4장. 제어문

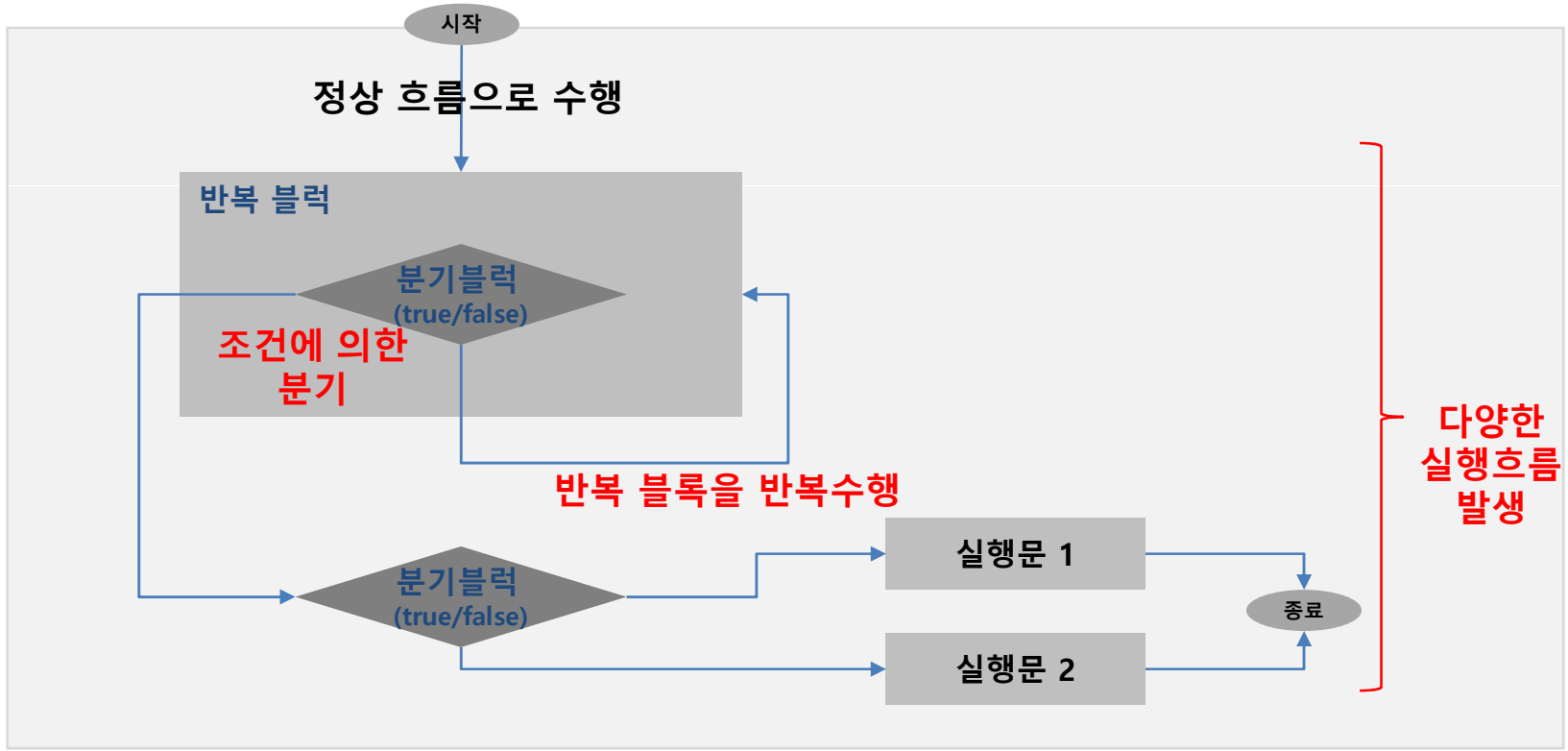
(조건문 / 반복문)

4.1. 코드 실행 흐름 이해

- Java 프로그램은 Main 메서드에서 시작하여 순차적으로 실행됨
- 프로그램 제어란 반복문을 통해서 반복을 하고, 분기문을 통해서 분기를 수행하는 것을 의미

코드 실행 흐름

```
public class Human {
  public static void main (String [] args) {
  }
}
```

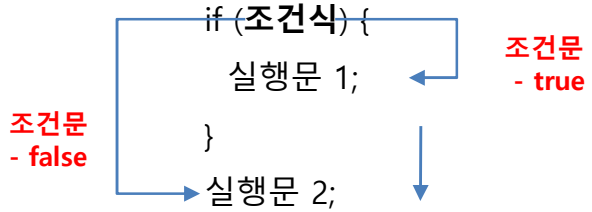


4.2.1. 조건문 (if 문)

- 프로그램의 분기는 조건문의 true, false 의 결과에 따라서 이루어짐.
- 분기문은 중괄호 ({ / })로 이루어진 부분을 시작과 끝으로 인식함.

조건문 (if 문)

프로그램 문법



Case 1 : 조건문이 true일 경우
- 조건문의 중괄호 블록 안에 있는 실행문 1 수행함.
- 그 후 실행문 2가 수행됨

Case 2 : 조건문이 false일 경우
- 실행문 2수행
- 조건의 충족되지 않으므로 실행문 1은 건너뜀

프로그램 사례

예제) 점수가 60점 이상일 경우 합격 판정하는 프로그램

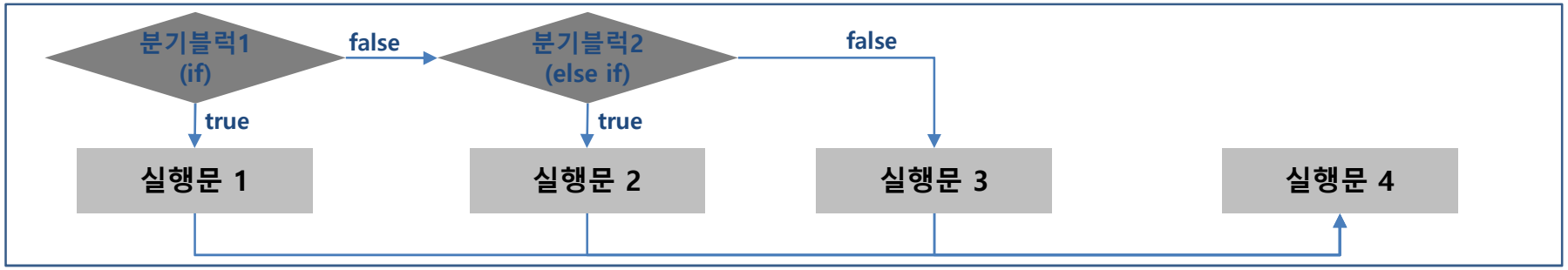
```
public class Human {
  public static void main (String [] args) {
    int kor = 80;
    System.out.println("점수= " + kor);

    if (kor >= 60) {
      System.out.println("점수가 60점보다 높으므로
                          합격입니다.");
    }
    System.out.println("프로그램을 종료합니다.");
```

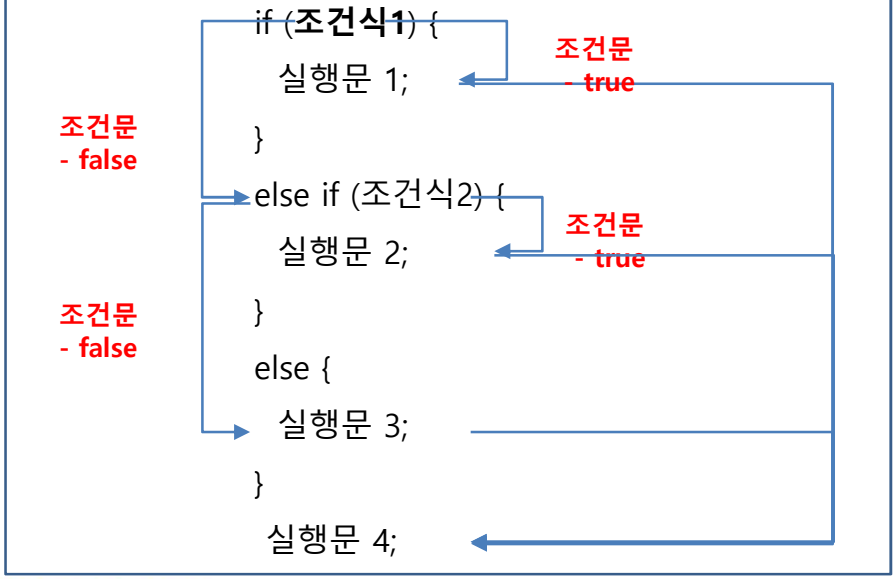
4.2.2. 조건문(if ~ else문)

- 프로그램의 분기는 조건문의 true,false 의 결과에 따라서 이루어짐.
- else는 조건문이 false일 경우 수행되는 제어문임

조건문 (if ~ else문)



프로그램 문법



프로그램 사례

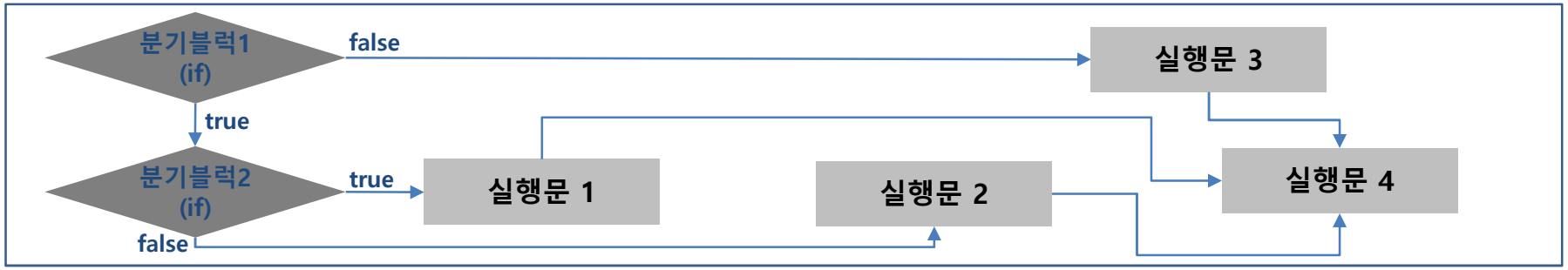
예제) 점수가 90이상이면 A, 80이상이면 B, 그 외는 C학점

```
public class Human {
  public static void main (String [] args) {
    int kor = 80;
    if (kor >= 90) {
      System.out.println("A학점입니다.");
    }
    else if (kor >= 80) {
      System.out.println("B학점입니다.");
    }
    else {
      System.out.println("C학점입니다.");
    }
    System.out.println("프로그램을 종료합니다.");
```

4.2.3. 조건문(중첩 if 문)

- 프로그램의 분기는 조건문의 true,false 의 결과에 따라서 이루어짐.
- if 문 안에 또 다른 if문이 있을 수 있으며, 이 역시 true/false에 따라서 분기가 이루어짐

조건문 (중첩 if 문)



프로그램 문법

조건문 - false

조건문 - true

조건문 - false

조건문 - true

```
if (조건식1) {
  if (조건식2) {
    실행문 1;
  }
  else {
    실행문 2;
  }
}
else {
  실행문 3;
}
실행문 4;
```

프로그램 사례

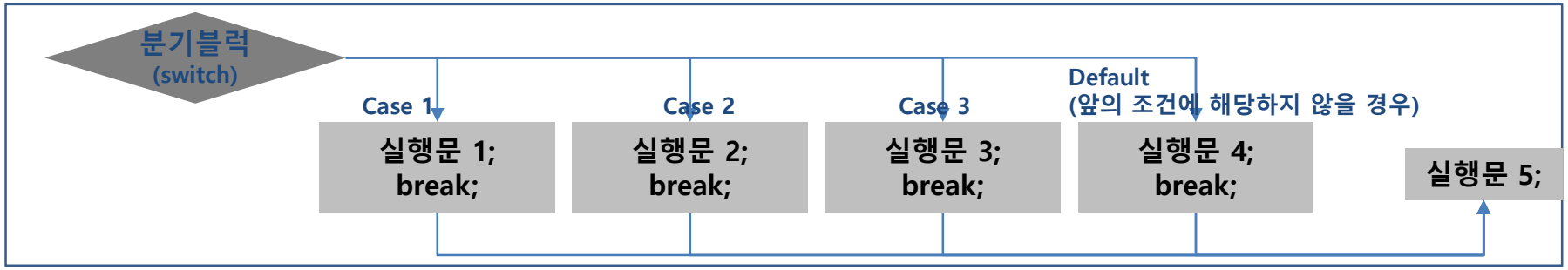
예제) 점수가 60이상이면 합격, 그 중에서도 90점 이상일 경우 장학생여부 판별하는 프로그램

```
public class Human {
  public static void main (String [] args) {
    int kor = 80;
    if (kor >= 60) {
      System.out.println("합격입니다..");
      if (kor >90) {
        System.out.println("당신은 장학생입니다.");
      }
      else {
        System.out.println("장학생이 되지 못한 것이 아쉽습니다.");
      }
    }
    System.out.println("프로그램을 종료합니다.");
}
```

4.2.4. 조건문(swich ~ case 문)

- 분기를 위한 조건이 특정 수치로 정의될 경우 switch ~ case 문을 사용함.
- case 문 처리 이후 break로써 분기문을 마무리 지어야 함.

조건문 (switch ~ case 문)



프로그램 문법

```
switch(변수) {
  case (값1) :
    실행문 1;
    break;
  case (값2) :
    실행문 2;
    break;
  case (값3) :
    실행문 3;
    break;
  default :
    실행문 4;
    break;
}
실행문 5;
```

변수 = 값1

변수 = 값2

변수 = 값3

변수 의 값이 위의 조건에 해당하지 않을때

열거형의 데이터 타입 변수

Break문으로 통해 Switch 조건문 빠져나옴

주의사항 :
각 case별 break문이 생략될 경우 하위로 조건을 추가적으로 탐색함.
그리고 default도 수행됨.

프로그램 사례

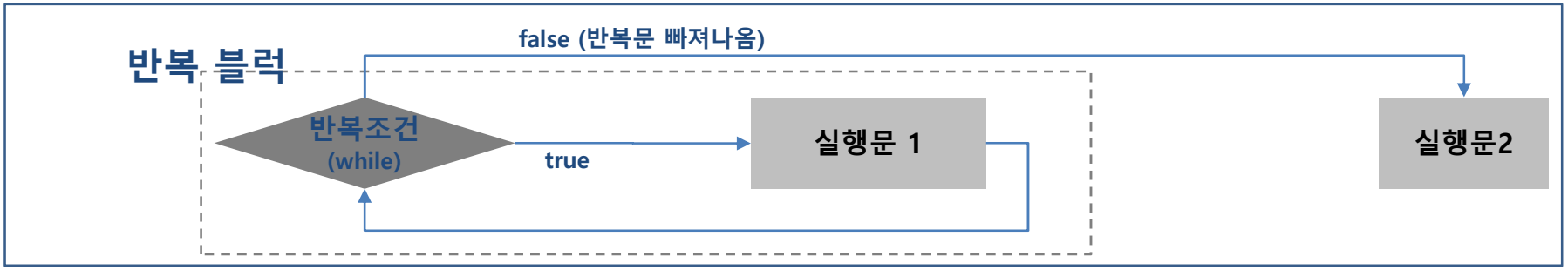
예제) 회원 grade별로 회원을 표시하는 프로그램.

```
public class Human {
  public static void main (String [] args) {
    char  grade = 'A';
    switch (grade) {
      case 'A' :
        System.out.println("우수회원입니다."); break;
      case 'B' :
        System.out.println("일반회원입니다."); break;
      default :
        System.out.println("비회원입니다."); break;
    }
    System.out.println("프로그램을 종료합니다.");
  }
}
```

4.3.1. 반복문(while 문)

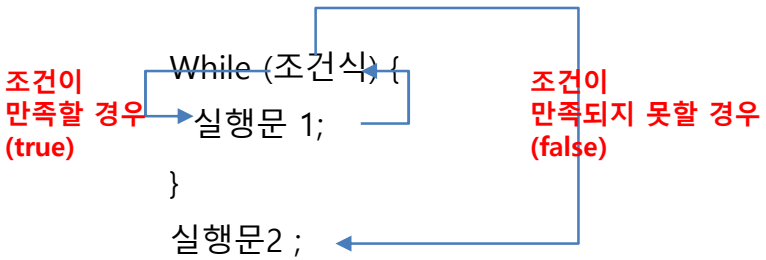
- 반복문은 반복조건이 true일 동안 반복블럭내에서 지속 반복함.
- 반복조건이 false일 경우 반복블럭을 빠져나와 다음 프로그램을 수행함.

조건문 (while 문)



프로그램 문법

반복의 조건을 확인하는 조건문에 의해 반복적으로 실행문을 실행하는 것을 의미한다.



프로그램 사례

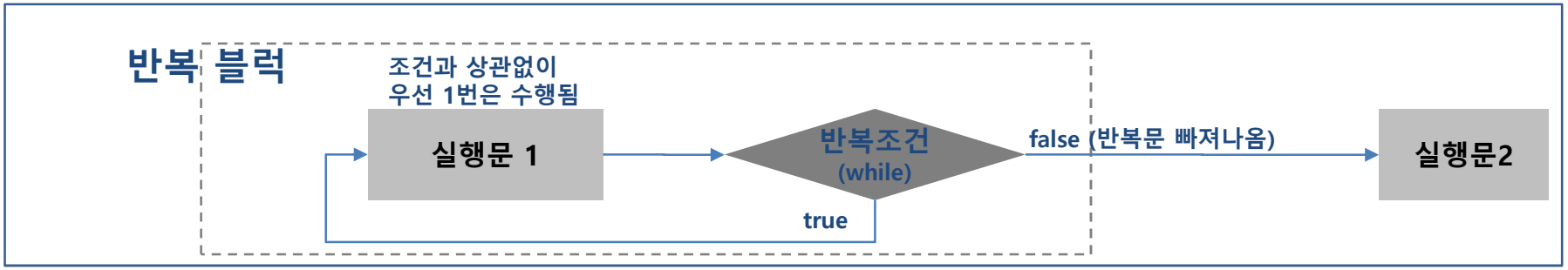
예제) 1~10까지의 누적합산을 처리하는 프로그램

```
public class Human {
  public static void main (String [] args) {
    int total = 0;                  // 누적값을 저장하는 변수
    int index = 1;                  // 반복 인덱스를 저장하는 변수
    while (index <= 10) {
      total = total + index;
      index = index + 1;
    }
    System.out.println("1~10까지의 합산은 = " + total);
  }
}
```

4.3.2. 반복문(do~while 문)

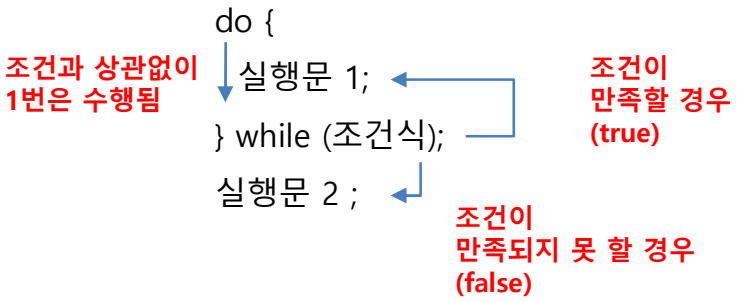
- 반복문은 반복조건이 true일 동안 반복블럭내에서 지속 반복함.
- do 문장에 의해서 우선 한번은 실행한 후 반복은 추후 확인함

조건문 (while 문)



프로그램 문법

while 문과 do~while의 차이점은 do에 의해서 우선 한번은 실행된다는 차이가 있다.
즉, do~while문은 반복의 조건문이 블록 뒤에 있음



프로그램 사례

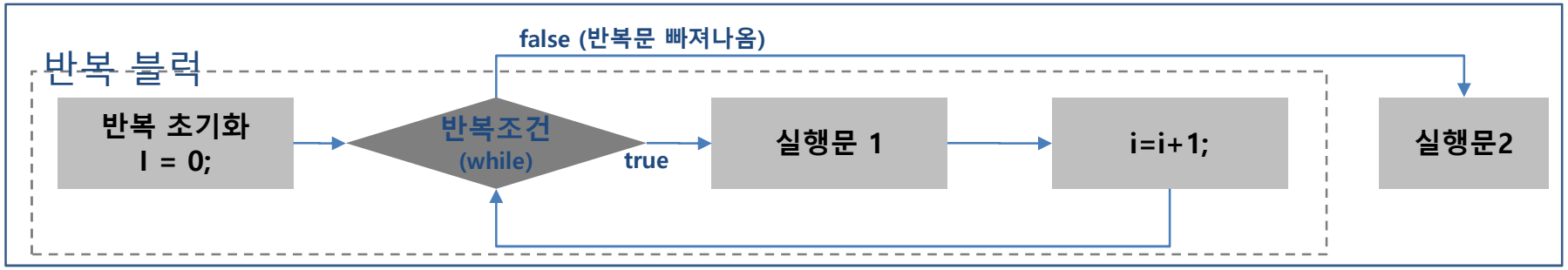
예제) 1~10까지의 누적합산을 처리하는 프로그램

```
public class Human {
  public static void main (String [] args) {
    int total = 0;              // 누적값을 저장하는 변수
    int index = 0;              // 반복 인덱스를 저장하는 변수
    do {
      index = index + 1;
      total = total + index;
    } while (index < 10) ;
    System.out.println("1~10까지의 합산은 = " + total);
  }
}
```

4.3.3. 반복문(for 문)

- 반복문은 반복조건이 true일 동안 반복블럭내에서 지속 반복함.
- 반복조건이 false일 경우 반복블럭을 빠져나와 다음 프로그램을 수행함.

조건문 (for 문)



프로그램 문법

반복문은 반복 index만큼 반복하면서 실행문을 실행하는 일들이 많음. 따라서 이와 같은 경우는 아래의 순서를 따르며, for문을 수행하는 적이 많음

1. 반복 index 초기화
2. 반복 조건문 설정
3. 반복 index의 증감

```
for (index= 0 ; index < 10 ; index=index+1) {
     // 반복 인덱스 초기화
     // 반복의 조건 수행
     // 반복 인덱스의 증감
  실행문 1;
}
실행문 2;
```

프로그램 사례

예제) 1~10까지의 누적합산을 처리하는 프로그램

```
public class Human {
  public static void main (String [] args) {
    int total = 0;                  // 누적값을 저장하는 변수
    for (int index = 1; index <= 10 ; index=index+1) {
      total = total + index;
    }
    System.out.println("1~10까지의 합산은 = ", total);
  }
}
```