanner s = new Scanner(System.in);
06         int choice = 1;
07         while ( choice == 1 ) {
08             int a;
09
10             System.out.println("숫자를 입력하세요.");
11             a = s.nextInt();
12
13             if (a % 2 == 0) {
14                 System.out.println("짝수입니다.");
15             } else {
16                 System.out.println("홀수입니다.");
17             }
18         }
19     }
20 }
```

3. 반복문

예제 4-7 짝수와 홀수 판별 지속 여부 확인하기

```
18
19     System.out.println("계속하고 싶다면 예 1, 그만하고 싶다면 아니요 0을
    입력하세요.")
20
21     choice = s.nextInt();
22 }
23
24 System.out.println("모든 숫자를 확인했습니다.");
25 }
26 }
```

실행 결과

숫자를 입력하세요.

20

짝수입니다.

계속하고 싶다면 예 1, 그만하고 싶다면 아니요 0을 입력하세요.

1

숫자를 입력하세요.

21

홀수입니다.

계속하고 싶다면 예 1, 그만하고 싶다면 아니요 0을 입력하세요.

0

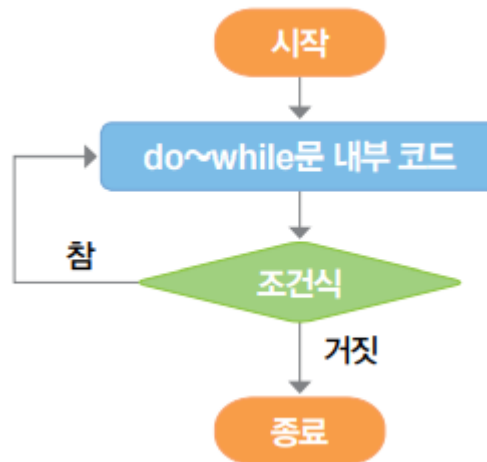
모든 숫자를 확인했습니다.

3. 반복문

■ do~while문

- while문과 for문처럼 조건식에 의해 반복 실행된다는 점은 비슷함
- 하지만 조건식을 확인하기 전에 코드 블록의 코드가 한 번 실행된다는 점이 다름

```
do {  
    // 실행 코드  
} while (조건식);
```



[그림 4-20] do~while문의 순서도

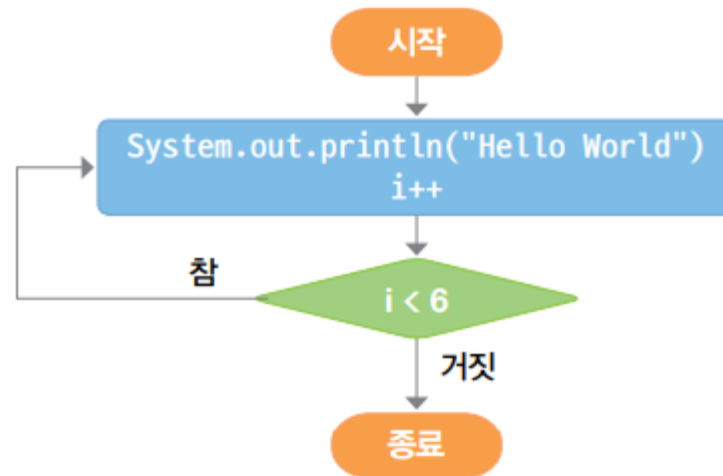
3. 반복문

do~while문 사용 예시

```
public class Example09 {  
    public static void main(String[] args) {  
        int i = 1;  
        do {  
            System.out.println("Hello World");  
            i++;  
        } while ( i < 6 );  
    }  
}
```

실행 결과

Hello World
Hello World
Hello World
Hello World
Hello World



[그림 4-21] Example09.java 프로그램의 do~while문 처리 과정

3. 반복문

예제 4-8 10보다 큰 수부터 10까지의 합 구하기

```
01 import java.util.Scanner;
02
03 public class Loop03 {
04     public static void main(String[] args) {
05         Scanner s = new Scanner(System.in);
06         System.out.println("10보다 큰 숫자를 입력하세요.");
07         int x = s.nextInt();
08         int sum = 0;
09
10         do {
11             sum += x;
12             x--;
13
14         } while (x >= 10);
15         System.out.println("합계 : " + sum);
16     }
17 }
```

실행 결과

10보다 큰 숫자를 입력하세요.

20

합계 : 165

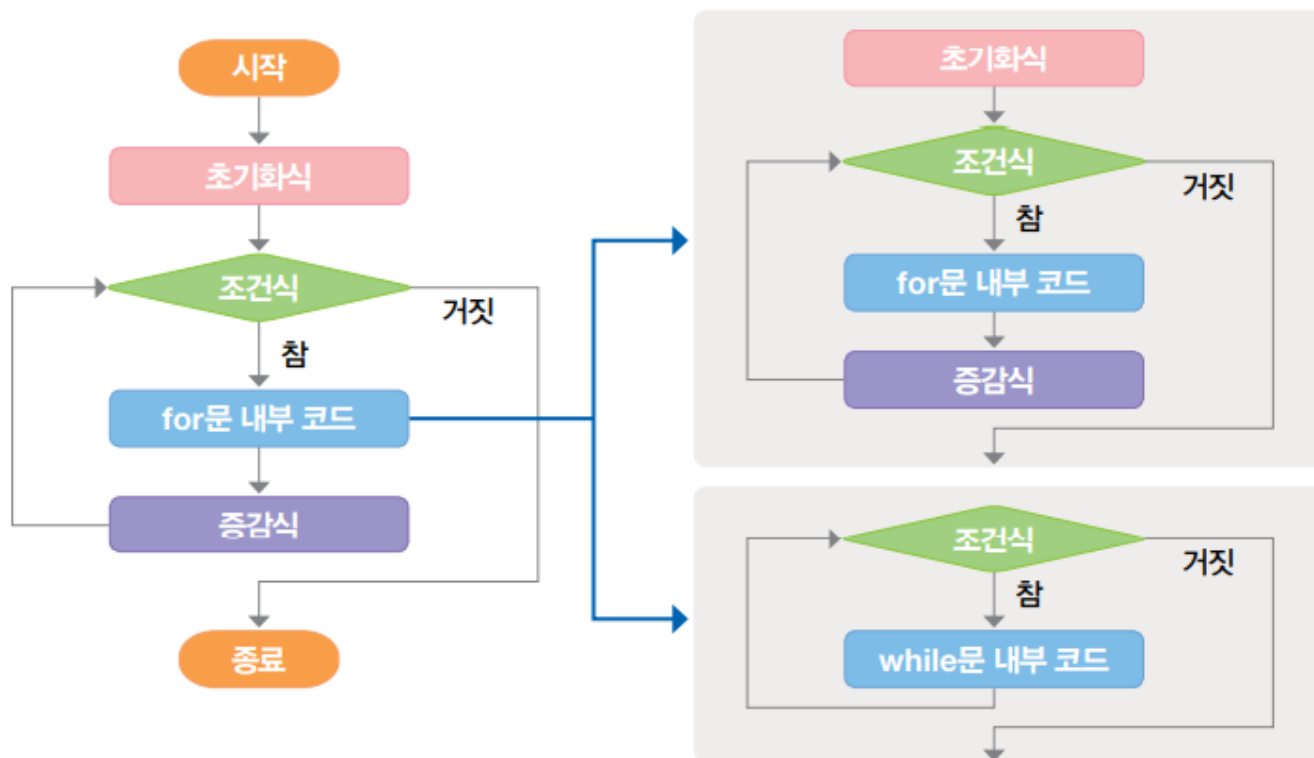
3. 반복문

■ 중첩 반복문

- 반복문 안에 또 다른 반복문을 원하는 만큼 포함한 것
- 외부 루프의 첫 번째 반복이 시작된 후 내부 루프가 시작됨
 - 내부 루프가 반복되고 종료되자마자 외부 루프의 첫 번째 반복이 완료됨
 - 두 번째 반복으로 이동
 - 가장 바깥쪽 루프의 반복을 마칠 때까지 계속 반복됨

3. 반복문

■ 중첩 반복문



[그림 4-22] 중첩 반복문의 순서도

3. 반복문

중첩 반복문 사용 예시

```
public class Example10 {  
    public static void main(String[] args) {  
        for (int i = 2; i <= 4; i++) {  
            System.out.println("구구단 " + i + " 단");  
            for (int j = 1; j <= 10; j++) {  
                System.out.println(i + " * " + j + " = " + (i * j));  
            }  
        }  
    }  
}
```

실행 결과

구구단 2 단

2 * 1 = 2

2 * 2 = 4

2 * 3 = 6

...

구구단 3 단

3 * 1 = 3

3 * 2 = 6

3 * 3 = 9

...

구구단 4 단

4 * 1 = 4

4 * 2 = 8

4 * 3 = 12

...

4 * 10 = 40

3. 반복문

예제 4-9 별표로 역삼각형 만들기

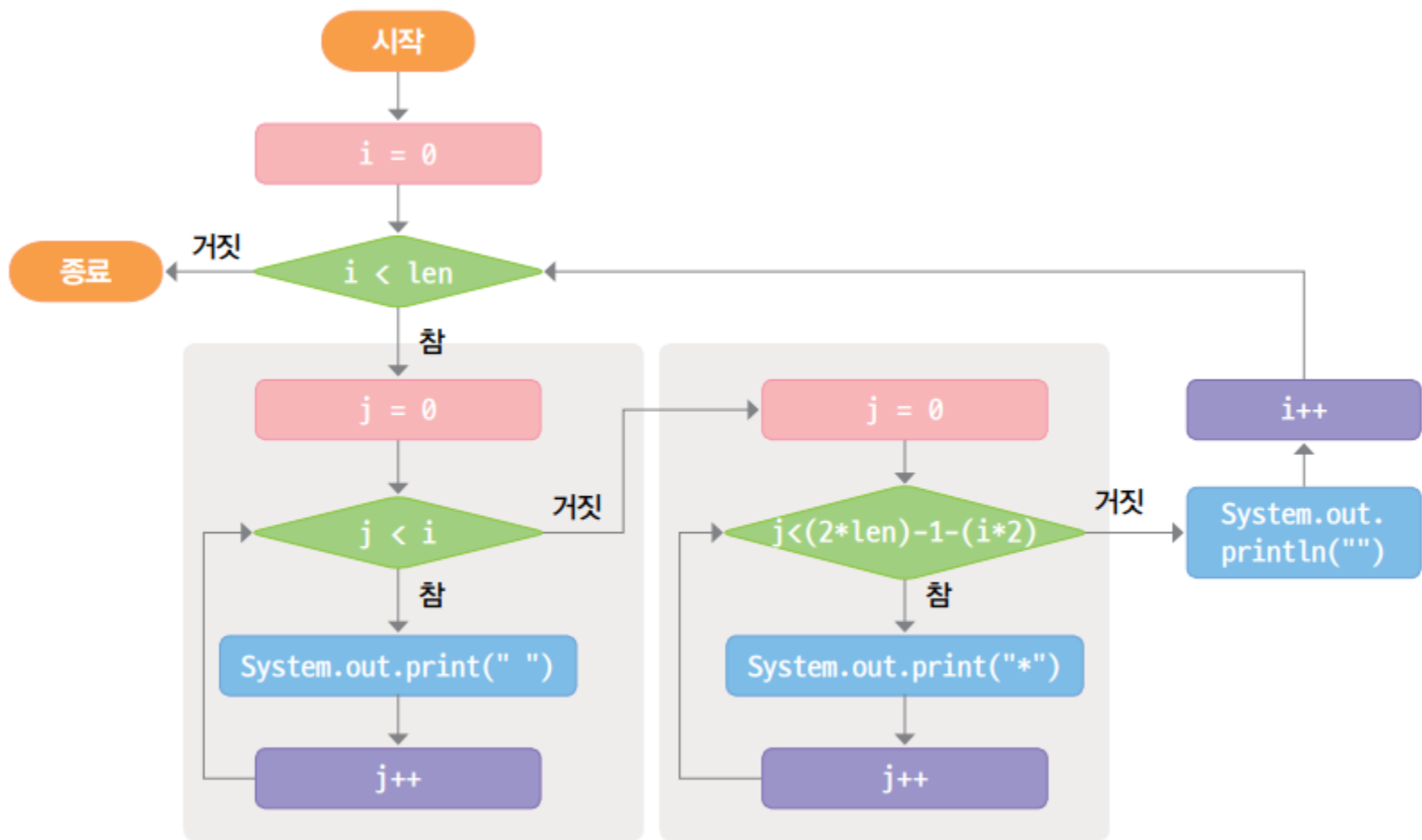
```
01 import java.util.Scanner;
02
03 public class Loop04 {
04     public static void main(String[] args) {
05         Scanner s = new Scanner(System.in);
06         System.out.print("길이 : ");
07
08         int len = s.nextInt();
09
10         for (int i = 0; i < len; i++) {
11             for (int j = 0; j < i; j++) {
12                 System.out.print(" ");
13             }
14             for (int j = 0; j < (2*len)-1-(i*2); j++) {
15                 System.out.print("*");
16             }
17             System.out.println("");
18         }
19     }
20 }
```

실행 결과

길이 : 5

*

3. 반복문



[그림 4-23] Loop04.java 프로그램의 중첩 반복문 실행 흐름

Section 04

분기문

4. 분기문

■ 분기문의 개요

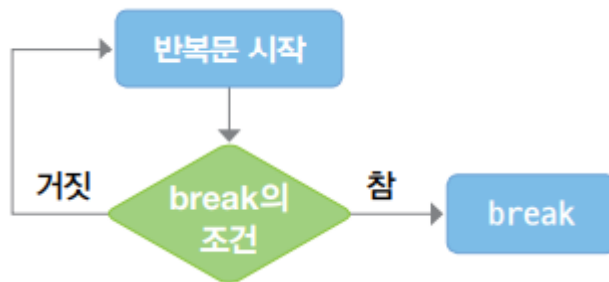
- 분기문
 - 원할 때마다 반복문에서 벗어나거나 반복문을 시작하도록 실행 흐름을 직접 제어할 수 있는 제어문
 - break문, continue문, return문

4. 분기문

■ break문

- 원할 때마다 반복문의 실행을 중지하거나 종료할 수 있음
- Break문은 for문, while문, do~while문 에서 모두 사용됨

```
break;
```



[그림 4-24] break문의 순서도

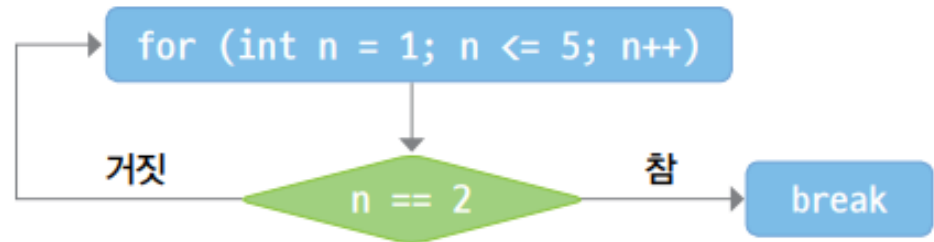
4. 분기문

break문 사용 예시

```
public class Example11 {  
    public static void main(String[] args) {  
  
        for (int n = 1; n <= 5; n++) {  
            System.out.println("*");  
            if (n == 2) {  
                break;  
            }  
        }  
    }  
}
```

실행 결과

*
*



[그림 4-25] break문의 순서도

4. 분기문

예제 4-10 i가 5가 되면 반복문 탈출하기

```
01 public class Break01 {  
02     public static void main(String[] args) {  
03         for (int i = 0; i < 10; i++) {  
04             if (i == 5)  
05                 break;  
06             System.out.println("i : " + i);  
07         }  
08     }  
09 }
```

실행 결과

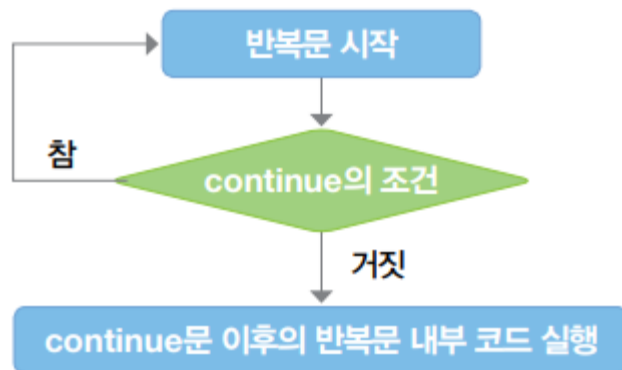
```
i : 0  
i : 1  
i : 2  
i : 3  
i : 4
```

4. 분기문

■ continue문

- 특정 조건을 건너뛰고 나머지를 계속 실행하려는 경우에 사용함
- 반복문의 블록 내부에서 for문의 증감식 또는 while문의 조건식으로 이동하는 역할

```
continue;
```



[그림 4-25] continue문의 순서도

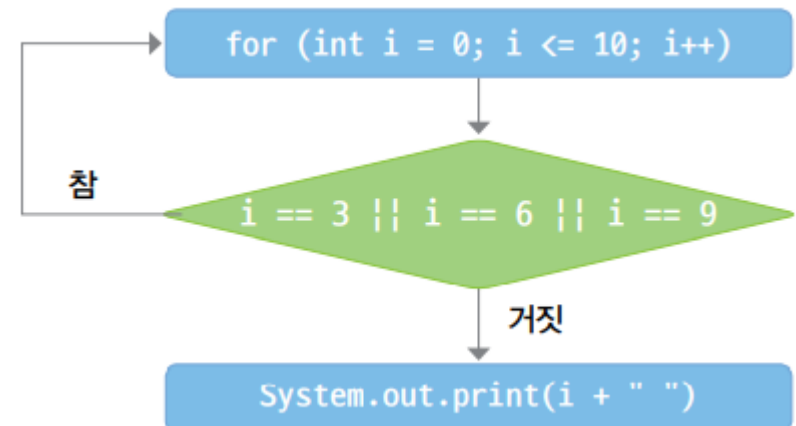
4. 분기문

continue문 사용 예시

```
public class Example12 {  
    public static void main(String[] args) {  
  
        for (int i = 0; i <= 10; i++) {  
            if (i == 3 || i == 6 || i == 9) {  
                continue;  
            }  
            System.out.print(i + " ");  
        }  
    }  
}
```

실행 결과

0 1 2 4 5 7 8 10



[그림 4-27] continue문의 순서도

4. 분기문

예제 4-11 3×2이면 건너뛰기

```
01 public class Continue01 {  
02     public static void main(String[] args) {  
03         for (int i = 1; i <= 4; i++) {  
04             for (int j = 1; j <= 3; j++) {  
05                 if (i == 3 && j == 2) {  
06                     continue;  
07                 }  
08                 System.out.println(i + " * " + j);  
09             }  
10         }  
11     }  
12 }
```

실행 결과

1 * 1
1 * 2
1 * 3
2 * 1
2 * 2
2 * 3
3 * 1
3 * 3
4 * 1
4 * 2
4 * 3

i가 3, j가 2이면 출력되지 않음

[프로젝트]

메뉴 정보 표시 및 종료하기

메뉴 정보 표시 및 종료하기

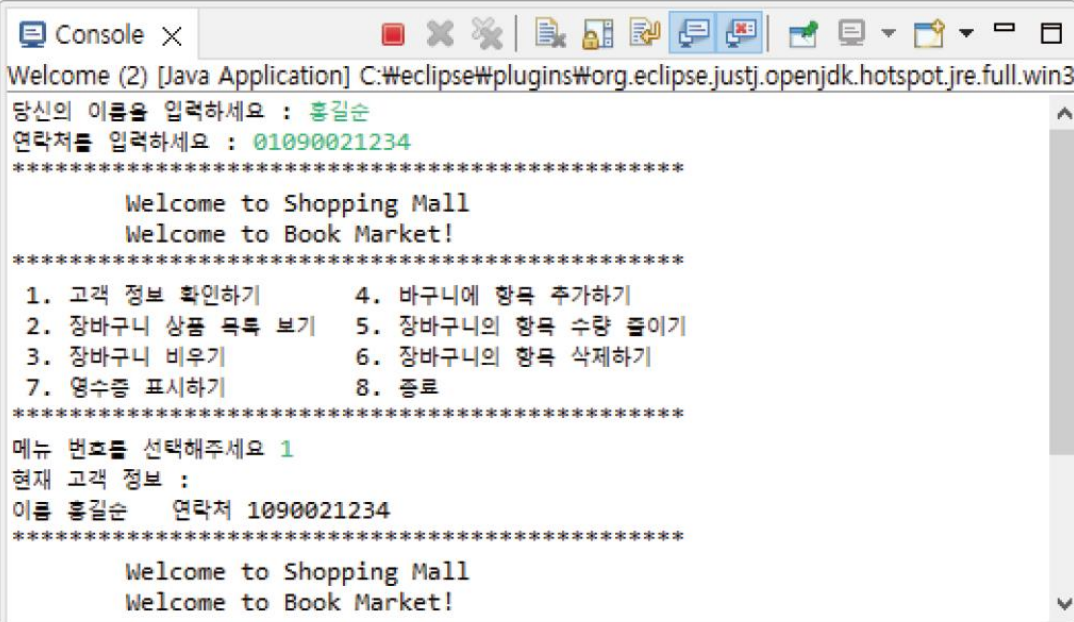
- 제어문을 이용하여 선택 가능한 메뉴 정보를 출력하고, 반복문을 이용하여 종료 메뉴를 선택하기 전까지 메뉴를 계속 선택할 수 있게 합니다



[그림 4-28] 메뉴 정보 표시 및 종료하기

메뉴 정보 표시 및 종료하기

- 제어문을 이용하여 선택 가능한 메뉴 정보를 출력하고, 반복문을 이용하여 종료 메뉴를 선택하기 전까지 메뉴를 계속 선택할 수 있게 합니다



```
Console X
Welcome (2) [Java Application] C:\Weclipse\plugins\Worg.eclipse.justj.openjdk.hotspot.jre.full.win3
당신의 이름을 입력하세요 : 홍길순
연락처를 입력하세요 : 01090021234
*****
Welcome to Shopping Mall
Welcome to Book Market!
*****
1. 고객 정보 확인하기      4. 바구니에 항목 추가하기
2. 장바구니 상품 목록 보기  5. 장바구니의 항목 수량 줄이기
3. 장바구니 비우기        6. 장바구니의 항목 삭제하기
7. 영수증 표시하기        8. 종료
*****
메뉴 번호를 선택해주세요 1
현재 고객 정보 :
이름 홍길순   연락처 1090021234
*****
Welcome to Shopping Mall
Welcome to Book Market!
```

[그림 4-29] 메뉴 정보 표시 및 종료하기 실행 결과

메뉴 정보 표시 및 종료하기

■ 메뉴 정보 표시하기

- 01 메뉴 정보 출력하기

- Welcome.java 파일을 열고, 메뉴 번호를 선택하면 메뉴 정보를 출력하는 코드를 추가함

- 02 고객 정보 출력하기

- [프로젝트4-1]의 case 1: 내부에 코드를 한 줄 추가함

- ✓ '메뉴 번호 1'의 경우 입력받은 고객 정보를 출력하기 위함

메뉴 정보 표시 및 종료하기

■ 메뉴 선택 종료하기

- 01 메뉴 선택 계속하기

→ 선택할 메뉴를 보여주는 코드를 감싸도록 반복문을 작성