Errata for
PHP and Algorithmic Thinking for the Complete Beginner
Second Edition

Exercise 14.3-5 Finding the Sum of Digits

Write a PHP script that prompts the user to enter a three-digit integer and then calculates the sum of its digits. Solve this exercise without using the integer remainder (%) operator.

14.4 Review Questions: True/False

15. The following statement displays the word "HELLO".

```
echo substr(strtoupper("hello there!"), 0, 5)+;
```

14.5 Review Exercises

3. Write a PHP script that prompts the user to enter his or her name and then creates a secret password consisting of three letters (in lowercase) randomly picked up from his or her name, and a random four-digit number. For example, if the user enters "Vassilis Bouras" a secret password can probably be one of "sar1359" or "vbs7281" or "bor1459". Space characters are not allowed in the secret password.
4. Write a PHP script that prompts the user to enter a three-digit integer and then reverses it. For example, if the user enters the number 375, the number 573 must be displayed. Solve this exercise without using the integer remainder (%) operator.

15.8 How to Negate Boolean Expressions

However, there is a small detail that you should be careful with. If both AND (`&&`) and OR (`||`) operators co-exist in a complex Boolean expression, then the expressions that use the OR (`||`) operators in the negated Boolean expression must be enclosed in parentheses, in order to preserve the initial order of precedence. For example, if the original Boolean expression is

```
$x >= 5 && $x <= 10 || $y == 3
```

24.3 Review Questions: True/False

4. The following PHP script

```
<?php
  $a = 5;
  $total = $total + $a;
  echo $total;
?>
```

satisfies the property of effectiveness.

25.4 Review Questions: True/False

14. The following PHP script

```
<?php
  do {
    echo "Hello";
    $i++;
  } while ($i <= 10);
?>
```

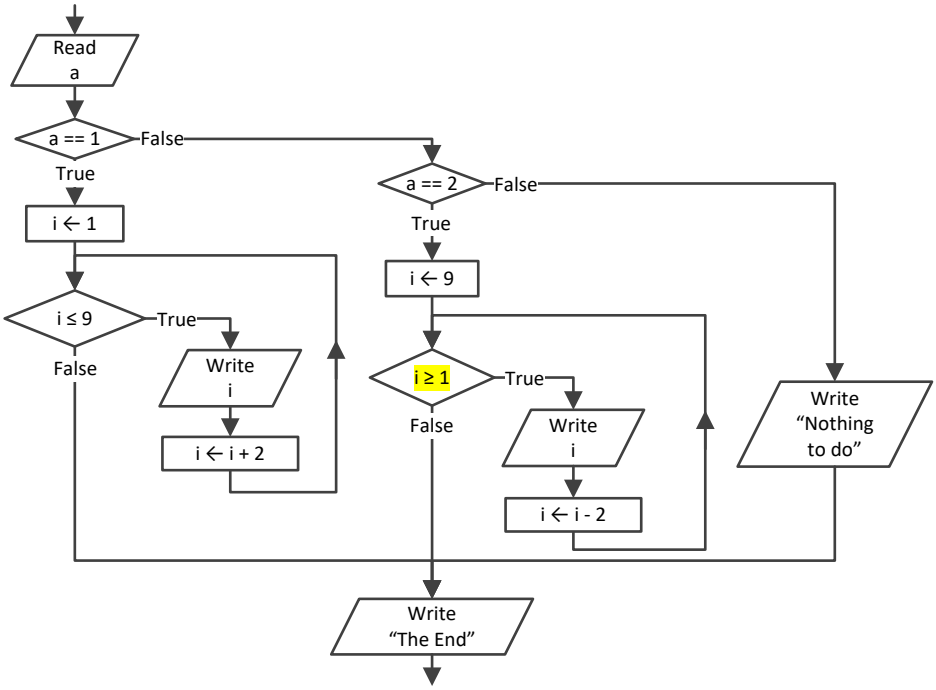
satisfies the property of effectiveness.

28.3 The "Ultimate" Rule

```
$positives_given = 0;            //Initialization of $positives_given
while ($positives_given != 3) {  //This is dependent on $positives_given
```

```
    A statement or block of statements
  if ($x > 0) {
    $positives_given += 1;       //Update/alteration of $positives_given
  }
}
```

Exercise 29.2-4 Designing the Flowchart Fragment



29.4 Review Exercises

8. Design the flowchart that corresponds to the following PHP script.

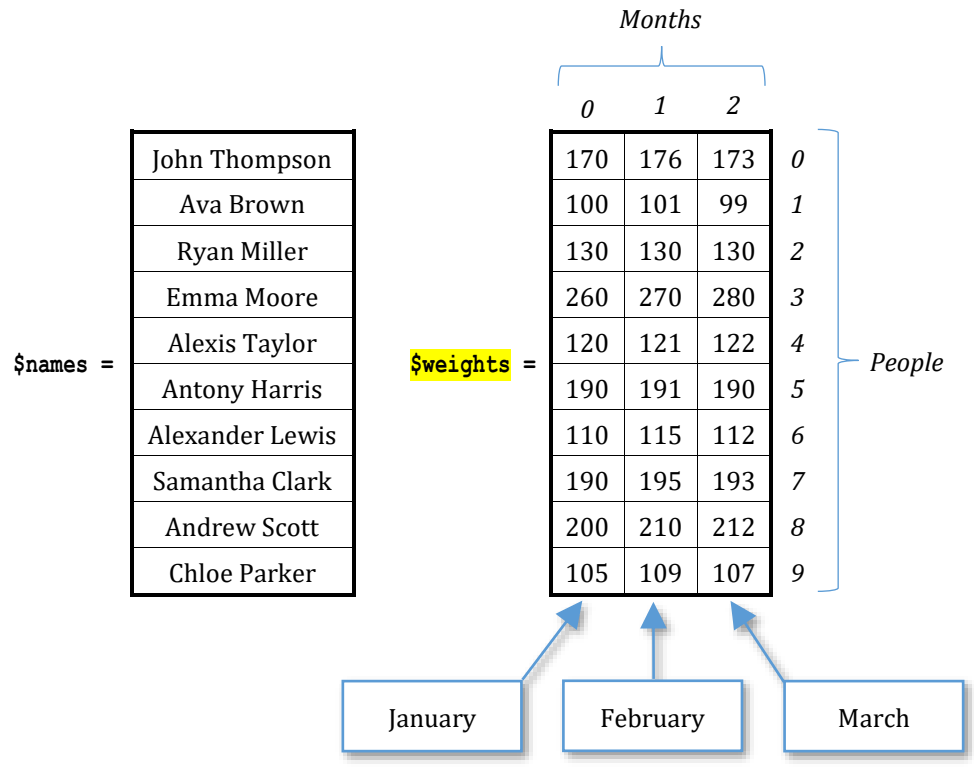
```
<?php
  $a = trim(fgets(STDIN));
  if ($a > 0) {
    $i = 0;
    while ($i <= $a) {
      echo 3 * $i + $i / 2.0;
      $i++;
    }
  }
  else {
    $i = 10;
    do {
      echo 2 * $i - $i / 3.0;
      $i -= 3;
    } while ($i >= $a);
  }
?>
```

Exercise 31.2-3 Designing Arrays

Design the necessary arrays to hold the names of ten people as well as the average weight (in pounds) of each person for January, February, and March. Then add some typical values to the arrays.

Solution

In this exercise, you need a one-dimensional array for names, and a two-dimensional array for people's weights.



31.7 How to Add User-Entered Values to a One-Dimensional Array

There is nothing new here. Instead of reading a value from the keyboard and assigning that value to a variable, you can directly assign that value to a specific array element. The next PHP script prompts the user to enter the names of four people, and assigns them to the elements at index positions 0, 1, 2, and 3, of the array `$names`.

31.13 Review Questions: True/False

35. If array `$b` contains 30 elements (arithmetic values), the following code fragment doubles the values of all of its elements.

```
for ($i = 29; $i >= 0; $i--) {
  $b[$i] = $b[$i] * 2;
}
```

31.14 Review Questions: Multiple Choice

10. If array `$b` contains 30 elements (arithmetic values), the following code fragment

```
for ($i = 29; $i >= 1; $i--) {
  $b[$i] = $b[$i] * 2;
}
```

a. doubles the values of some of its elements.
b. doubles the values of all of its elements.
c. none of the above

Exercice 33.4-1 Finding the Average Value of Two Grades

file_33_4_1

```
<?php
  define("STUDENTS", 20);

  $names = [];
  $grades_lesson1 = [];
  $grades_lesson2 = [];

  for ($i = 0; $i <= STUDENTS - 1; $i++) {
    echo "Enter student name No",($i + 1), ": ";
    $names[$i] = trim(fgets(STDIN));

    echo "Enter grade for lesson 1: ";
```

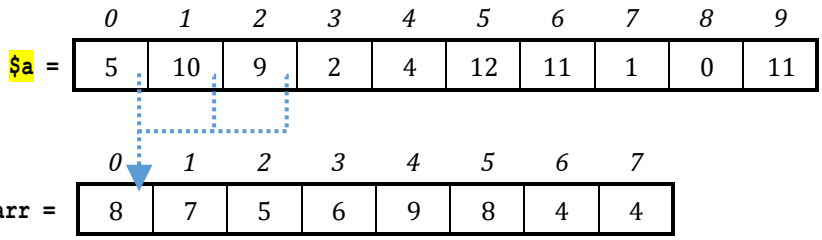
```
    $grades_lesson1[$i] = trim(fgets(STDIN));

    echo "Enter grade for lesson 2: ";
    $grades_lesson2[$i] = trim(fgets(STDIN));
  }

  //Calculate the average grade for each student
  //and display the names of those who are greater than 89
  for ($i = 0; $i <= STUDENTS - 1; $i++) {
    $total = $grades_lesson1[$i] + $grades_lesson2[$i];
    $average = $total / 2.0;
    if ($average > 89) {
      echo $names[$i], "\n";
    }
  }
?>
```

Exercise 34.1-1 Creating an Array that Contains the Average Values of its Neighboring Elements

Write a PHP script that lets the user enter 100 positive numerical values into an array. Then, the script must create a new array of 98 elements. This new array must contain, in each position the average value of the three elements that exist in the current and the next two positions of the given array.

Solution

Let's try to understand this exercise through an example using 10 elements.



Array `$new_arr` is the new array that is created. In array `$new_arr`, the element at position 0 is the average value of the elements in the current and the next two positions of array `$a`; that is, (5 + 10 + 9) / 3 = 8. The element at position 1 is the average value of the elements in the current and the next two positions of array `$a`; that is, (10 + 9 + 2) / 3 = 7, and so on.

Exercise 34.1-5 Creating Two Arrays – Separating Positive from Negative Values

✎ *Note that the arrays* `$pos` *and* `$neg` *contain a total number of* `$pos_index` *and* `$neg_index` *elements respectively. This is why the two last loop control structures iterate until variable* `$i` *reaches values* `$pos_index - 1` *and* `$neg_index - 1`*, respectively, and not until* `ELEMENTS - 1`*, as you may mistakenly expect.* ~~*Obviously the sum of* `$pos_index + $neg_index` *equals to* `ELEMENTS`.~~

Exercise 34.3-5 Finding the Minimum Value of a Two-Dimensional Array

This regards the e-book version only

✎ *In this exercise you can* ***not*** *do the following because if you do, and variable* `$j` *starts from 1, the whole column with index 0 won't be checked!*

```
//Find minimum
$minimum = $t[0][0];
for ($i = 0; $i <= CITIES - 1; $i++) {
  for ($j = 1; $j <= DAYS - 1; $j++) {
    if ($t[$i][$j] < $minimum) {
      $minimum = $t[$i][$j];
    }
  }
}
```

Exercise 34.4-7 The Five Best Scorers

Now, in order to sort all rows, you need to nest this code fragment in a for-loop that iterates for all of them, as shown next.

```
for ($i = 0; $i <= TEAMS - 1; $i++) {
  $swaps = false;
  for ($m = 1; $m <= PLAYERS - 1; $m++) {
    for ($n = PLAYERS - 1; $n >= $m; $n--) {
      if ($g[$i][$n] < $g[$i][$n - 1]) {
        $temp = $g[$i][$n];
        $g[$i][$n] = $g[$i][$n - 1];
        $g[$i][$n - 1] = $temp;

        $temp = $p[$i][$n];
        $p[$i][$n] = $p[$i][$n - 1];
        $p[$i][$n - 1] = $temp;
      }
    }
    if (!$swaps) break;
  }
}
```

Exercise 34.4-9 Sorting One-Dimensional Arrays While Preserving the Relationship with a Second Array

Write a PHP script that prompts the user to enter the total number of kWh consumed each month for a period of one year. It then displays each number of KWh consumed (in descending order) along with the name of the corresponding month. Use the selection sort algorithm.

36.2 How to Make a Call to a Method

Now, suppose that you want to calculate a result using the following expression

$$y = x^3 + \frac{1}{x}$$

Exercise 38.2-3 Progressive Rates and Electricity Consumption

The LAV Electricity Company charges subscribers for their electricity consumption according to the following table (monthly rates for domestic accounts).

Kilowatt-hours (kWh)	USD per kWh
kWh ≤ 400	$0.08
401 ≤ kWh ≤ 1500	$0.22
1501 ≤ kWh ≤ 2000	$0.35
2001 ≤ kWh	$0.50