인터넷 프로그래밍

1. 제어문

예제 if2.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int a=10, b=20;

  if(a >= b)
      out.print("a가 b보다 크다. <BR>");  //조건이 참이면 이 문장이
수행된다.
  else
      out.print("a가 b보다 작다. <BR>");  //조건이 거짓이면 이 문장이
수행된다.

  out.print("if문을 벗어났습니다.");
%>
```

실행결과 : a가 b보다 작다.
if문을 벗어났습니다.

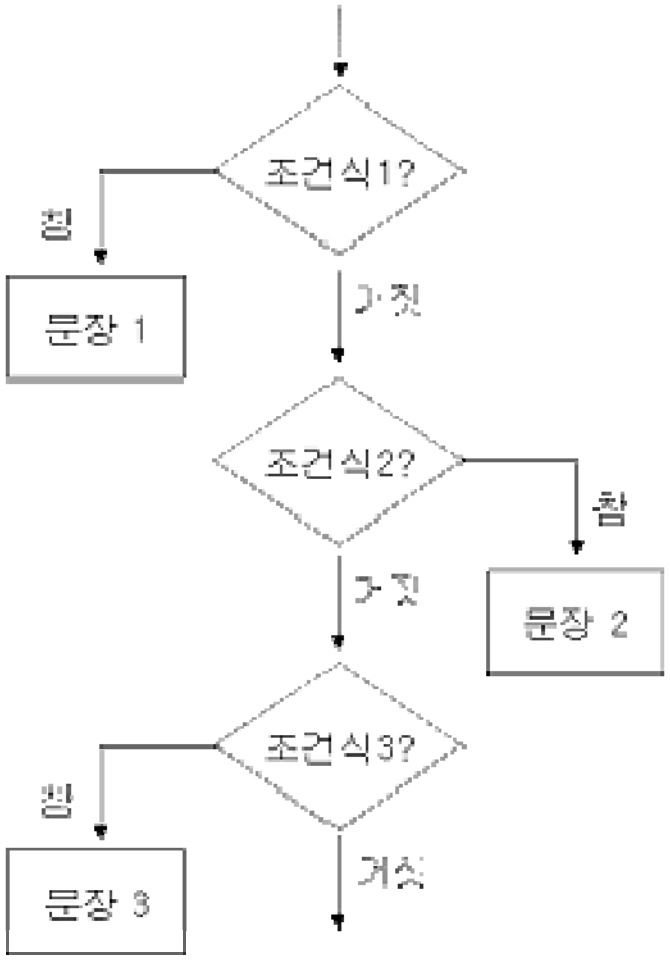
1. 제어문

사용 형식 3 : if ~ else if 구문

```
if (조건식1)
   문1;
else if(조건식2)
   문2;
else if(조건식3)
   문3;
else if(조건식4)
   문4;
```

또는

문장이 두 줄 이상일 때는 마찬가지로 중괄호를 이용해 묶는다



1. 제어문

예제 if3.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int score = 83, i;
  i = 83/10;

  if(i==10) {
    out.print("만점입니다. <BR>");
    out.print("수고하셨습니다. <BR>");
  } else if(i==9) {
    out.print("90점대 입니다 <BR>");
    out.print("조금만 더 노력하세요. <BR>");
  } else if(i==8) {
    out.print("80점대 입니다. <BR>");
    out.print("열심히 하세요. <BR>");
  } else {
    out.print("80점대 미만입니다. <BR>");
    out.print("많이 노력하세요. <BR>");
  }
%>
```

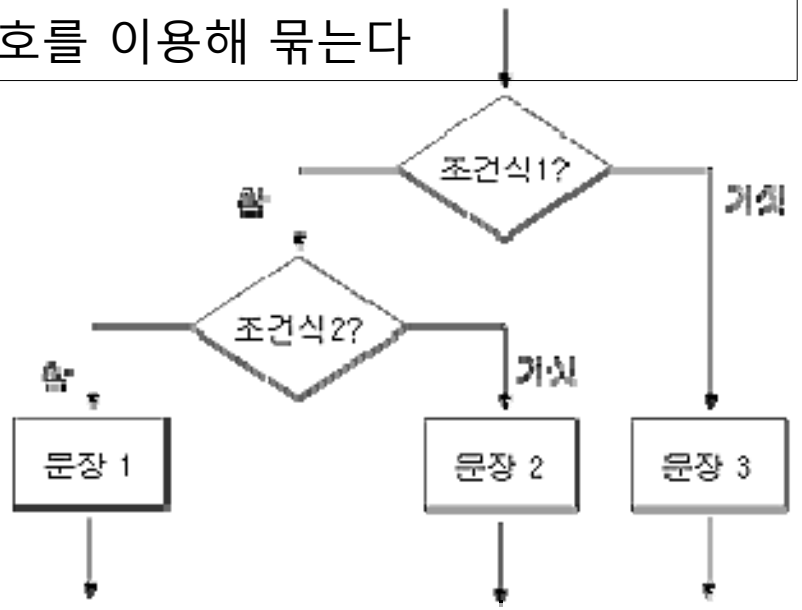
실행결과 : 80점대입니다.
열심히 하세요.

1. 제어문

사용 형식 4 : 중첩된 if구문

```
if (조건식1)
   if (조건식2)
        문장1;  //조건식1과 조건식2가 모두 참이면 이 문장을 수행
   else
        문장2;  //조건식1은 참이지만 조건식2가 거짓이면 이 문장을 수행
else
   문장3;       //조건식1이 거짓이면 이 문장을 수행
```

또는
문장이 두 줄이상일 때는 마찬가지로 중괄호를 이용해 묶는다



1. 제어문

예제 if4.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int score = 70;

  if(score >= 70) {
    if(score == 100) {
                out.print("합격했습니다. <BR>");
                out.print("만점입니다. <BR>");
               } else {
                out.print("합격했습니다. <BR>");
                out.print("하지만, 만점은 아닙니다. <BR>");
               }
  } else {
    out.print("불합격했습니다. <BR>");
  }
%>
```

실행결과 : 합격했습니다.
하지만, 만점은 아닙니다.

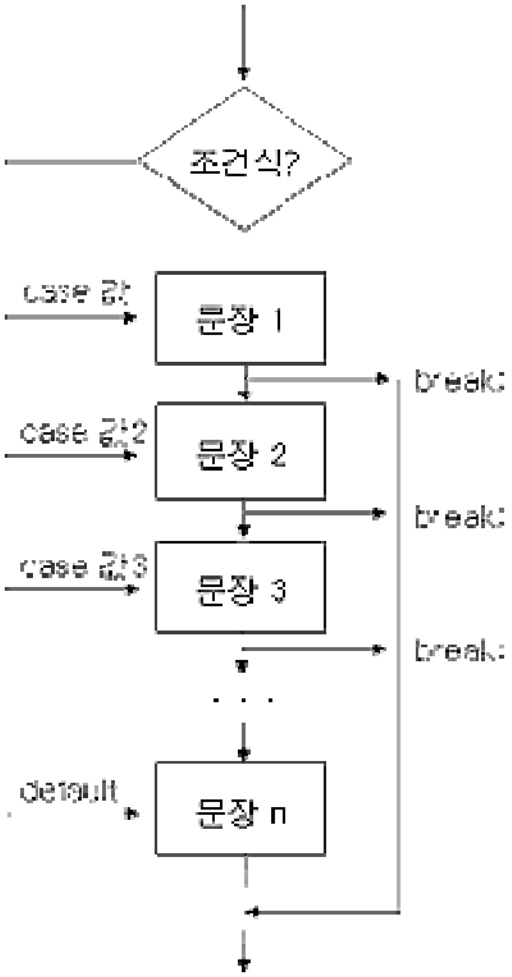
1. 제어문

1.2 switch 문

- 다중 선택을 하기 위한 구문으로 switch 문의 조건에서 쓰일 수 있는 자료형은 byte, short, char, int 유형만이 가능하다.

사용 형식 : switch 구문

```
switch (표현식) {
        case 표현식1:
                문장1;
                …
                break;
        case 표현식2:
                문장2;
                …
                break;
                …
               default:   // 만족하는 조건이 없을시 수행된다.
                문장3;
                …
}
```



1. 제어문

실행결과 : 점수는 : 96이고 학점은 : A

예제 switch1.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%
int  score = 96;
char grade;
switch (score/10) {
  case 10:  //case문에 break;를 쓰지 않는다면 아래의 case 구문까지 수행된다.
  case  9:
      grade = 'A';
      out.println("점수는 : " + score + " 이고 학점은 : " + grade);
      break;
  case  8:
      grade = 'B';
      out.println("점수는 : " + score + " 이고 학점은 : " + grade);
      break;
  case  7:
      grade = 'C';
      out.println("점수는 : " + score + " 이고 학점은 : " + grade);
      break;
  case  6:
      grade = 'D';
      out.println("점수는 : " + score + " 이고 학점은 : " + grade);
      break;
  default:
      grade = 'F';
      out.println("점수는 : " + score + " 이고 학점은 : " + grade);
}
%>
```

1. 제어문

- 만일 몇 가지 경우를 동일하게 처리하려면 break문 없는 case문만을 작성하고 다음에 나오는 case문에 원하는 작업과 break문을 기술한다

예제 switch2.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int year = 5;

  switch(year) {
    case 1:
    case 2:
    case 3:
       out.print("저학년이군요");
       break;
    case 4:
    case 5:
    case 6:
       out.print("고학년이군요");
       break;
  }
%>
```

실행결과 : 고학년이군요

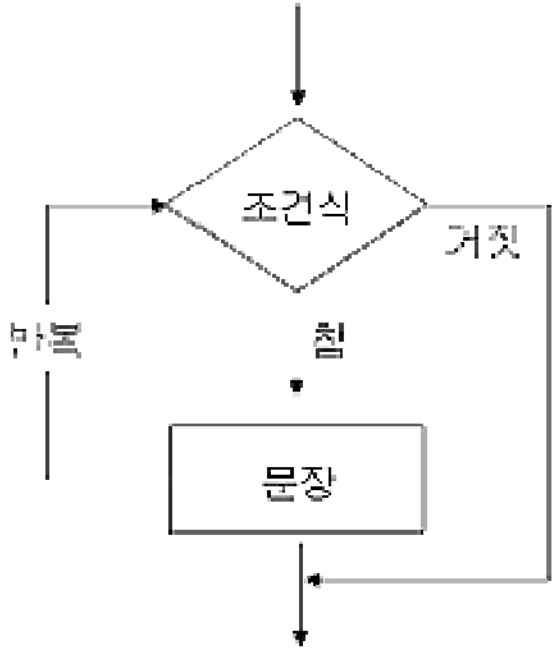
1. 제어문

1.3 while 문

- 조건식이 참인 동안 문장을 반복 수행한다.

사용 형식 : while 구문

while (조건식) 문장;
 	
또는
 	
while (조건식) {
  문장1;
  문장2;
  ...
}



1. 제어문

예제 while.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%
  int i, sum;
  i = sum = 0;

  while(i < 10) {
    i++;
    sum = sum + i;
      out.print("반복횟수 : " + i + "  지금까지의 합 : " + sum +
"<BR>");
  }
%>
```

실행결과 : 반복횟수 : 1 지금까지의 합 : 1
반복횟수 : 2 지금까지의 합 : 3
반복횟수 : 3 지금까지의 합 : 6
반복횟수 : 4 지금까지의 합 : 10
반복횟수 : 5 지금까지의 합 : 15
반복횟수 : 6 지금까지의 합 : 21
반복횟수 : 7 지금까지의 합 : 28
반복횟수 : 8 지금까지의 합 : 36
반복횟수 : 9 지금까지의 합 : 45
반복횟수 : 10 지금까지의 합 : 55

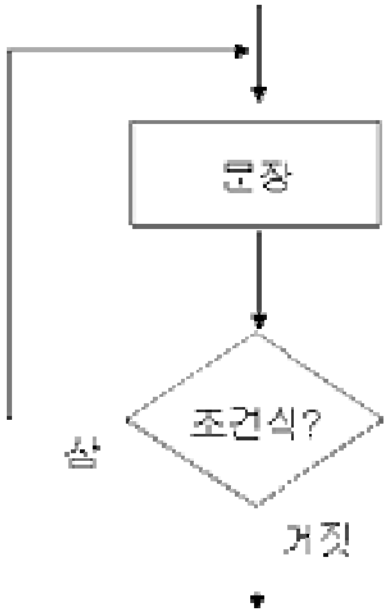
1. 제어문

1.4 do ~ while 문

- while 문과 마찬가지로 반복을 수행한다. while 문과의 차이점은 조건식의 판단이 나중에 이루어 진다는 것으로 조건식의 결과와 상관 없이 무조건 한번은 반복문이 실행된다.

사용 형식 : do ~ while 구문

do 문장; while (조건식);
 	
또는
 	
do {
  문장1;
  문장2;
  …
} while (조건식);



1. 제어문

예제 dowhile.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int i, sum;
  i = sum = 0;

  do {
    // 이 블록은 무조건 실행된다.
    i++;
    sum = sum + i;

    out.print("반복횟수 : " + i + "  지금까지의 합 : " + sum + "<BR>");
  } while(i > 10);  // 조건식을 비교해서 참일 경우만 반복을 계속 수행한다.
%>
```

실행결과 : 반복횟수 :1 지금까지의 합 :1

주의 : while문과 do~while문 사용시 조건식에서 사용되는 변수의 증감을 해주지 않으면 무한 반복에 빠질 수 있으므로 주의해야 한다.

1. 제어문

예제 errorwhile.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int i, sum;
  i = sum = 0;

  while(i < 10) {
    // 변수 i를 증가시키는 구문이 제외되면 무한 반복에 빠지게 되므로 주의한다.
    // i++;
    sum = sum + i;

    out.print("반복횟수 : " + i + "  지금까지의 합 : " + sum + "<BR>");
  }
%>
```

실행 결과 : 반복횟수 : 0 지금까지의 합 : 0
반복횟수 : 0 지금까지의 합 : 0
반복횟수 : 0 지금까지의 합 : 0
.
.
.
반복횟수 : 0 지금까지의 합 : 0
반복횟수 : 0 지금까지의 합 : 0
반복횟수 : 0 지금까지의 합 : 0

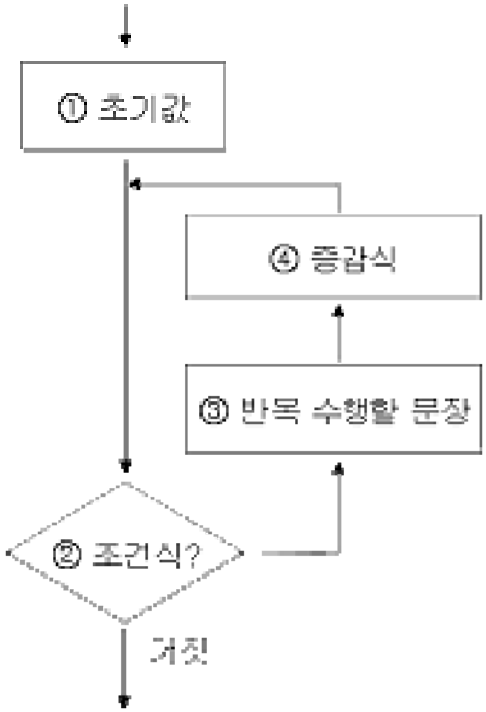
1. 제어문

1..5 for 문

- 초기값, 조건식, 증감식을 한꺼번에 모두 지정할 수 있는 반복문이다.

사용 형식 : for 구문

for(초기값; 조건식; 증감식) 문장;
 	
또는
 	
for(초기값; 조건식; 증감식) {
  문장1 ;
  문장2 ;
  ...
}



1. 제어문

- for문은 다음 단계로 동작한다.

① 반복 수행을 처음 시작할 때 변수를 초기화한다. 이 단계는 한번만 실행된다.
② 조건식을 검사하고 조건식이 참이면,
③ 반복을 수행하고 거짓이면 for문을 종료한다.
④ 변수에 저장된 값을 지정한 증감식에 따라서 값을 증감한 후, 단계 ②로 간다.

1. 제어문

예제 for1.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int i;

  for(i=10; i>0; i--)
    out.print("i의 값은 : " + i +"<BR>");
%>
```

실행결과 : i의 값은 : 10
i의 값은 : 9
i의 값은 : 8
i의 값은 : 7
i의 값은 : 6
i의 값은 : 5
i의 값은 : 4
i의 값은 : 3
i의 값은 : 2
i의 값은 : 1

1. 제어문

- 다음 예제와 같이 초기값에 사용되는 변수는 for문내에서도 선언될 수 있으며 이 변수는 for문내에서만 사용 가능한 지역 변수가 된다. 또한, 다른 제어문이나 반복문들과 마찬가지로 for문도 중첩해서 사용할 수 있다

예제 for2.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  for(int i=1; i<=10; i++) {
    for(int j=1; j<=i; j++) {
      out.print(" * ");
    }
    out.print("<BR>");
  }
%>
```

실행결과 : *

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

참고 : while, do~while 문을 무한 반복 하려면 while(true)처럼 조건에 참값인 'true' 값을 넣으면 된다.
그리고 for 문은 for(;;) 와 같이 for 문에 세미콜론(;)만 연속해서 두 번 넣으면 된다.

1. 제어문

- 다음 형식은 반복문의 무한 반복형식이다.

```
while(true) {
  문장1;
  문장2;
}

do {
  문장1;
  문장2;
} while(true);

for(;;) {
  문장1;
  문장2;
}
```

주의 : 무한 반복에서 벗어날 수 있는 구문(break 문)이 반복문내에 존재해야 한다.

1. 제어문

예제 loop.jsp

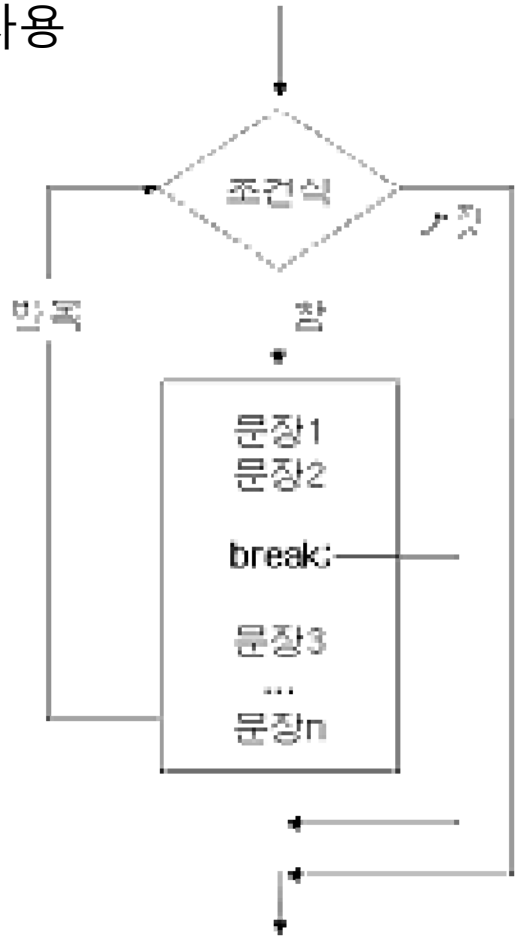
```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int i=0;
  //while 조건문에 'true' 값을 넣으면 무한 반복한다.
  while(true) {
    i++;

    //무한 반복을 종료하기 위해 break문을 이용한다.
    if(i>10)
      break;

    out.print(i + "번 반복합니다.<BR>");
  }
%>
```

실행결과 : 1번 반복합니다.
2번 반복합니다.
3번 반복합니다.
4번 반복합니다.
5번 반복합니다.
6번 반복합니다.
7번 반복합니다.
8번 반복합니다.
9번 반복합니다.
10번 반복합니다.

1. 제어문

1.6 break 문

- break 문은 다음의 3가지 기능을 수행하며 자신이 속한 반복문만 벗어난다.
  ① switch 문을 벗어나는데 사용
  ② 반복문(while, do~while, for)을 벗어나는데 사용



1. 제어문

예제 break.jsp

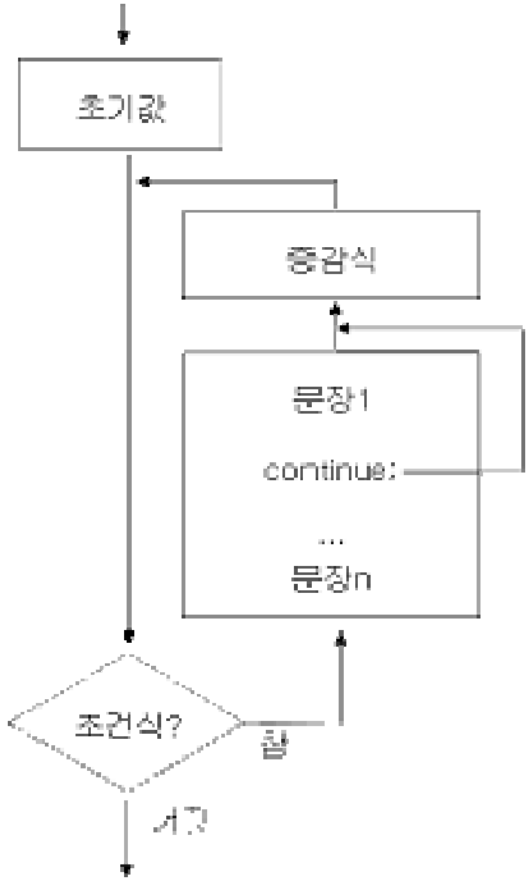
```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  int i=0;

  //while의 조건문에 'true' 값을 넣어 무한 반복한다.
  while(true) {
    //만약 변수 i의 값이 10이면 break문에 의해 for문을 벗어난다.
    if(i==10)
      break;

    i++;

    out.print(i + "번 수행 <BR>");
  }

  out.print("while문을 벗어났습니다.<BR>");
%>
```

실행결과 : 1번 수행
2번 수행
3번 수행
4번 수행
5번 수행
6번 수행
7번 수행
8번 수행
9번 수행
10번 수행
While문을 벗어났습니다.

1. 제어문

1.7 continue 문

- 반복문(while, do~while, for 문)에서 사용하며 다음 그림과 같이 반복문 내에서 continue 문을 만나면 수행을 중지하고 반복문의 처음으로 이동한다.



1. 제어문

예제 continue.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  for(int i=0; i<=10; i++) {
    /* i를 2로 나누었을 때 나머지가 0이면 즉, 짝수이면 continue 구문 이후 문장을
       수행하지 않고 증감한 후에 제어를 다시 for 문으로 옮긴다. */
    if(i%2 == 0)
      continue;

    out.print(i + "<BR>");
  }
%>
```

실행결과 : 1
3
5
7
9

2. 배열

- 배열은 동일한 자료형의 데이터를 연속적으로 저장하고 사용하기 위한 자료형으로 배열을 사용하기 위해서 우선 다음과 같이 선언을 해야 한다. 또한 배열의 선언시에는 배열 크기를 지정하지 않는다.

사용 형식 : 배열 선언하기
배열에 저장될 자료형 배열이름[ ] ; 또는 배열에 저장될 자료형[ ] 배열이름;

2. 배열

예제 array1.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%!
  //정수형의 number 배열을 선언한다.
  int number[];
  //문자열형의 name 배열을 선언한다.
  String[] name;
%>
```

참고 : 자바에서는 배열도 객체로 다루기 때문에 객체를 생성하는 방법과 동일하게 new 연산자를 사용하여 배열의 메모리를 생성한다. 배열의 메모리를 생성하는 방법은 배열을 선언한 후 메모리를 할당하거나, 또는 배열의 선언과 동시에 메모리를 할당할 수도 있다.

2. 배열

사용 형식 : 배열 객체 생성하기
배열이름=new 배열에 저장 될 자료형[배열크기]; 또는 배열에 저장 될 자료형 배열이름[ ]=new 배열에 저장 될 자료형[배열크기];

예제 array2.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%!
  //정수형의 num 배열을 선언하고 배열의 크기가 10인 배열 객체를 생성한다.
  int num[] = new int[10];

  //문자열형의 name 배열을 선언한다.
  String[] name;
%>

<%
  //선언된 name 배열에 크기가 5인 배열 객체를 생성한다.
  name = new String[5];
%>
```

2. 배열

- 배열은 배열의 각 요소들을 가리키는 정수 값인 "인덱스"를 통해 배열의 값들에 접근 할 수 있다.

- 인덱스는 0부터 배열크기-1 범위까지의 정수를 사용할 수 있다.

- 배열의 각 요소를 특정 값으로 초기화하는 방법은 다음과 같다.

① 배열 선언시 초기화
② 배열의 객체를 생성하는 과정에서 배열을 초기화
③ 배열 객체의 생성 후 배열의 각 요소에 값을 대입하여 초기화

사용 형식 : 배열의 초기화 방법
배열에 저장될 자료형 배열이름[ ]={값리스트}; 또는 배열이름=new 배열에 저장될 자료형[ ] {값리스트};

2. 배열

예제 array3.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%
  //배열을 선언하면서 1부터 10까지의 값으로 초기화
  int number[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

  //배열의 선언, 객체 생성 그리고 초기화
  String[] season = new String[] {"봄","여름","가을","겨울"};

  //배열의 객체를 생성한 후 인덱스를 이용하여 각 배열 요소에 값을 대입
  String[] fruit = new String[3];
  fruit[0] = "바나나";
  fruit[1] = "사과";
  fruit[2] = "키위";
  out.print("좋아하는 숫자는 : " + number[8] + "<BR>");
  out.print("좋아하는 계절은 : " + season[0] + "<BR>");
  out.print("좋아하는 과일은 : " + fruit[2] + "<BR>");

  //배열의 길이는 length 속성을 이용하여 구할 수 있다.
  out.print("배열 number[]의 길이는 : " + number.length);
%>
```

실행결과 :
좋아하는 숫자는 : 9
좋아하는 계절은 : 봄
좋아하는 과일은 : 키위
배열 number[]의 길이는 : 10

주의 : 배열 선언시 배열의 초기값을 지정한 경우에는 new 연산자 없이 자동으로 메모리가 할당된다.
초기화 값의 수에 따라 배열의 크기가 자동으로 결정되기 때문에 배열의 크기를 넣지 않아도 된다.

2. 배열

- 다차원 배열을 선언하는 방법은 기본적으로 일차원 배열을 선언하는 방법과 같으며 원하는 차원 수만큼의 배열 기호([ ])를 추가한다.

사용 형식 : 다차원 배열의 선언
배열에 저장될 자료형 배열이름[ ][ ] ; 또는 배열에 저장될 자료형[ ][ ] 배열이름;

사용 형식 : 다차원 배열 객체 생성
배열이름=new 배열에 저장될 자료형[배열크기][배열크기]; 또는 배열에 저장될 자료형 배열이름[ ][ ]=new 배열에 저장될 자료형[배열크기][배열크기];

2. 배열

예제 array4.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<%
  // 3행 2열의 이차원 배열 생성
  String[][] list = new String[3][2];

  list[0][0] = "홍길동";
  list[0][1] = "hong@mail.net";

  list[1][0] = "이순신";
  list[1][1] = "lee@sun.net";

  list[2][0] = "강감찬";
  list[2][1] = "kang@abc.co.kr";

  out.print(list[0][0] + "의 메일주소는 : " + list[0][1] + "<BR>");
  out.print(list[1][0] + "의 메일주소는 : " + list[1][1] + "<BR>");
  out.print(list[2][0] + "의 메일주소는 : " + list[2][1] + "<BR>");
%>
```

실행결과 : 홍길동의 메일주소는 : hong@mail.net
이순신의 메일주소는 : lee@sun.net
강감찬의 메일주소는 : kang@abc.co.kr

2. 배열

- 만약, 다차원 배열을 선언시에 초기화 시키려면 다음과 같이 중괄호({})를 사용하면 된다. 또한, 일차원 배열과 동일하게 초기화 값의 수에 따라 행과 열의 크기가 자동으로 결정되기 때문에 배열의 행과 열을 넣을 필요가 없다.

예제 array5.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%
  // 3행 2열의 이차원 배열을 선언시 초기화
  String[][] list = {
    {"홍길동", "hong@mail.net"},
    {"이순신", "lee@sun.net"},
    {"강감찬", "kang@art.co.kr"}
  };

  out.print(list[0][0] + "의 메일주소는 : " + list[0][1] + "<BR>");
  out.print(list[1][0] + "의 메일주소는 : " + list[1][1] + "<BR>");
  out.print(list[2][0] + "의 메일주소는 : " + list[2][1] + "<BR>");
%>
```

실행결과 : 홍길동의 메일주소는 : hong@mail.net
이순신의 메일주소는 : lee@sun.net
강감찬의 메일주소는 : kang@abc.co.kr