Internet Fundamentals & Introduction to Web Technologies

Course: IT (044615)

Lecture: 6

Introduction to PHP

Dr. Ramez Hajislam

Origin and Uses of PHP

- Developed by Rasmus Lerdorf in 1994
- PHP is a server-side scripting language, embedded in XHTML pages
- PHP has good support for form processing
- PHP can interface with a wide variety of databases
- Open Source:
  - http:/www.php.net

Overview of PHP

- When a PHP document is requested of a server, the server will send the document first to a PHP processor
- The result of the processing is the response to the request
- Two modes of operation
  - Copy mode in which plain HTML is copied to the output
  - Interpret mode in which PHP code is interpreted and the output from that code sent to output
  - The client never sees PHP code, only the output produced by the code

Overview of PHP

- PHP has typical scripting language characteristics
  - Dynamic typing, untyped variables
  - Associative arrays
  - Pattern matching
  - Extensive libraries:
    - http://www.php.net

General Syntactic Characteristics

- PHP code is contained between the tags <?php and ?>
- Code can be included with the PHP include

```
Include(“table2.inc”);
```

- When a file is included, the PHP interpreter reverts to copy mode
  - Thus, code in an include file must be in

    <?php and ?> tags
- All variable names in PHP begin with $ and continue as usual for variables
- Variable names are case sensitive
- *However* keywords and function names are *not* case sensitive

The reserved words of PHP

and	else	global	require	virtual
break	elseif	if	return	xor
case	extends	include	static	while
class	false	list	switch	
continue	for	new	this	
default	foreach	not	true	
do	function	or	var	

PHP Syntax

- One line comments can begin with # or // and continue to the end of the line
- Multi-line comments can begin with /* and end with */
- PHP statements are terminated with semicolons
- Curly braces are used to create compound statements
- Variables cannot be defined in a compound statement unless it is the body of a function

Primitives, Operations, Expressions

- Four scalar types:
  - **boolean, integer, double, string**
- Two compound types:
  - **array**, *object*
- Two special types:
  - *resource* and **NULL**

Variables

- Variables are not declared except in order to specify scope or lifetime
- A variable that has not been assigned a value is *unbound* and has the value NULL
  - NULL is coerced to 0 if a number is needed, to the empty string if a string is needed
  - Both of these coercions count as boolean FALSE
- IsSet, Unset functions:
  - IsSet($fruit); return boolean value
  - unset($fruit) Unassign the variable

Integer Type

- PHP distinguishes between integer and floating point numeric types
- Integer is equivalent to long in C, that is, usually 32 bits

Double Type

- Literal double type numeric values include a period and/or the exponent sign: either e or E
- Double type values are stored internally as double precision floating point values

String Type

- Characters in PHP are one byte.
- String literals are enclosed in single or double quotes
  - Double quoted strings have escape sequences interpreted and variables interpolated
  - Single quoted strings have neither escape sequence interpretation nor variable interpolation
  - A literal $ sign in a double quoted string must be escaped with a backslash, \
- Double-quoted strings can cover multiple lines, the included end of line characters are part of the string value

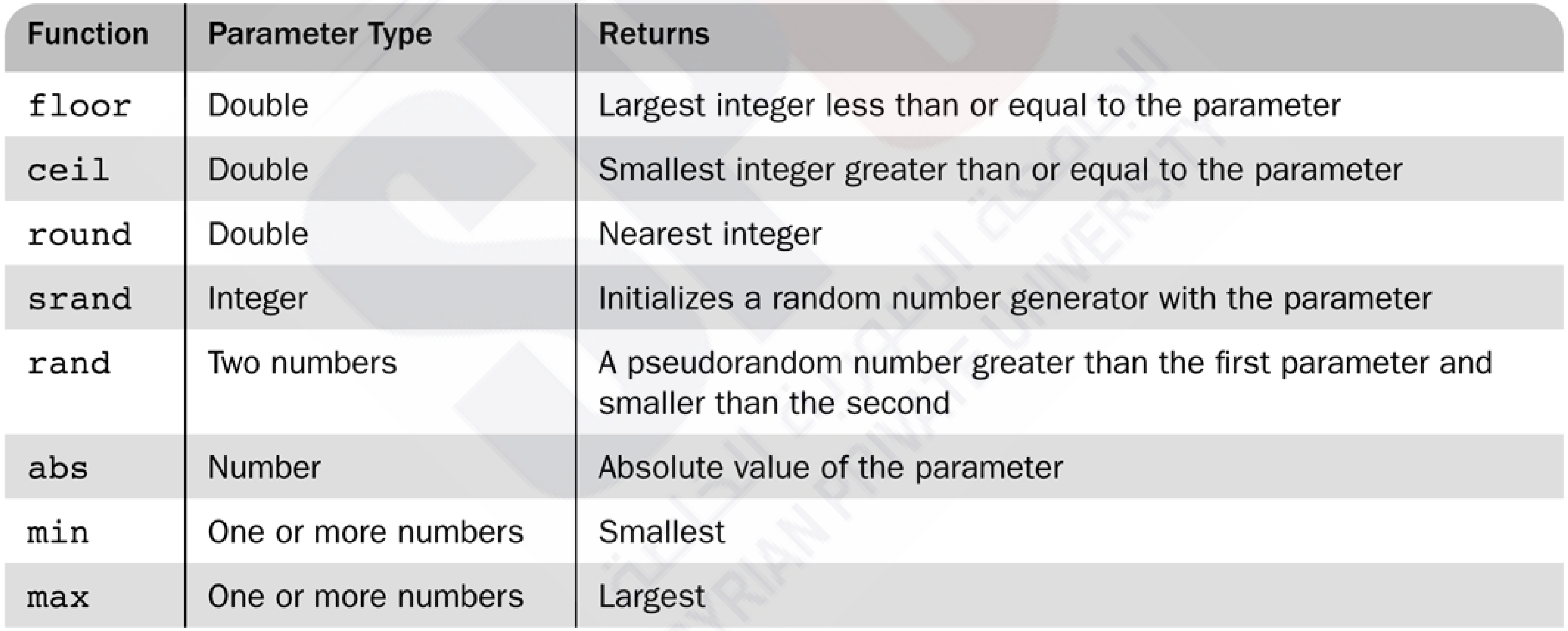
Boolean Type

- The boolean type has two values :TRUE and FALSE
- Other type values are coerced as needed by context, for example, in control expressions
  - The integer value 0, the empty string and the literal string “0” all count as false
  - NULL counts as false
  - The double value 0.0 counts as false. Beware, however, that double calculations rarely result in the exact value 0.0

Arithmetic Operators and Expressions

- PHP supports the usual operators supported by the C/C++/Java family: +, -, *, /, %, ++, --
- Integer divided by integer results in integer if there is no remainder but results in double if there is a remainder
  - 12/6 is 2
  - 12/5 is 2.4
- A variety of numeric functions is available:

Some useful predefined functions



Function	Parameter Type	Returns
`floor`	Double	Largest integer less than or equal to the parameter
`ceil`	Double	Smallest integer greater than or equal to the parameter
`round`	Double	Nearest integer
`srand`	Integer	Initializes a random number generator with the parameter
`rand`	Two numbers	A pseudorandom number greater than the first parameter and smaller than the second
`abs`	Number	Absolute value of the parameter
`min`	One or more numbers	Smallest
`max`	One or more numbers	Largest

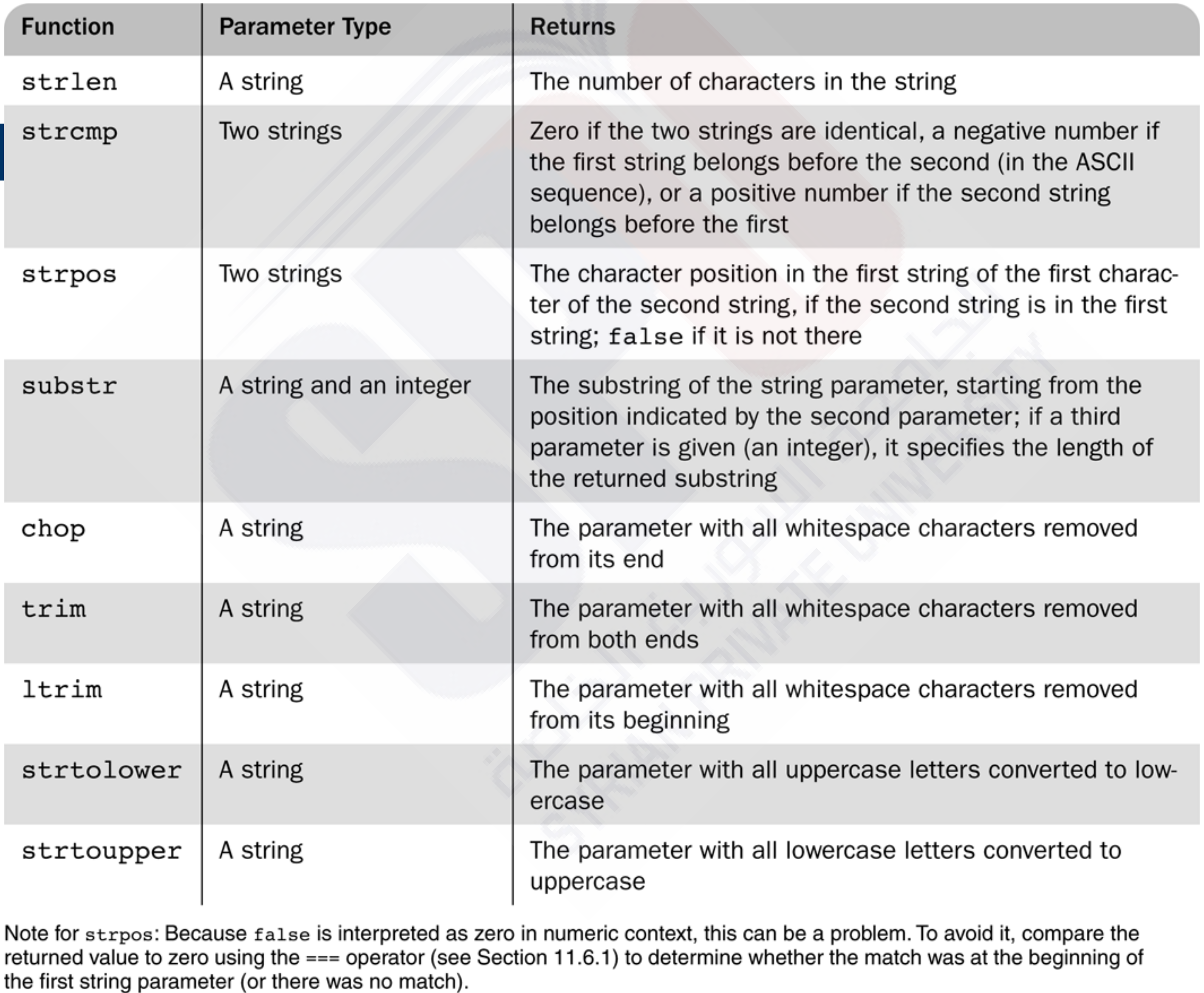
11.4 String Operations

- String catenation is indicated with a period (.)
- Characters are accessed in a string with a subscript enclosed in curly braces
- $str = “apple”; (then $str{4}= “e”)
- Many useful string functions are provided
  - `strlen` gives the length of a string
  - `strcmp` compares two strings as strings
  - Chop removes whitespace from the end of a string

Scalar Type Conversions

- Implicit type conversions as demanded by the context in which an expression appears
  - A string is converted to an integer if a numeric value is required and the string has only a sign followed by digits
  - A string is converted to a double if a numeric value is required and the string is a valid double literal (including either a period or e or E)
- Type conversions can be forced in three ways
  - `(int)$sum;`
  - `intval($sum;` We have also: `doubleval, strval`
  - `settype($x, “integer”)`
- Type can be determined with the `gettype` function and with the `is_int` function and similar functions for other types

Some commonly used string functions



Function	Parameter Type	Returns
`strlen`	A string	The number of characters in the string
`strcmp`	Two strings	Zero if the two strings are identical, a negative number if the first string belongs before the second (in the ASCII sequence), or a positive number if the second string belongs before the first
`strpos`	Two strings	The character position in the first string of the first character of the second string, if the second string is in the first string; `false` if it is not there
`substr`	A string and an integer	The substring of the string parameter, starting from the position indicated by the second parameter; if a third parameter is given (an integer), it specifies the length of the returned substring
`chop`	A string	The parameter with all whitespace characters removed from its end
`trim`	A string	The parameter with all whitespace characters removed from both ends
`ltrim`	A string	The parameter with all whitespace characters removed from its beginning
`strtolower`	A string	The parameter with all uppercase letters converted to lowercase
`strtoupper`	A string	The parameter with all lowercase letters converted to uppercase

Note for `strpos`: Because `false` is interpreted as zero in numeric context, this can be a problem. To avoid it, compare the returned value to zero using the === operator (see Section 11.6.1) to determine whether the match was at the beginning of the first string parameter (or there was no match).

Assignment Operators

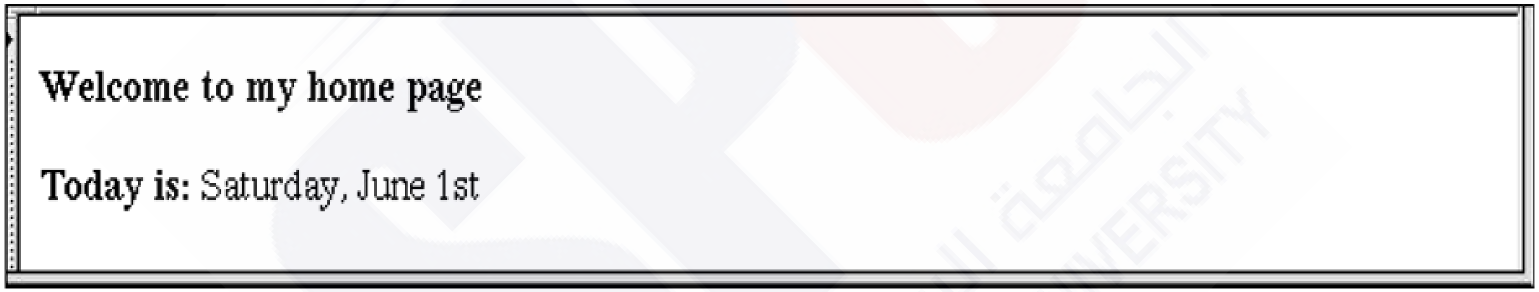
- The assignment operators used in C/C++/Java are supported in PHP: =, +=, -=

Output

- The `print` function is used to send data to output
  - `print` takes string parameters, PHP coerces as necessary
- The C printf function is also available
  - `printf(“x = %5d is %s\n”, $x, $size);`

    Displays `$x` as an integer and `$size` as a string

Display of the output of `today.php`



```
Welcome to my home page

Today is: Saturday, June 1st
```

Relational Operators

- PHP has the usual comparison operators: >, < <=, >=, == and !=
- PHP also has the identity operator ===
  - This operator does not force coercion (same types and values).
- The regular comparisons will force conversion of values as needed
  - Comparing a string with a number (other than with ===) will result in the string converting to a number if it can be. Otherwise the number is converted to a string
  - If two strings are compared (other than with ===) and the strings can both be converted to numeric values, the conversion will be done and the converted values compared
  - Use `strcmp` on the strings if the latter feature is a problem

Boolean Operators

- PHP supports &&, || and ! as in C/C++/Java
- The lower precedence version `and` and `or` are provided
- The `xor` operator is also provided

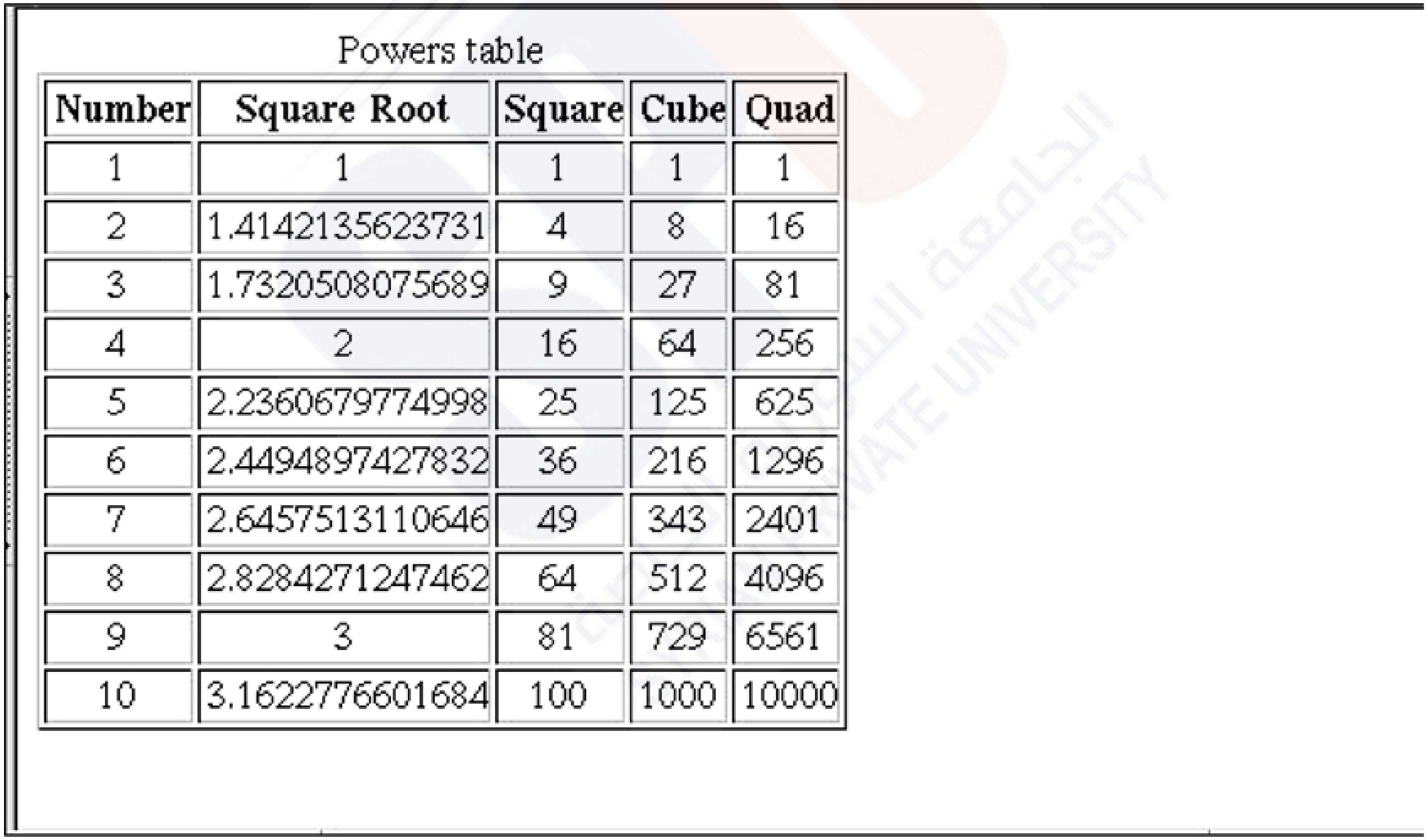
Selection Statements

- PHP provides an I with almost the same syntax as C/C++/Java
  - The only difference is the elseif (note, not elsif as in Perl)
- The `switch` statement is provided with syntax and semantics similar to C/C++/Java
  - The case expressions are coerced before comparing with the control expression
  - `break` is necessary to prevent execution from flowing from one case to the next

Loop Statements

- PHP provides the while and for and do-while as in JavaScript
- The for loop is illustrated in the example `powers.php`
- This example also illustrates a number of mathematical functions available in PHP

The output of `powers.php`



Powers table

Number	Square Root	Square	Cube	Quad
1	1	1	1	1
2	1.4142135623731	4	8	16
3	1.7320508075689	9	27	81
4	2	16	64	256
5	2.2360679774998	25	125	625
6	2.4494897427832	36	216	1296
7	2.6457513110646	49	343	2401
8	2.8284271247462	64	512	4096
9	3	81	729	6561
10	3.1622776601684	100	1000	10000

Arrays

- Arrays in PHP combine the characteristics of regular arrays and hashes
  - An array can have elements indexed numerically. These are maintained in order
  - An array, even the same array, can have elements indexed by string. These are not maintained in any particular order
- The elements of an array are, conceptually, key/value pairs

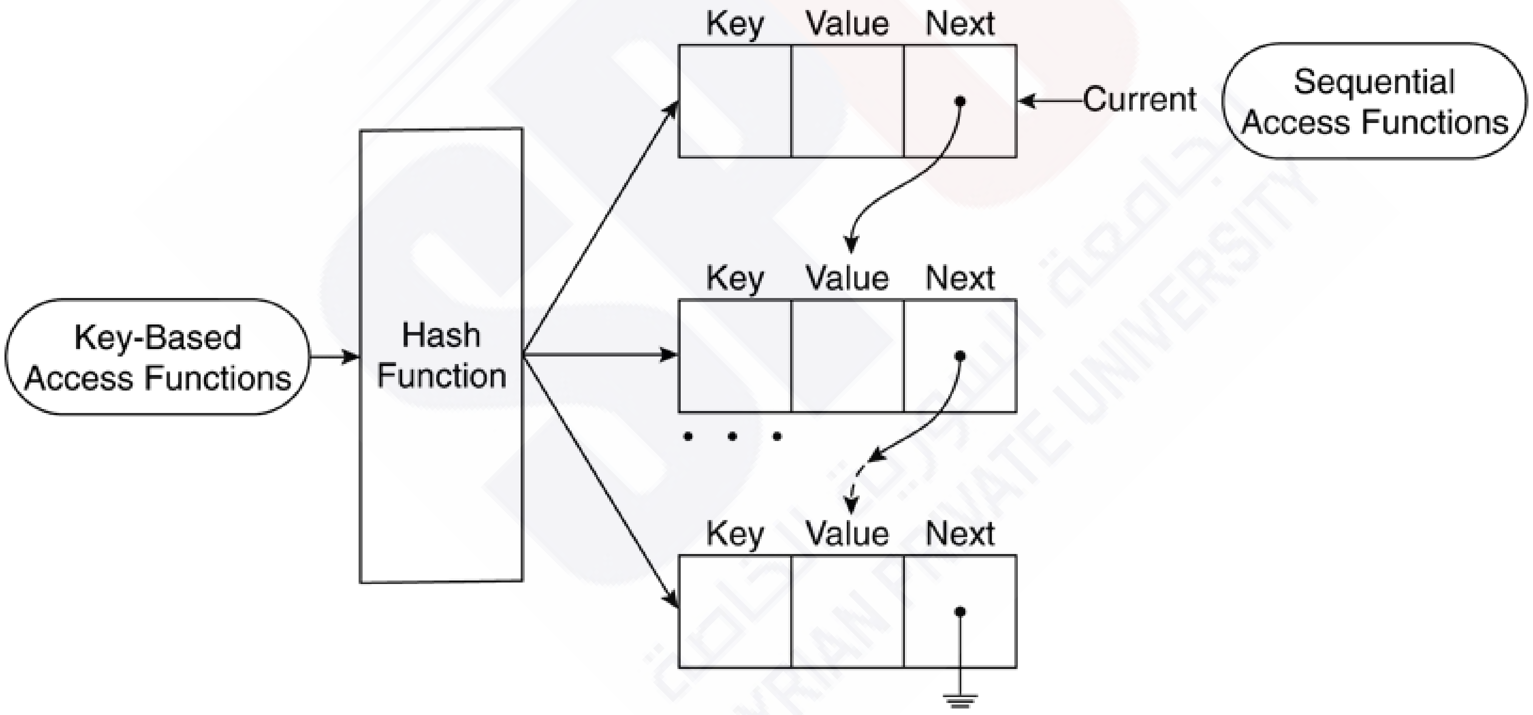
Array Creation

- Two ways of creating an array
  - Assigning a value to an element of an array
  - Using the array function
- Create a numerically indexed array
  - `$A = array(23, ‘xiv’, “bob”, 777);`
- Create an array with string indexes
  - `$B = array(“x” => “xerxes”, “y” => “ytrbium”);`

Functions for Dealing with Arrays

- The unset function can be used to remove an array or an element of an array: `unset($list[2])`
- The `array_keys` function returns a list of the keys of an array
- The `array_values` returns a list of values in an array
- The `array_key_exists` function returns true if a given key is actually present in a given array
- `is_array` determines if its argument is an array
- `implode` converts an array of strings to a single string, separating the parts with a specified string
- `explode` converts a string into a list of strings by separating the string at specified characters

Logical internal structure of arrays



Accessing Array Elements

- Array elements are accessed by using a subscript in square brackets
- An array can be assigned to a list of variables
  - `list($x, $y, $z) = array(“xx”, “yy”, “zz”);`

Sequential Access to Array Elements

- PHP maintains a marker in each array, called the current pointer
  - Several functions in PHP manipulate the current pointer
  - The pointer starts at the first element when the array is created
- The next function moves the pointer to the next element and returns the value there
- The each function move the pointer to the next element and returns the key/value pair at the previous position
  - The key and value can be accessed using the keys “key” and “value” on the key/value pair
- Both functions return false if no more elements are available
- `prev` moves the pointer back towards the beginning of the array
- `reset` moves the pointer to the beginning of the array

Arrays as Stacks

- PHP provides the `array_push` function that appends its arguments to a given array
- The function `array_pop` removes the last element of a given array and returns it

11.7 Iterating Through an Array

- The foreach statement has two forms for iterating through an array

  ***foreach** (array **as** scalar_variable) loop body*

  ***foreach** (array **as** key => value) loop body*

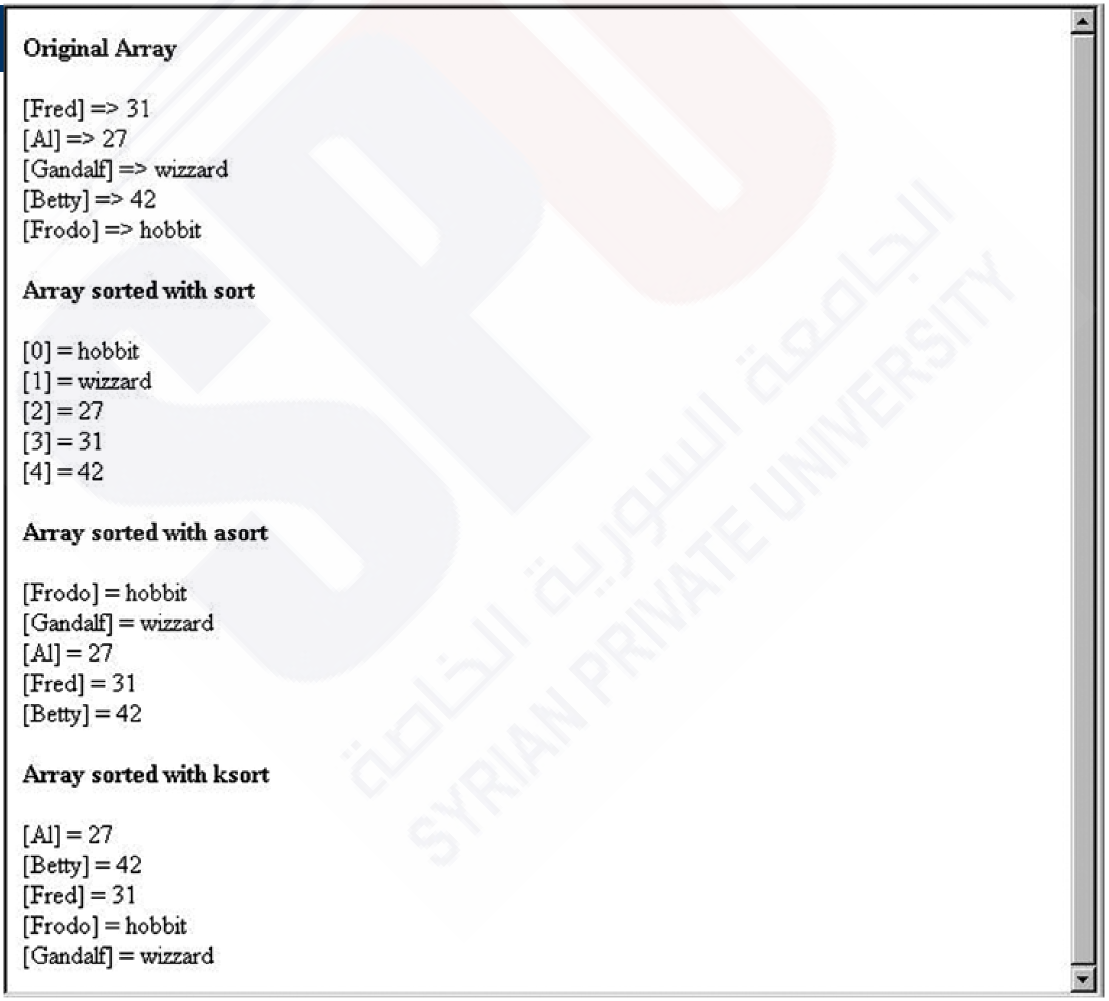
- The first version assigns each value in the array to the scalar_variable in turn
- The second version assigns each key to key and the associated value to value in turn
- In this example, each day and temperature is printed

```
$lows=array("Mon"=>23, "Tue" => 18, "Wed" => 27);
foreach ($lows as $day => $temp)
print("The low temperature on $day was $temp <br
   />");
```

Sorting Arrays

- The `sort` function sorts the values in an array and makes a numerically subscripted array from the sorted list
- The function `asort` sorts the values in an array but keeps the original key/value association
- The function `ksort` is similar to asort but sorts by keys
- The functions `rsort`, `arsort` and `krsort` are similar but sort in reverse order
- The example `sorting.php` illustrates the various sort functions

The output of `sorting.php`



```
Original Array

[Fred] => 31
[Al] => 27
[Gandalf] => wizzard
[Betty] => 42
[Frodo] => hobbit

Array sorted with sort

[0] = hobbit
[1] = wizzard
[2] = 27
[3] = 31
[4] = 42

Array sorted with asort

[Frodo] = hobbit
[Gandalf] = wizzard
[Al] = 27
[Fred] = 31
[Betty] = 42

Array sorted with ksort

[Al] = 27
[Betty] = 42
[Fred] = 31
[Frodo] = hobbit
[Gandalf] = wizzard
```

General Characteristics of Functions

- Function syntax

```
function name([parameters]) {
...
}
```

- The parameters are optional, but not the parentheses
- Function names are not case sensitive
- A return statement causes the function to immediately terminate and return a value, if any, provided in the return
- A function that reaches the end of the body without executing a return, returns no value

Parameters

- A formal parameter, specified in a function declaration, is simply a variable name
- If more actual parameters are supplied in a call than there are formal parameters, the extra values are ignored
- If more formal parameters are specified than there are actual parameters in a call then the extra formal parameters receive no value
- PHP defaults to pass by value
  - Putting an ampersand in front of a formal parameter specifies that pass-by-reference
  - An ampersand can also be appended to the actual parameter (which must be a variable name)

The Scope of Variables

- A variable defined in a function is, by default, local to the function
- A global variable of the same name is not visible in the function
- Declaring a variable in a function with the global declaration means that the functions uses the global variable of that name

Lifetime of Variables

- The usual lifetime of a local variable is from the time the function begins to execute to the time the function returns
- Declaring a variable with the static keyword means that the lifetime is from the first use of the variable to the end of the execution of the entire script
- In this way a function can retain some ‘history’

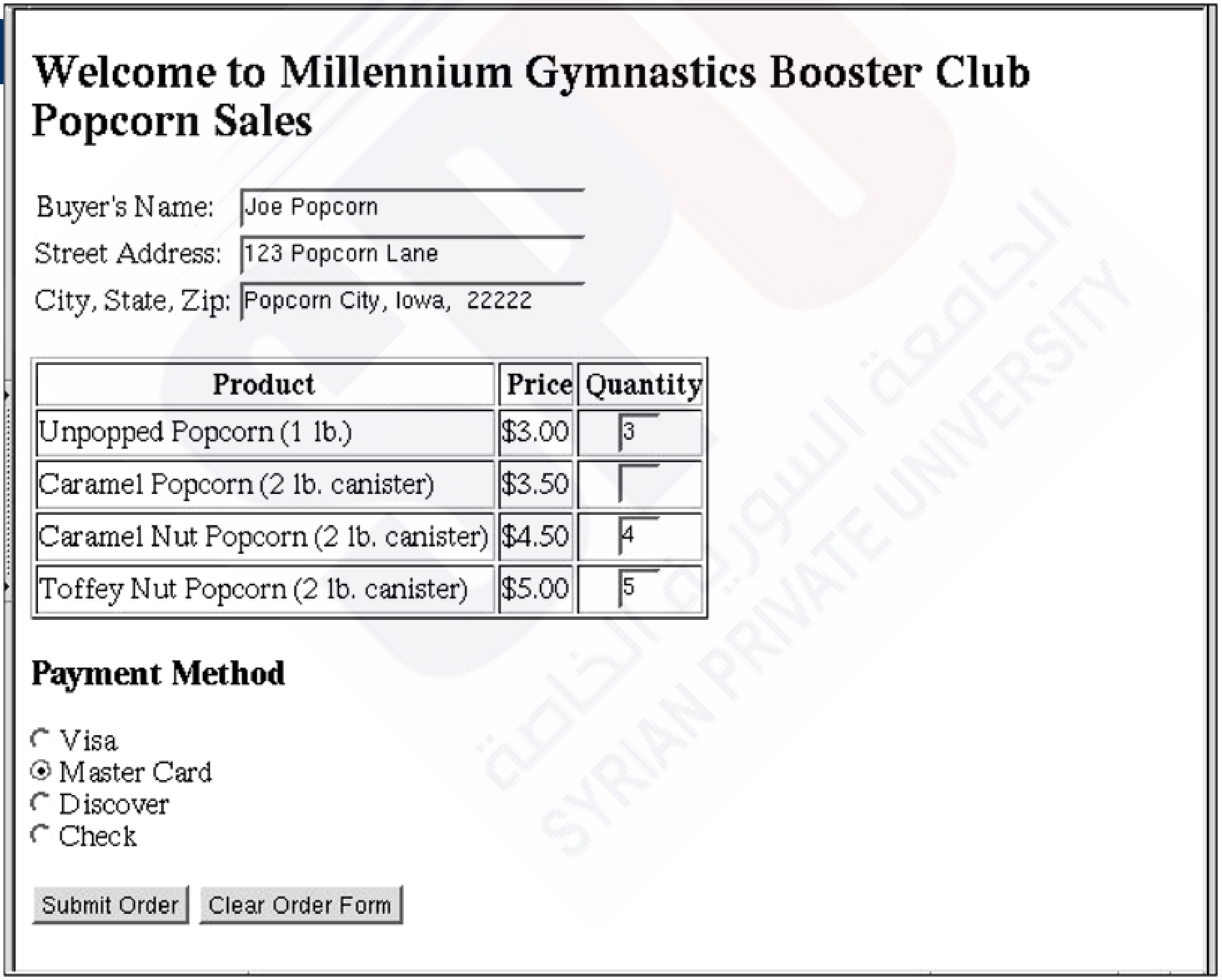
Pattern Matching

- PHP provides both POSIX regular expressions and Perl regular expressions
  - These are generally the same but differ in certain details
- The `preg_match` function matches a pattern, given as a string, with a string
- The `preg_split` function splits a string into an array of strings based on a pattern describing the separators
- The `word_table.php` example illustrates pattern matching in PHP

Form Handling

- The values from forms can be accessed in PHP using the $_POST and $_GET arrays
  - Some web servers allow more direct access, though this has security implications
- The files `popcorn3.html` and `popcorn3.php` implement the popcorn order form using PHP
  - The printf function is used to get two decimal places printed for currency values

The display of `popcorn3.html`



Welcome to Millennium Gymnastics Booster Club Popcorn Sales

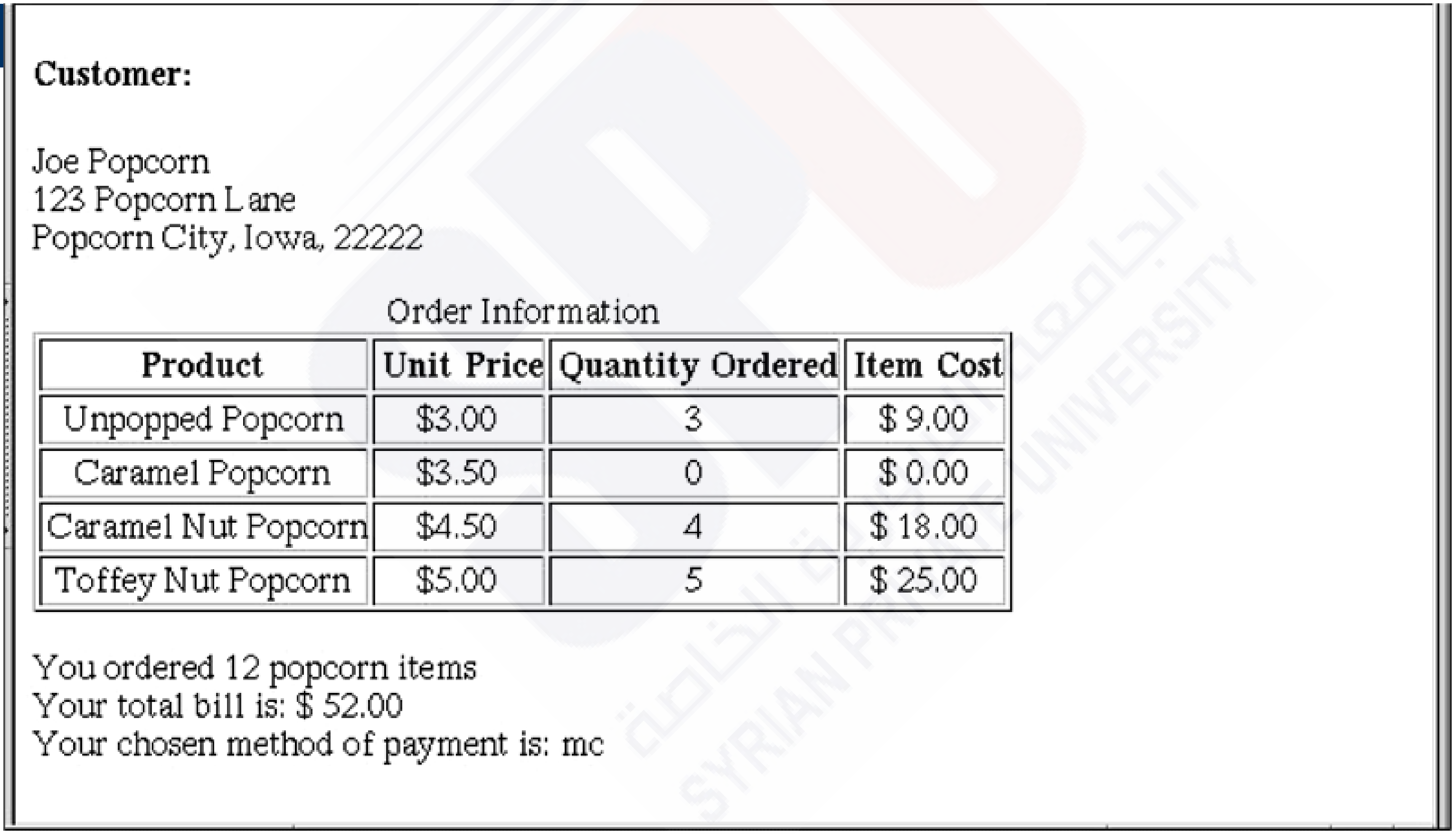
Buyer's Name:	Joe Popcorn
Street Address:	123 Popcorn Lane
City, State, Zip:	Popcorn City, Iowa, 22222

Product	Price	Quantity
Unpopped Popcorn (1 lb.)	$3.00	3
Caramel Popcorn (2 lb. canister)	$3.50	
Caramel Nut Popcorn (2 lb. canister)	$4.50	4
Toffey Nut Popcorn (2 lb. canister)	$5.00	5

Payment Method

Visa [✓ Master Card] Discover Check

The output of `popcorn3.php`



Customer:

Joe Popcorn
123 Popcorn Lane
Popcorn City, Iowa, 22222

Order Information

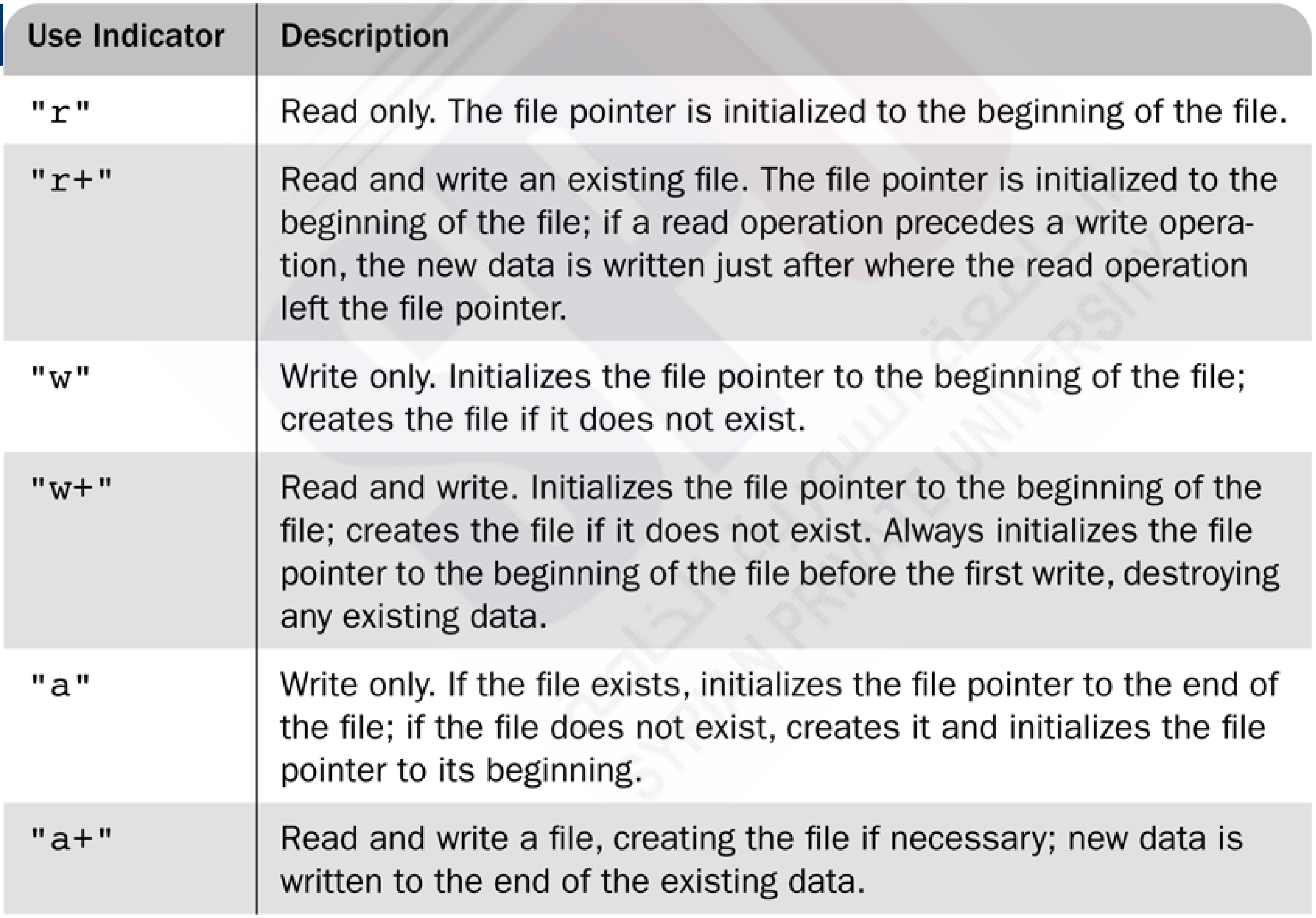
Product	Unit Price	Quantity Ordered	Item Cost
Unpopped Popcorn	$3.00	3	$ 9.00
Caramel Popcorn	$3.50	0	$ 0.00
Caramel Nut Popcorn	$4.50	4	$ 18.00
Toffey Nut Popcorn	$5.00	5	$ 25.00

You ordered 12 popcorn items
Your total bill is: $ 52.00
Your chosen method of payment is: mc

Opening and Closing Files

- The PHP function `fopen` is used to create a file handle for accessing a file given by name
- A second argument to fopen gives the mode of access
- The `fopen` function returns a file handle
- Every open file has a current pointer indicating a point in the file
- Normally input and output operations occur at the current pointer position
- The `file_exists` function tests if a file, given by name, exists
- The function `fclose` closes a file handle

File use indicators



Use Indicator	Description
`"r"`	Read only. The file pointer is initialized to the beginning of the file.
`"r+"`	Read and write an existing file. The file pointer is initialized to the beginning of the file; if a read operation precedes a write operation, the new data is written just after where the read operation left the file pointer.
`"w"`	Write only. Initializes the file pointer to the beginning of the file; creates the file if it does not exist.
`"w+"`	Read and write. Initializes the file pointer to the beginning of the file; creates the file if it does not exist. Always initializes the file pointer to the beginning of the file before the first write, destroying any existing data.
`"a"`	Write only. If the file exists, initializes the file pointer to the end of the file; if the file does not exist, creates it and initializes the file pointer to its beginning.
`"a+"`	Read and write a file, creating the file if necessary; new data is written to the end of the existing data.

Reading from a File

- The `fread` function reads a given number of bytes from a file given by a file handle
  - The entire file can be read by using the filesize function to determine the number of bytes in the file
- The `file` function returns an array of lines from a file named as a parameter
  - No explicit open and close are required for using this function, it does not use a file handle parameter
- The `file_get_contents` method returns the content of a named file as a single string
- The `fgetc` function returns a single character
- The `feof` function returns TRUE if the last character read was the end of file marker, that is, the read was past the end of the file

Writing to a File

- If a file handle is open to for writing or appending, then the `fwrite` function can be used to write bytes to the file
- The `file_put_contents` function writes a given string parameter to a named file, not a file handle

Locking Files

- The flock function will locka a named file
- The function takes a second parameter giving the mode of the lock
  - 1 specifies others can read
  - 2 specifies no other access is allowed
  - 3 removes the lock

Cookies

- HTTP is a *stateless* protocol, that is, the server treats each request as completely separate from any other
- This, however, makes some applications difficult
  - A shopping cart is an object that must be maintained across numerous requests and responses
- The mechanism of cookies can be used to help maintain state by storing some information on the browser system
- A cookie is a key/value pair that is keyed to the domain of the server
  - This key/value pair is sent along with any request made by the browser of the same server
- A cookie has a lifetime which specifies a time at which the cookie is deleted from the browser

Cookies and Security

- Cookies are only returned to the server that created them
- Cookies can be used to determine usage patterns that might not otherwise be ascertained by a server
- Browsers generally allow users to limit how cookies are used
  - Browsers usually allow users to remove all cookies currently stored by the browser
- Systems that depend on cookies will fail if the browser refuses to store them

11.12 PHP Support for Cookies

- PHP provides the `setcookie` function to set a cookie in a response
  - The first parameter is the cookie's name
  - The second, optional, parameter gives the cookie's value
  - The third, optional, parameter gives the expiration
- The cookie must be set before setting content type and before providing any other output
- The `$_COOKIES` array provides access to cookies in the HTTP request

Session Tracking

- Some applications need to keep track of a session
- Sessions are represented internally in PHP with a session id
  - A session consists of key/value pairs
- A session can be initialized or retrieved by using the `session_start` function
  - This function retrieves `$_SESSION`, an array containing the key/value pairs for each cookie in the current request