도전! PHP

01. 다음 코드는 함수에 대한 정의와 호출을 모두 담고 있습니다. 코드를 보고 결과를 유추해 보시오.

```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width,
initial-scale=1.0">
	<title>함수에 대한 정의와 호출</title>
</head>
<body>
	<?php
		$d = addnum(10, 20);

		function addnum($a, $b) {
			$c = $a + $b;
			return $c;
		}

		$e = addnum(100, 200);
		print "₩$d = $d <br> ₩$e = $e";
	?>
</body>
</html>
```

정답

$d = 30
$e = 300

결과 화면

```
$d = 30
$e = 300
```

해설

함수를 사용하기 위해서는 기본적으로 선언을 한 다음 호출해야 합니다. 그렇다면 이 코드는 문제가 있는 것입니다. 즉 선언을 하기 전에 호출을 했기 때문입니다. PHP에서는 이런 경우에도 에러 없이 원하는 결과 값을 구할 수도 있지만 되도록 이와 같은 방법은 사용하지 않는 게 좋습니다. 반드시 함수를 선언한 다음 호출해 사용하는 습관을 들이세요.

02. 두 문자열을 입력받은 다음 문자열을 연결하고 되돌려 보내는 함수를 만들어 보시오.

정답

```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width, initial-scale=1.0">
	<title>함수</title>
</head>
<body>
```

```
        <?php
                function addstring($a, $b) {
                        $c = "$a. $b 님 즐거운 시간 되십시오.";
                        return $c;
                }

                $d = addstring("안녕하세요. ", "홍길동");
                print $d;
        ?>
</body>
</html>
```

결과 화면

안녕하세요. . 홍길동 님 즐거운 시간 되십시오.

해설

결과는 “안녕하세요. 홍길동 님 즐거운 시간 되십시오.”가 출력될 것입니다. 두 문자열을 연결할 때는 PHP에서 닷(.)을 사용하시는 건 아시죠. $a.$b는 두 변수에 저장된 문자열을 연결한다는 뜻입니다. 만약 +를 사용한다면 잘못된 결과가 나오게 됩니다.

03. 값 전달과 참조 전달 이해를 위한 예제입니다. 다음 코드를 보고 차이점을 설명해 보시오.

샘플1

```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width,
initial-scale=1.0">
	<title>함수</title>
</head>
<body>
	<?php
		function getmenu($a) {
			$a = "Coffee";
		}
		$b = "Tea";
		getmenu($b);

		print $b;
	?>
</body>
</html>
```

샘플2

```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width,
initial-scale=1.0">
	<title>함수</title>
</head>
<body>
	<?php
		function getmenu(&$a) {
```

```
                $a = "Coffee";
        }
        $b = "Tea";
        getmenu($b);

        print $b;
    ?>
</body>
</html>
```

정답

샘플 1 결과 값 : Tea
샘플 2 결과 값 : Coffee

결과 화면

Tea

Coffee

해설

샘플 1의 경우 변수에 저장된 내용을 전달했기 때문에 $a 변수에는 $b의 내용 “Tea”만 저장됩니다. 즉 두 변수는 전혀 별개이며 서로 전달한 것입니다.
샘플 2는 $b의 주소를 $a 변수와 동일하도록 주소 연산자 &를 사용했습니다. 두 변수는 주소는 같기 때문에 어느 하나의 변수 값을 변경하더라도 두 변수의 값은 변경되게 됩니다.

04. require와 include의 차이점을 설명하시오.

정답

require와 include는 미리 만들어 놓은 변수나 함수를 파일에 저장해 놓은 다음 필요할 때 불러와 사용할 수 있습니다. 이 두 기능은 동일하지만 에러가 발생했을 때는 require는 프로그램의 실행이 정지되지만 include는 프로그램의 실행이 계속됩니다. 그 이외에는 결과가 같습니다.

05. 함수의 지역 변수와 글로벌 변수를 사용해 만든 코드입니다. 변수에 저장된 값을 설명하시오.

```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width,
initial-scale=1.0">
	<title>함수</title>
</head>
<body>
	<?php
		$x = $y = 10;
		function funcscope() {
			$x = 20;
			global $y;
			static $z = 0;

			$x++; $y++; $z++;
			print "₩$x = $x, ₩$y = $y, ₩$z = $z";
			print "<br>";
			return $z;
		}
		$k = funcscope();
		$k = funcscope();
		print "₩$x = $x, ₩$y = $y, ₩$k = $k";
	?>
</body>
</html>
```

정답

$x = 21, $y = 11, $z = 1
$x = 21, $y = 12, $z = 2
$x = 10, $y = 12, $k = 2

결과 화면

```
$x = 21, $y = 11, $z = 1
$x = 21, $y = 12, $z = 2
$x = 10, $y = 12, $k = 2
```

해설

함수의 밖과 함수 내부의 변수는 이름이 같다 하더라도 저장되는 공간이 다르다는 건 알고 있을 것입니다. global과 static의 이해가 필요합니다. global은 전역 변수를 사용하겠다는 뜻입니다. static은 현 함수의 작업이 끝나더라도 변수의 값을 기억하고 있습니다. 만약 static을 붙이지 않게 되면 함수 작업이 끝나면 변수는 초기화됩니다.