Chapter 6. 제어구조 : Control Structures

제어 구조는 프로그램이 수행되는데 가장 필수적인 조건 분기, 반복 처리 등의 코드 블록입니다.

1. if 문

if 문은 “만약 조건이 맞으면 A 코드 블록을 실행하고, 아니면 B 코드 블록을 실행” 하는 기능을 합니다.

1.1 조건이 참일때만 처리하는 if 문

다음은 조건이 참인 경우만을 처리하기 위해 사용하는 구조입니다. 「조건식_A」 문이 참이면 「문장_1」 이 처리됩니다. 참일 경우 처리되는 문장이 여러 개일 경우에는 ② 방식으로 작성합니다만 한개일 경우에도 사용합니다.

①	②

if (조건식_A)
　　문장_1;
	if (조건식_A) {
	　　문장_1;
	　　문장_2;
	}

if.php		결과
1	$point_A = 95;	
2	$point_B = 88;	
3	if ($point_A >= 90)	
4	echo "A 학점";	A 학점
5	if ($point_B < 90) {	
6	echo "B 학점";	B 학점
7	echo "열심히";	열심히
8	}	

1.2 조건이 참, 거짓일 때 처리하는 if 문 : if ~ else

조건이 참, 거짓일 때 각각을 별도로 처리하기 위해서 if ~ else 문을 사용합니다.

```
if ( 조건식_A ) {
    문장_1;
} else {
    문장_2;
}
```

ifelse_1.php		결과
1	$point = 88;	
2	if ($point >= 90) {	
3	echo "A 학점"; // 90 이상이면 수행	B 학점
4	} else echo {	
5	echo "B 학점"; // 90 미만이면 수행	
6	}	

1.3 조건식이 여러 개일 때 처리하는 if 문 : if ~ else if, if ~ elseif

조건식들을 여러 개 연결해서 처리하기 위한 구조입니다. 첫 번째 if 문의 조건식이 거짓일 때 추가 조건식을 위해 else if 문을 이용합니다. else if는 elseif 처럼 붙여서 사용해도 동일하게 기능합니다.

①	②

if (조건식_A) {
  문장_1;
} else if (조건식_B) {
  {
  문장_2;
  }
  : :
  } else {
  문장_n;
  }
	if (조건식_A) {
	  문장_1;
	} elseif (조건식_B) {
	  {
	  문장_2;
	  }
	  : :
	  } else {
	  문장_n;
	  }

	ifelse_2.php	결과
1	$point = 88;	B 학점
2	if ($point >= 90) {	
3	echo "A 학점"; // 90 이상이면 수행	
4	} elseif ($point >= 80) {	
5	echo "B 학점"; // 80 이상이고 90 미만이면 수행	
6	} else {	
7	echo "C 학점"; // 80 미만이면 수행	
8	}	

1.4 중첩 if 문

조건식들을 여러 개 연결해서 처리하기 위한 구조입니다. 첫 번째 if 문의 조건식이 true일 때 추가 조건식을 위해 중첩 if 문을 사용합니다.

```
if ( 조건식_A ) {
  if ( 조건식_B ) {
  문장_1;
  } else {
  문장_2;
  }
else {
}
```

ifelse_3.php		결과

01
02
03
04
05
06
07
08
09
10
11
	$a = 75;
	$b = 88;
	if ($a >= 70) {
	  if ($b >= 80){
	  echo "2 과목 통과";
	  } else {
	  echo "1 과목 통과";
	  }
	} else {
	  echo "통과 실패";
	}
		2 과목 통과

1.5 조건식의 형태

논리 연산자를 이용하여 여러 개의 조건식을 하나의 조건식으로 표현할 수 있습니다. 예를 들어 80 이상인 조건과 90 미만인 조건은 다음과 같이 하나의 조건식으로 표현됩니다.

if ($point < 90)) {
  if ($point >= 80)
}
	➡	if (($point < 90) && ($point >= 80))

if_logical.php		결과

1
2
3
4
	$point = 88;
	if (($point < 90) && ($point >= 80)){
	  echo "B 학점";
	}
		B 학점

2. while, do ~ while 문을 이용한 반복 수행

while 문과 do ~ while 문은 코드 블록을 반복 수행하기 위한 루프 구조입니다.

2.1 while 문

조건식이 true일 동안 코드 블록을 반복적으로 수행합니다. while 문은 조건식을 먼저 연산하기 때문에 flase면 코드 블록을 수행하지 않고 빠져 나갑니다.

일반적으로 ① 형식처럼 코드 블록을 { .. } 기호로 감싸서 사용합니다만, ② 형식처럼 while (조건식); ~~ endwhile; 문을 사용할 수도 있습니다.

①	②

while (조건식) {
  코드 블록
}
	while (조건식):
	  코드 블록
	endwhile;

다음 2개의 코드는 동일하게 수행됩니다. 조건식이 true 이라면 코드 블록이 수행되고 조건식이 false면 while 문을 빠져 나갑니다.

while_1.php		while_2.php		결과

1
2
3
4
	$i = 1;
	while ($i <= 10) {
	  echo $i++;
	}
		1
		2
		3
		4
			$i = 1;
			while ($i <= 10):
			  echo $i++;
			endwhile;
				12345678910

2.2 do ~ while 문

조건식이 true일 동안 코드 블록을 반복적으로 수행합니다. while 문은 조건식을 나중에 연산하기 때문에 코드 블록은 최소한 한 번은 수행됩니다.

```
do {
  코드 블록
} while (조건식);
```

dowhile_1.php 코드는 코드 블록을 먼저 수행후 while 문의 조건식이 연산됩니다. dowhile_2.php 코드는 do ~ while 문이 무한 반복처리되는 것을 방지하기 위해 break 문을 사용한 경우입니다.

dowhile_1.php		결과	dowhile_2.php		결과

1
2
3
4
	$i = 1;
	do {
	  echo $i++;
	} while ($i > 0);
		무한
		반복
			1
			2
			3
			4
			5
				$i = 1;
				do {
				  echo $i++;
				  if ($i > 5) break;
				} while ($i <= 10);
					1
					2
					3
					4
					5

3. for 문을 이용한 반복 처리

3.1 단일 for 문

for 문은 코드 블록을 반복 처리하기 위한 루프 구조입니다.

일반적으로 ① 형식처럼 코드 블록을 { .. } 기호로 감싸서 사용하지만 ② 형식처럼 for(): ~~ endfor; 문을 사용할 수도 있습니다.

①	②

for (초기값; 조건식; 증가값) {
  코드 블록
}
	for (초기값; 조건식; 증가값):
	  코드 블록
	endfor;

for_1.php 코드는 조건식이 true일 때까지 코드 블록이 수행되며, for_2.php는 증가값을 2씩 증가시키는 경우입니다.

for_1.php		for_2.php	

1
2
3
	for ($i=1; $i<=10; $i++) {
	  echo $i;
	}
		1
		2
		3
			for ($i=1; $i<=10; $i=$i+2):
			  echo $i;
			endfor;

for_3.php 코드는 for 문을 종료할 조건식이 없는 경우이며, break 문을 이용하여 종료할 수 있습니다.

for_4.php 코드는 코드 블록을 무한 반복으로 수행하는 for 문이며 break 문을 이용하여 종료할 수 있습니다.

for_3.php

```
1  for ($i=1; ; $i++) {
2      if ($i > 10) break;
3      echo $i;
4  }
```

for_4.php

```
1  $i=1;
2  for (; ;) {
3      if ($i > 10) break;
4          echo $i;
5      $i++;
6  }
```

3.2 중첩 for 문

for 문 내부에 for 문을 삽입하여 중첩된 for 문을 구성할 수 있습니다. 중첩 for 문은 다음과 같이 2차원 형태의 데이터를 처리하는데 사용됩니다. 물론 3중첩 이상의 for 문 구조도 가능합니다.(배열 참조)

과목 / 학생번호	국어	영어	수학
1	75	82	85
2	81	78	90
:			
10	88	83	76

이 표의 데이터 값이 score 배열 변수에 저장되었다고 가정할 때 학생번호별 총점를 구하는 코드는 for_5.php와 같습니다. 또한 과목별 총점을 구하는 코드 for_6.php와 같습니다.

for_5.php

```
for ($i=0; $i<10; $i++) {
  sum[$i] = 0;
  if ($j=0; $j<3; $j++) {
    sum[$i] = sum[$i] + score[$i][$j];
  }
}
```

for_6.php

```
for ($i=0; $i<3; $i++) {
  sum[$i] = 0;
  if ($j=0; $j<10; $j++) {
    sum[$i] = sum[$i] + score[$i][$j];
  }
}
```

4. foreach 문

foreach 문은 배열이나 객체와 같이 하나의 변수명에 복수의 값를 저장하는 구조에 사용됩니다. 배열과 객체에 있는 많은 값들의 추출은 반복적인 작업이며, 이때 foreach 문을 사용합니다. 배열 원소 각각에 대해 for 문을 수행한다고 이해하면 됩니다.
foreach 문은 다음과 같이 2가지 문법이 있습니다.

①	foreach (반복구조 as $value) {
	  처리 문장
	}

②	foreach (반복구조 as $key => $value) {
	  처리 문장
	}

설명	- 반복구조 : 배열명(객체명) - $key : 인덱스 값이 저장되는 변수명 - $value : 각 항목의 값이 저장되는 변수명

①, ② 문법의 차이는 인덱스를 코드에서 사용할 지의 여부입니다. 즉 인덱스를 코드에서 사용 ② 문법을 사용합니다.
주의할 점은 처리 문장에서 각 원소의 값을 변경해도 원래 배열에는 기본적으로 반영되지 않습니다. 처리 문장을 통해 각 원소의 값도 변경하려면 &(주소) 기호를 이용해야 합니다.

4.1 숫자 인덱스 배열 처리

다음 코드는 $num_arr 배열의 각 원소 값을 $value 변수에 저장하여 2를 곱하는 예제입니다.

foreach_1.php		결과
1	$num_arr = array(3, 5, 7);	
2	foreach ($num_arr as $value) {	
3	$value = $value * 2;	
4	echo "$value ";	6 10 14
5	}	
6	print_r($num_arr);	Array ([0] => 3 [1] => 5 [2] => 7)

- 2~5번 라인 : $num_arr 배열의 각 원소의 값을 $value 변수에 저장하여 2를 곱함
- 6번 라인 : 배열의 정보 확인. 원래 배열의 데이터 값에 변경 없음

다음 코드는 주소를 이용하여 원래 배열의 원소 값을 변경하는 예제입니다.

foreach_2.php		결과
1	$num_arr = array(3, 5, 7);	
2	foreach ($num_arr as &$value) {	
3	$value = $value * 2;	
4	echo "$value ";	6 10 14
5	}	
6	print_r($num_arr);	Array ([0] => 6 [1] => 10 [2] => 14)

- 2번 라인 : & 기호를 사용하여 $value 변수의 주소를 이용.
- 6번 라인 : 배열의 정보 확인. $value 변수의 주소에 2를 곱했기 때문에 원래 배열의 데이터 값이 변경됨.

다음 코드는 ② 문법을 이용하여 인덱스를 코드에서 사용하는 예제입니다.

foreach_3.php		결과
1	$num_arr = array(3, 5, 7);	
2	foreach ($num_arr as $key => $value) {	0 => 3
3	echo "{$key} => {$value} ";	1 => 5
4	}	2 => 7

- 2번 라인 : 인덱스 키 값을 $key 변수에 저장
- 3번 라인 : $key 변수의 값 사용

4.2 연관 배열 처리

연관 배열의 키 값은 문자열인 배열입니다.
다음 코드는 연관 배열의 각 원소 값을 출력하는 예제입니다.

foreach_4.php		결과
1	$car_arr = array('Company'=>'Hyundai', 'car'=>'Santefe', 'year'=>'2022');	
2	foreach ($car_arr as $value) {	Hyundai
3	echo "{$value} ";	Santefe
4	}	2022

다음 코드는 연관 배열의 각 키 값과 원소 값을 출력하는 예제입니다.

foreach_5.php		결과
1	$car_arr = array('Company'=>'Hyundai', 'car'=>'Santefe', 'year'=>'2022');	
2	foreach ($car_arr as $key => $value) {	Company => Hyundai
3	echo "{$key} => {$value} ";	car => Santefe
4	}	year => 2022

5. switch ~ case 문

switch ~ case 문은 중첩된 if 문처럼 동작합니다. 다만 조건식의 true, false 값으로 코드 블록을 수행하지 않고 여러 개의 특정 값(1, 2, ..)에 따라 코드 블록을 수행합니다.
식의 값이 값_1에 해당되면 코드 블록_A를 수행하고 switch 문을 종료합니다. 값_1, 값_2, .. 등 어떤 값에도 해당되지 않으면 default 문의 코드 블록_N을 수행하고 종료합니다. 각 case 구조에 break 문이 없으면 아래쪽의 case 문이 실행된다는 점을 주의합니다. default 문을 생략해도 됩니다.

```
switch (식) {
    case 값_1:
        코드 블록_A;
        break;
    case 값_2:
        코드 블록_B;
        break;
    case 값_3:
        코드 블록_C;
        break;
      :
    default:
        코드 블록_N;
        break;
}
```

switch.php		결과
01	$fruits = "banana";	
02	switch ($fruits) {	
03	case "apple":	
04	echo "사과 선택";	
05	break;	
06	case "banana":	
07	echo "바나나 선택";	
08	break;	바나나 선택
09	case "grape":	
10	echo "포도 선택";	
11	break;	
12	default:	
13	echo "과일 없음";	
14	break;	
15	}	

6. break, continue 문

코드 블록이 반복 수행되는 도중에 중단하거나(break), 건너뛸 수(continue) 있습니다.

6.1 break 문

반복 코드 블록 내에서 break 문을 만나게 되면 코드 블록의 반복 수행이 종료됩니다. do ~ while 문이나 for 문에서의 사용을 참고하기 바랍니다.

6.2 continue 문

continue 문은 특정 조건이 true면 현재 반복 수행을 건너띄고 다음 반복 수행으로 넘어갑니다.

continue.php 코드에서 2번 라인의 $i%2 연산은 $i 변수값이 홀수일 때 1(true)이 되므로 3번 라인의 continue 문이 처리되어 현재의 반복 수행은 건너띄고 다음 반복으로 넘어가게 됩니다. $i 변수값이 짝수일 때 0(false)값이 되므로 continue 문은 처리되지 않고 5번 라인을 수행하게 됩니다.

continue.php		결과
1	for ($i=1; $i<=10; $i++) {	
2	if ($i % 2) { // $i 값이 홀수면 1(true), 짝수면 0(false)	2
3	continue; // 현재의 반복수행 코드블록 건너띔	4
4	}	6
5	echo $i;	8
6	}	10