

WEBD 236

Web Information Systems Programming

Week 4

Copyright © 2012 Todd Whittaker
(todd.whittaker@franklin.edu)



Agenda

- This week's expected outcomes
- This week's topics
- This week's homework
- Upcoming deadlines
- Solutions to homework 2
- Questions and answers



Week 4 Outcomes

- Create HTML forms
- Access form data from a script
- Construct algorithms using selection and repetition structures.

Web Forms

- Basic HTML-based web-form elements
 - Fieldsets
 - Labels
 - Text boxes
 - Text areas
 - Buttons
 - Radio buttons
 - Check boxes
 - Combo boxes

Web

- Basic HTML-based web forms
 - Fieldsets
 - Labels
 - Text boxes
 - Text areas
 - Buttons
 - Radio buttons
 - Check boxes
 - Combo boxes

The screenshot shows a web form with two fieldsets. The first fieldset, titled 'Personal information', contains three input elements: a text box for 'First Name' with the value 'Bill', a text box for 'Last Name' with the value 'Jones', and a text area for 'Comments' containing the text 'This is my comment that <i>stretches</i> over two lines.'. Below these is a dropdown menu for 'Inquiry Type' with 'Customer service' selected. The second fieldset, titled 'Demographic information', contains two groups of radio and check boxes. The 'Gender' group has 'Male' selected. The 'Interests' group has 'Reading' and 'Sports' selected. At the bottom of the form are 'Submit' and 'Clear' buttons.

UNIVERSITY 

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments' name='comments' rows='5'
      cols='40'></textarea>
  </fieldset>
  <input type='submit' value='Submit' />
  <input type='button' value='Clear' />
</form>
```

FRANKLIN UNIVERSITY 

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments'
      cols='40'></textarea>
```

Any input, textarea, select, etc., within the form will be submitted

- action: script to receive data
- method: get or post

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments'
      rows='5'></textarea>
```

fieldset groups related fields graphically (no effect on what the server receives). legend describes the fieldset.

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments' name='comments' rows='5'>
```

Labels are click targets. Clicking a label activates the input control (associated by `for='someid'` in the label and `id='someid'` in the input control).

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments' name='comments' rows='5'
      cols='40'></textarea>
```

PHP receives these data into the `$_GET`, `$_POST`, or `$_REQUEST` superglobals according to the name attribute.

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments' name='comments' rows='5'>
```

`type='text'`
produces a single-line
text box for input.

Web Forms

```
<form action='formelements.php' method='post'>
  <fieldset>
    <legend>Personal information</legend>
    <label for='firstName'>First Name</label>
    <input type='text' id='firstName' name='firstName' />
    <label for='lastName'>Last Name</label>
    <input type='text' id='lastName' name='lastName' />
    <label for='comments'>Comments</label>
    <textarea id='comments' name='comments' rows='5'
      cols='40'></textarea>
```

For large, multi-line
input, use a
textarea. Any text
between the
begin/end tags is put
inside the text area.

Web Forms

```
<label for='inquiryType'>Inquiry Type</label>  
<select id='inquiryType' name='inquiryType'>  
  <option value='error'>Choose an option</option>  
  <option value='prodInfo'>Product information</option>  
  <option value='custSvc'>Customer service</option>  
  <option value='returns'>Returns</option>  
  <option value='other'>Other</option>  
</select>  
</fieldset>
```

Each option appears on its own line, value is what is submitted to the server for the name specified in the select.

Web Forms

```
<fieldset>  
  <legend>Demographic information</legend>  
  <fieldset>  
    <legend>Gender</legend>  
    <input type='radio' name='gender' id='male' value='M' />  
    <label for='male'>Male</label>  
    <input type='radio' name='gender' id='female' value='F' />  
    <label for='female'>Female</label>  
  </fieldset>  
</fieldset>
```

For type=radio, you use different ids, but the same name. Only the selected one within the name group will be submitted.

Web Forms

```
<fieldset>
  <legend>Interests</legend>
  <input type='checkbox' name='interests[]' id='games'
value='games' />
  <label for='games'>Games</label>
  <input type='checkbox' name='interests[]' id='reading'
value='reading' />
  <label for='reading'>Reading</label>
  <input type='checkbox' name='interests[]' id='gardening'
value='gardening' />
  <label for='gardening'>Gardening</label>
  <input type='checkbox' name='interests[]' id='sports'
value='sports' />
</fieldset>
</fieldset>
<input type='submit' value='Submit' />
<input type='reset' value='Clear' />
</form>
```

Web Forms

```
<fieldset>
  <legend>Interests</legend>
  <input type='checkbox' name='interests[]' id='games'
value='games' />
  <label for='games'>Games</label>
  <input type='checkbox' name='interests[]' id='reading'
value='reading' />
  <label for='reading'>Reading</label>
  <input type='checkbox' name='interests[]' id='gardening'
value='gardening' />
  <label for='gardening'>Gardening</label>
  <input type='checkbox' name='interests[]' id='sports'
value='sports' />
</fieldset>
</fieldset>
<input type='submit' value='Submit' />
<input type='reset' value='Clear' />
</form>
```

For type=checkbox, you use different ids, but the same name. The name appears as an array (using []). This creates an array on the server.

Web Forms

```
<fieldset>
  <legend>Interests</legend>
  <input type='checkbox' name='interests[]' id='games'
value='games' />
  <label for='games'>Games</label>
  <input type='checkbox' name='interests[]' id='reading'
value='reading' />
  <label for='reading'>Reading</label>
  <input type='checkbox' name='interests[]' id='gardening'
value='gardening' />
  <label for='gardening'>Gardening</label>
  <input type='checkbox' name='interests[]' id='sports'
value='sports' />
  </fieldset>
</fieldset>
<input type='submit' value='Submit' />
<input type='reset' value='Clear' />
</form>
```

An input with type=submit creates a button, as does reset.

PHP Scripts for Form Data

- Can use `$_GET`, `$_POST`, or `$_REQUEST` arrays to access form data.
 - All are arrays with the key equal to the name given to the form element.
 - The values are strings from the data in the form.
 - Multiple-select form elements have array values
 - Multiple select combo-boxes
 - Checkboxes

PHP Scripts for Form Data

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
    <title>Form Elements result</title>
  </head>
  <body>
    <h1>Form Elements result</h1>
    <?php echo dumpArray($_REQUEST); ?>
  </body>
</html>
```

PHP Scripts for Form Data

```
<?php
function dumpArray($elements) {
  $result = "<ol>\n";
  foreach ($elements as $key => $value) {
    if (is_array($value)) {
      $result .= "<li>Key <b>$key</b> is an array
        containing:" . dumpArray($value) . "</li>\n";
    } else {
      $value = nl2br(htmlspecialchars($value));
      $result .= "<li>Key <b>$key</b> has value
        <b>$value</b></li>\n";
    }
  }
  return $result . "</ol>\n";
}
?>
```

PHP Scripts for Form Data

```
<?php
function dumpArray($elements) {
    $result = "<ol>\n";
    foreach ($elements as $key => $value) {
        if (is_array($value)) {
            $result .= "<li>Key <b>$key</b> is an array
                containing:" . dumpArray($value) . "</li>\n";
        } else {
            $value = nl2br(htmlspecialchars($value));
            $result .= "<li>Key <b>$key</b> has value
                <b>$value</b></li>\n";
        }
    }
    return $result . "</ol>\n";
}
?>
```

Recursion: A function calling itself. Similar to a loop, but has an implicit stopping case.

PHP Scripts for Form Data

```
<?php
function dumpArray($elements) {
    $result = "<ol>\n";
    foreach ($elements as $key => $value) {
        if (is_array($value)) {
            $result .= "<li>Key <b>$key</b> is an array
                containing:" . dumpArray($value) . "</li>\n";
        } else {
            $value = nl2br(htmlspecialchars($value));
            $result .= "<li>Key <b>$key</b> has value
                <b>$value</b></li>\n";
        }
    }
    return $result . "</ol>\n";
}
?>
```

htmlspecialchars converts < into <, > into >, & into &, etc.

PHP Scripts for Form Data

```
<?php
function dumpArray($elements) {
    $result = "<ol>\n";
    foreach ($elements as $key => $value) {
        if (is_array($value)) {
            $result .= "<li>Key <b>$key</b> is an array
                containing:" . dumpArray($value) . "</li>\n";
        } else {
            $value = nl2br(htmlspecialchars($value));
            $result .= "<li>Key <b>$key</b> has value
                <b>$value</b></li>\n";
        }
    }
    return $result . "</ol>\n";
}
?>
```

`nl2br` converts newlines into `
` so that text areas maintain their line breaks.

PHP Scripts for Form Data

```
<?php
function dumpArray($elements) {
    $result = "<ol>\n";
    foreach ($elements as $key => $value) {
        if (is_array($value)) {
            $result .= "<li>Key <b>$key</b> is an array
                containing:" . dumpArray($value) . "</li>\n";
        } else {
            $value = nl2br(htmlspecialchars($value));
            $result .= "<li>Key <b>$key</b> has value
                <b>$value</b></li>\n";
        }
    }
    return $result . "</ol>\n";
}
?>
```

Form Elements result

1. Key **firstName** has value **Bill**
2. Key **lastName** has value **Jones**
3. Key **comments** has value **This is my comment that *stretches* over two lines.**
4. Key **inquiryType** has value **customerService**
5. Key **gender** has value **M**
6. Key **interests** is an array containing:
 1. Key **0** has value **reading**
 2. Key **1** has value **sports**

PHP Scripts for Form Data

- Caveats

- The previous code dumped out the entire request, which is rarely what you want to do.
- If you are looking for a specific value, you should use `isset` to see if it was submitted.

PHP Scripts for Form Data

```
function safeGet($array, $key, $default=false) {  
    if (isset($array[$key])) {  
        $value = $array[$key];  
        if (!is_array($value)) {  
            $value = htmlspecialchars(trim($array[$key]));  
        }  
        if ($value) {  
            return $value;  
        }  
    }  
    return $default;  
}  
$interests = safeGet($_REQUEST, 'interests');  
$firstName = safeGet($_REQUEST, 'firstName');
```

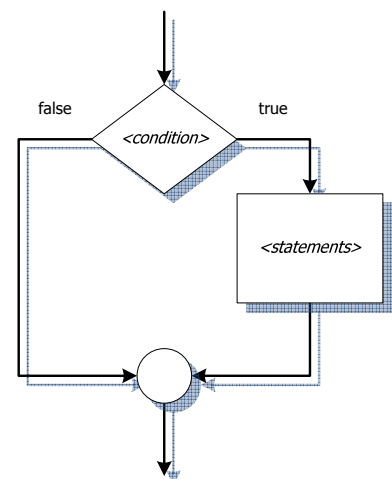
Control Statements

- Control statements alter the flow of a program from straight line to conditional or repeated
 - If/else statements
 - Switch statements
 - For loops
 - Foreach loops
 - While loops
 - Do/while loops

if/else Statements

- Keywords `if` and `else` implement conditional execution

```
if (<condition>) {  
    <statements>  
}
```

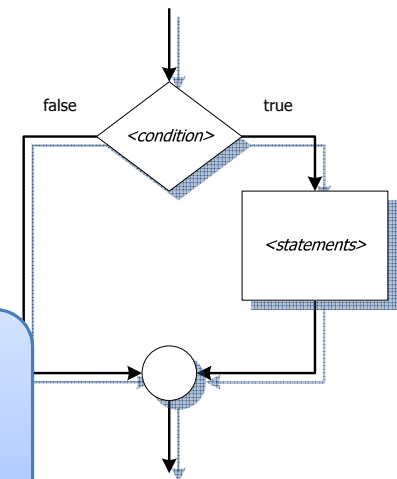


if/else Statements

- Keywords `if` and `else` implement conditional execution

```
if (<condition>) {  
    <statements>  
}
```

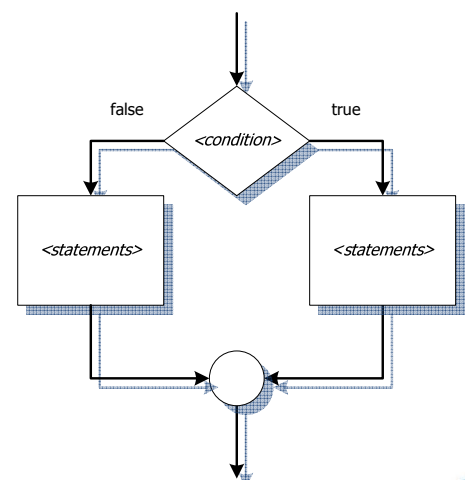
condition should be a boolean expression (after type juggling).



if/else Statements

- Keywords `if` and `else` implement conditional execution

```
if (<condition>) {  
    <statements>  
} else {  
    <statements>  
}
```



switch Statements

- Switch statements
 - A shortcut to compare many values and conditionally execute code based strictly on *equality*.
 - *Good* for a limited number of enumerable options.
 - *Bad* for testing ranges of values, deciding between two mutually exclusive options.

31

switch Statements

- Switch statements – suitability

Example	If/else or switch?
Determining point values of letter grades.	
Determining letter grades from a percentage.	
Determining insurance rates based on age.	
Determine the name of a month based on a month number.	
Determine form letter salutation based on gender.	

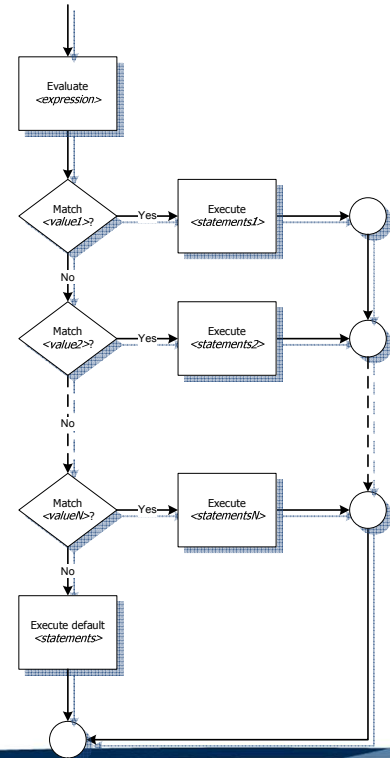
32

switch Statements

- Switch statements

```
switch (<expression>) {  
  case <value1>:  
    <statements1>;  
    break;  
  case <value2>:  
    <statements2>;  
    break;  
  //...  
  case <valueN>:  
    <statements2>;  
    break;  
  default:  
    <statements>;  
    break;  
}
```

What if "break" is missing?

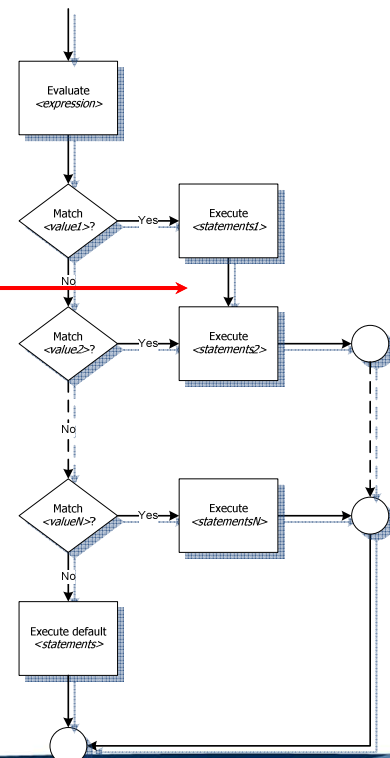


switch Statements

- Switch statements

```
switch (<expression>) {  
  case <value1>:  
    <statements1>;  
  
  case <value2>:  
    <statements2>;  
    break;  
  //...  
  case <valueN>:  
    <statements2>;  
    break;  
  default:  
    <statements>;  
    break;  
}
```

Execution "falls through" to the next case!

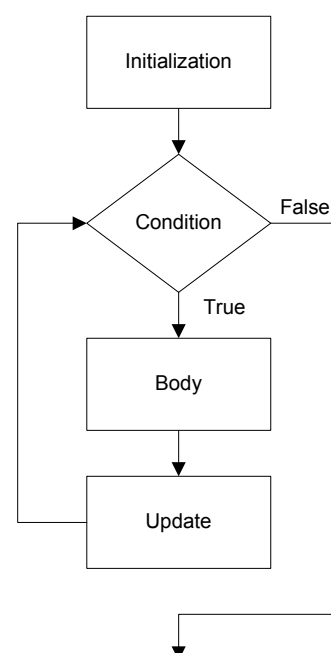


Repetition

- Four parts to every loop
 - Initialization
 - Continuation condition
 - Body
 - Update

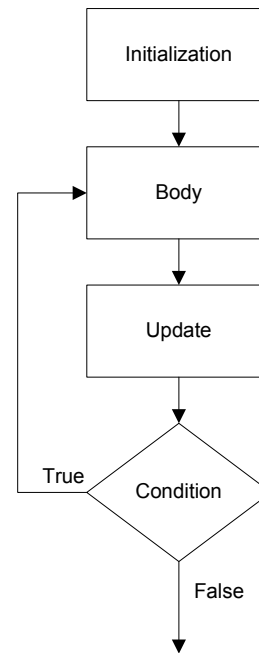
Repetition

- Pre-test loops
 - Condition is evaluated before the body of the loop is executed.
 - Key idea: body may not ever execute.



Repetition

- Post-test loops
 - Condition is evaluated after the body of the loop is executed.
 - Key idea: body always executes at least once



37

while loops

- While loops:
 - Pre-test loop syntax

```
while (condition) {  
    body_statements;  
}
```

All that is really required. But, which of the four parts are missing?

38

while loops

- While loops:
 - Pre-test loop syntax

```
initialization;  
while (condition) {  
    body_statements;  
    update_statement;  
}
```

39

for loops

- For loops:
 - Pre-test loop syntax

```
for (initialization; condition; update) {  
    body_statements;  
}
```

40

for loops

- For loops:
 - Pre-test loop syntax

```
for (initialization; condition; update) {  
    body_statements;  
}
```

Equivalent to:

```
initialization;  
while (condition) {  
    body_statements;  
    update;  
}
```

41

FRANKLIN UNIVERSITY

for vs. while loops

- When to use for vs. while
 - Equivalent at runtime
 - while loops are a little more flexible (i.e. the update step can be conditional or in the middle of the body)
 - for loops are generally used for counting (i.e. the bounds are known)

42

FRANKLIN UNIVERSITY

do...while loops

- do...while loops:
 - Post-test loop syntax

```
do {  
    body_statements;  
} while (condition);
```

```
initialization;  
do {  
    body_statements;  
    update;  
} while (condition);
```

do...while loops

- do...while loops:
 - Post-test loop syntax

```
do {  
    body_statements;  
} while (condition);
```

```
initialization;  
do {  
    body_statements;  
    update;  
} while (condition);
```

Required elements.

All 4 elements.

do...while loops

- Post-test loops
 - Body always guaranteed to execute at least once.
 - But, we could still copy-and-paste the body above a pre-test loop and achieve the same results.

foreach/as loops

- Specifically designed to solve the problem of iterating through *every* element of an array.

```
foreach ($array_var as $value) {  
    do_something;  
}
```

```
foreach ($array_var as $key => $value) {  
    do_something;  
}
```

foreach/as loops

- Specifically designed to solve the problem of iterating through *every* element of an array.

```
foreach ($array_var as $value) {  
    do_something;  
}
```

Assigns values from the array into `$value` each trip through the loop.

```
foreach ($array_var as $key =>  
    do_something;  
}
```

foreach/as loops

- Specifically designed to solve the problem of iterating through *every* element of an array.

Assigns the index into `$key` and the value into `$value` each trip through the loop.

```
foreach ($array_var as $key => $value) {  
    do_something;  
}
```

foreach/as loops

- Example:

```
$alphabet = array(  
    'a' => 'apple',  
    'b' => 'banana',  
    'c' => 'carrot'  
);  
  
foreach ($alphabet as $value) {  
    print "$value<br />";  
}  
  
foreach ($alphabet as $key => $value) {  
    print "$key is for $value<br />";  
}
```

Prints:

apple
banana
carrot

foreach/as loops

- Example:

```
$alphabet = array(  
    'a' => 'apple',  
    'b' => 'banana',  
    'c' => 'carrot'  
);  
  
foreach ($alphabet as $value) {  
    print "$value<br />";  
}  
  
foreach ($alphabet as $key => $value) {  
    print "$key is for $value<br />";  
}
```

Prints:

a is for apple
b is for banana
c is for carrot

foreach/as loops

- Example:

```
$alphabet = array(  
    'a' => 'apple',  
    'b' => 'banana',  
    'c' => 'carrot'  
);  
  
foreach ($alphabet as $value) {  
    print "$value<br />";  
}  
  
foreach ($alphabet as $key => $value) {  
    print "$key is for $value<br />";  
}
```

If `$alphabet` is not an associative array, the keys will be numeric, numbered $[0, n-1]$, where n is the length of the array as determined by `sizeof` or `count`.

foreach/as loops

- Ex: roll 2 dice many times. Find probabilities.

```
function probability($count=10000) {  
    $rolls = array();  
    for ($i = 2; $i <= 12; ++$i) {  
        $rolls[$i] = 0;  
    }  
    for ($i = 0; $i < $count; ++$i) {  
        $die1 = mt_rand(1, 6);  
        $die2 = mt_rand(1, 6);  
        $rolls[$die1 + $die2]++;  
    }  
    foreach ($rolls as $die => $times) {  
        print "$die was rolled $times times, for a  
        probability of ". $times/$count * 100 . "%."  
    }  
}
```

foreach/as loops

- Ex: roll 2 dice many times

```
function probability($rolls) {
    $count = array();
    for ($i = 2; $i <= 12; $i++) {
        $count[$i] = 0;
    }
    for ($i = 0; $i < $rolls; $i++) {
        $roll = rand(1, 6) + rand(1, 6);
        $count[$roll]++;
    }
    foreach ($count as $die => $times) {
        echo "$die was rolled $times times, for a
probability of ". $times/$count * 100 . "%.";
    }
}
```

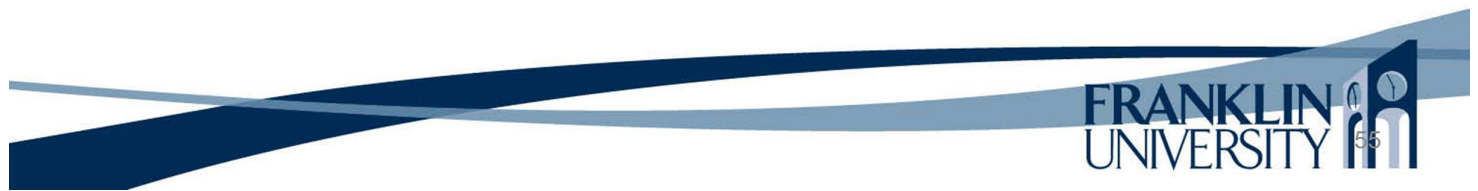
Real probability of rolling a 7 is 6 in 36, or 16.67%.

2 was rolled 262 times, for a probability of 2.62%.
3 was rolled 557 times, for a probability of 5.57%.
4 was rolled 847 times, for a probability of 8.47%.
5 was rolled 1183 times, for a probability of 11.83%.
6 was rolled 1332 times, for a probability of 13.32%.
7 was rolled 1658 times, for a probability of 16.58%.
8 was rolled 1333 times, for a probability of 13.33%.
9 was rolled 1132 times, for a probability of 11.32%.
10 was rolled 834 times, for a probability of 8.34%.
11 was rolled 579 times, for a probability of 5.79%.
12 was rolled 283 times, for a probability of 2.83%.

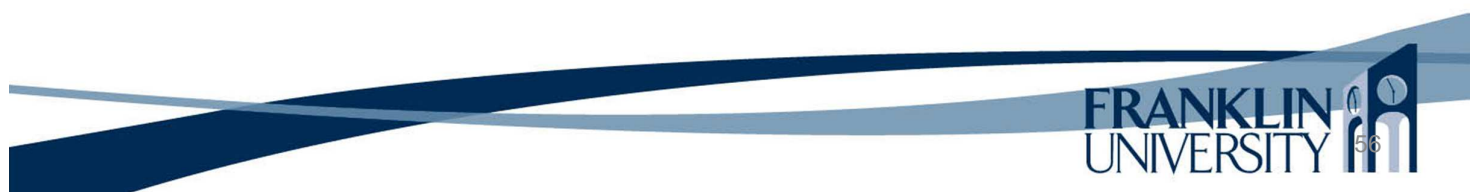
Upcoming Deadlines

- Readings for next week
 - Chapters 9 and 10 in *PHP and MySQL*
- Assignments
 - Homework 3 due January 29
 - Lab 2 due February 12
- Next week:
 - Strings, numbers, and dates

Solution to HW 2



Solution to Lab 1



General Q & A

- Questions?
- Comments?
- Concerns?