How I approach Trial Lab Test 1 Set B

[**First step**](#_8xjzxvnqlwr2) **1**

[**Q1 Part A**](#_hqpjt8w1bdlr) **1**

[**Q1 Part B**](#_5y818foh7unh) **3**

[**Q2 Part A**](#_uxa1pbygohta) **6**

[**Q2 Part B**](#_fra8p610a3of) **7**

[**Q2 Part C**](#_9qnefyexedll) **8**

[**Q3 Part A**](#_cpgnegvf9lo8) **10**

[**Q3 Part B**](#_z3p8hxebmmk7) **11**

[**Q3 Part C**](#_i76fndjfflep) **13**

# Download Files (used in this tutorial)

Click [here](https://smu.sg/2019-is113-trialLT1SetB-approach)

# First step

For all assessments, I will go through the whole paper first to decide which question I want to tackle first and approximately how much time I need on each question. Personally, I will solve questions that I’ve confidence in first to ensure I have sufficient time to work on them properly so as to secure the marks. I will tackle questions that I’m unsure of later.

My aim is to get as many marks as possible within the time limits. It is not to attempt all questions.

# Q1 Part A

Enter name and email first.

|  |
| --- |
| <!DOCTYPE html>  <!--  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  -->  <html> ... |

* *Modify* ***q1-A.html*** *such that if the user clicks on the words ("Trump", "Clinton", "Kim", or "Moon"), the corresponding radio button options will be selected/unselected.*

|  |
| --- |
| …  <form method='post' action='q1-A.php'>  <label> <input type='radio' name='person' value='trump'>Trump </label>  <label> <input type='radio' name='person' value='clinton'>Clinton  </label>  <label> <input type='radio' name='person' value='kim'>Kim </label>  <label> <input type='radio' name='person' value='moon'>Moon </label>  <br><br>  <input type='submit'>  </form>  ... |

* *2. Once the user selects an option (e.g. a person’s name) and click on the SUBMIT button on page* ***q1-A.html****, the form submits to* ***q1-A.php****.*

The given resource has already done this correctly.

**Test** that the form is submitted to q1-A.php. Don’t let the bugs accumulate.

* *3. Modify q1-A.php such that appropriate message will be displayed at the top of the page using Heading-1.*
* *○ For example, if the user selects “Trump” in q1-A.html, then q1-A.php must display:*
* **

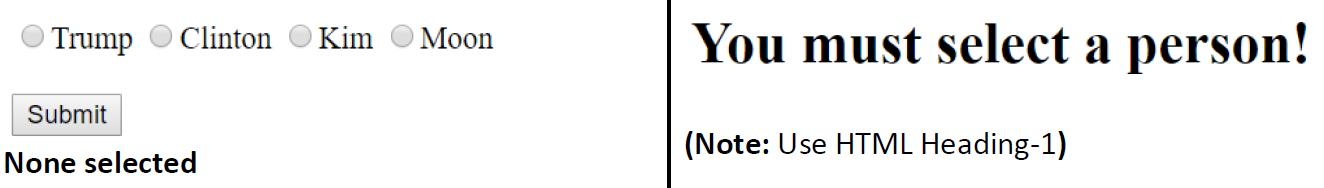
|  |
| --- |
| <?php  /\*  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  \*/  $messages = [  "trump" => "Make America Great Again",  "clinton" => "More Women in Office",  "kim" => "Nukes Fly High and Far",  "moon" => "One Korea One People"  ];  // default  $msg = '';  // process form  if ( isset($\_POST['person'])) {  $name = $\_POST['person'];  $msg = $messages[ $name ];  }  ?>  <!DOCTYPE html>  <html>  <body>  <h1> <?=$msg?> </h1>  </body>  </html> |

**Test**. Don’t let the bugs accumulate.

* *4. Modify q1-A.php such that appropriate image will be displayed.*
* *○ For example, if the user selects “Trump” in q1-A.html, then q1-A.php must display the image file trump.jpg.*

|  |
| --- |
| ...  // default  $msg = '';  $img = '';  // process form  if ( isset($\_POST['person'])) {  $name = $\_POST['person'];  $msg = $messages[ $name ];  $img = "<img src='$name.jpg' />";  }  ?>  <!DOCTYPE html>  <html>  <body>  <h1> <?=$msg?> </h1>  <?=$img?>  </body>  </html> |

**Test**. Test all the given test cases and more that you can think of. Don’t let the bugs accumulate.

* 

This test case is not satisfied.

|  |
| --- |
| ...  if ( isset($\_POST['person'])) {  ...  } else {  $msg = 'You must select a person!';  }  ?>  ...  <h1> <?=$msg?> </h1>  ... |

**Test**. Test all the given test cases and more that you can think of. Don’t let the bugs accumulate.

# Q1 Part B

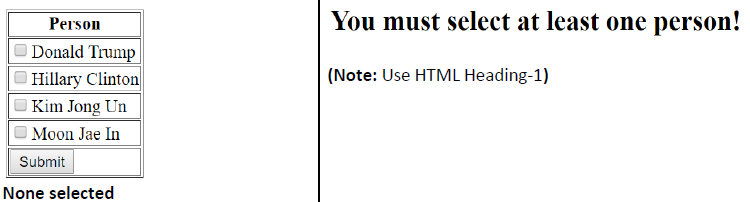
* *1. Modify q1-B.php such that it displays people from the associative array $people. It displays them in an HTML table as shown below.*
* *Each person is displayed in a new table row as a checkbox option.*
* *For example, for “Donald Trump”:*
* *● The checkbox option value is "trump".*
* *● The label/text associated with this checkbox option is "Donald Trump".*
* *● This information can be retrieved from the associative array $people.*

|  |
| --- |
| <?php  /\*  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  \*/  ...  <form method='post' action='q1-B-display.php'>  <table border='1'>  <tr>  <th>Person</th>  </tr>  <?php  foreach ($people as $name => $display) {  echo "  <tr>  <td>  <label>  <input type='checkbox' name='persons[]'  value='$name'>$display  </label>  </td>  </tr>  ";  }  ?>  ... |

**Test**. Don’t let the bugs accumulate.

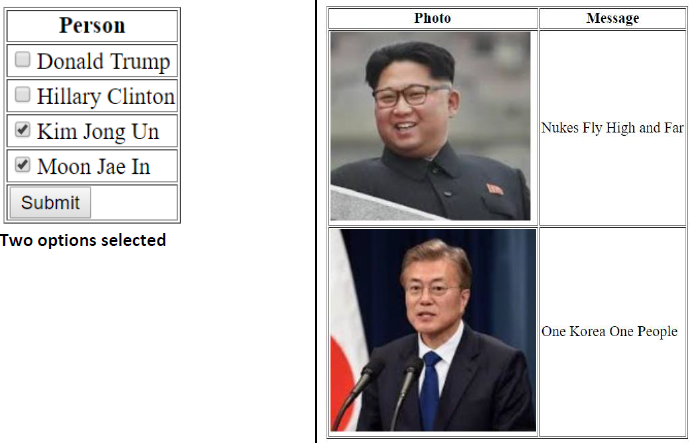
* *2. Once the user selects people and clicks on the SUBMIT button on page q1-B.php, the form submits to q1-B-display.php.*

The given resource has already done this correctly.

* *3. Modify q1-B-display.php such that appropriate messages AND images will be displayed in an HTML table.*
* **

|  |
| --- |
| <?php  /\*  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  \*/  ...  // default  $msg = '';  $persons = [];  // process form  if ( isset($\_POST['persons'])) {  $persons = $\_POST['persons'];  } else {  $msg = 'You must select at least one person!';  }  ?>  <!DOCTYPE html>  <html>  <body>  <?php  if ( $msg == '') {  var\_dump($persons);  } else {  echo "  <h1>$msg</h1>  ";  }  ?>  </body>  </html> |

**Test**. Don’t let the bugs accumulate.

* 

|  |
| --- |
| ...  if ( $msg == '') {  echo "  <table border='1'>  <tr>  <th>Photo</th>  <th>Message</th>  </tr>  ";    foreach($persons as $name) {  echo "  <tr>  <td>  <img src='$name.jpg' />  </td>  <td> {$messages[$name]} </td>  </tr>  ";  }  echo "  </table>  ";  } else {  echo "  <h1>$msg</h1>  ";  }  ... |

**Test**. Test all the given test cases and more that you can think of. Don’t let the bugs accumulate.

# Q2 Part A

* *1. Complete page q2-calculate.php such that it retrieves the following user input from the form.*
* *○ quantity*
* *○ lucky\_number*
* *○ bet\_amount*
* *2. The page then must display the information at the top of the page using HTML Heading-3.*

|  |
| --- |
| <?php  /\*  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  \*/  ...  // Form Processing  // YOUR CODE GOES HERE  $quantity = $\_POST['quantity']; // notice the given code uses this variable  $lucky\_number = $\_POST['lucky\_number']; // notice the given code uses this variable  $bet\_amount = $\_POST['bet\_amount'];  // Generate # of sets (each set contains 3 numbers)  $random\_sets = generateRandomSets($quantity); // DO NOT MODIFY THIS LINE  // Check if lucky number is found  $result = calculate($random\_sets, $lucky\_number); // DO NOT MODIFY THIS LINE  …  <html>  <body>  <h3>Lucky Number: <?=$lucky\_number?> </h3>  <h3>Bet Amount: <?=$bet\_amount?> </h3>  </body>  </html> |

The question didn’t say anything about checking the submitted values, thus, I didn’t do any validation checks.

# Q2 Part B

* *1. Inside q2-calculate.php, complete the function generateRandomSets($quantity).*
* *2. The function takes ONE (1) parameter $quantity, which is a user-defined value (from Part A).*
* *3. The function generates $quantity number of number sets, where:*
* *○ Each number set consists of THREE (3) integers and;*
* *○ Each integer ranges between ZERO (0) and NINE (9), both numbers inclusive.*
* *4. Please see the partial implementation of the function inside the file for details of what the return value looks like.*
* *HINT: Explore the use of PHP rand function*

I read the document for rand() function.

If called without the optional min, max arguments rand() returns a pseudo-random integer between 0 and getrandmax(). If you want a random number between 5 and 15 (inclusive), for example, use rand(5, 15).

|  |
| --- |
| function generateRandomSets($quantity) {  $num\_numbers = 3; // generate 3 random integers  $result = [];  ...    for ( $i = 0; $i < $quantity; $i++) {  /\*  Even though the question stated 3 random integers,  since the resource provides $num\_numbers, I will use the variable  instead of hardcoding to 3  \*/  $set = [];  for ($x = 0; $x < $num\_numbers; $x++) {  $set[] = rand(0,9);  }    $result[] = $set;  }  // var\_dump($result); // Test your own code by using var\_dump  return $result;  } |

# 

**Test**. Even if you are not given test cases, think up your own test cases and do your own testing. Don’t let the bugs accumulate.

# Q2 Part C

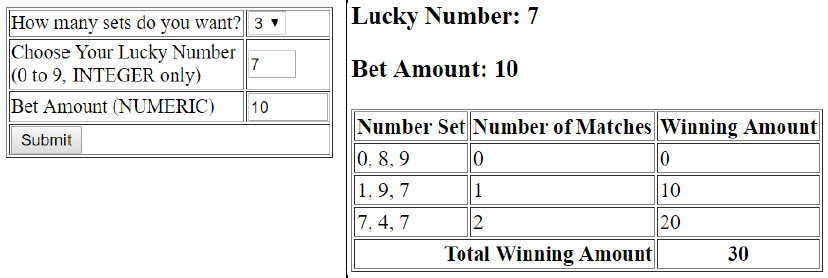
* *1. Complete q2-calculate.php. Inside q2-calculate.php, complete the function calculate($random\_sets, $lucky\_number).*
* *2. The function takes TWO (2) parameters:*
* *○ $random\_sets is the value returned by the function generateRandomSets() (from Part B).*
* *■ It is an (indexed) array of (indexed) arrays.*
* *○ $lucky\_number is the user-defined form input (from Part A).*
* *3. The function checks each number set and calculates number of matches for that number set.*
* *4. Please see the partial implementation of the function inside the file for details of what the return value looks like.*

|  |
| --- |
| function calculate($random\_sets, $lucky\_number) {  $result = [];  $num\_numbers = 3; // each set consists of 3 randomly generated integers  ...  // Part B  // YOUR CODE GOES HERE  // I create my own $random\_sets for my own testing  // $random\_sets = [  // [0, 2, 3],  // [0, 2, 7],  // [7, 2, 7],  // [7, 7, 7],  // ]; // for testing  foreach ($random\_sets as $set) {  $count = 0;  foreach ($set as $num) {  if ( $num == $lucky\_number) {  $count++;  }  }  $result[] = $count;  }  // var\_dump($random\_sets);  // var\_dump($lucky\_number);  // var\_dump($result);  return $result;  } |

**Test**. When actual values are random, create your own hardcoded values for testing. Don’t let the bugs accumulate.

* *The user also chose THREE (3) sets of numbers in q2.html.*
* *● The user chose number 7 as the lucky\_number in q2.html with the bet\_amount of 10 in q2.html.*
* *● The user clicks on the SUBMIT button.*
* *● Page q2-calculate.php generated THREE (3) sets of numbers, where each set consists of THREE (3) randomly selected integers.*
* *● q2-calculate.php displays the user input lucky\_number and bet\_amount at the top of the page.*

All these were done.

* *● For each number set, the page calculates winning amount and displays it in an HTML table.*
* *○ Above, the user’s second number set (1, 9, 7) contains lucky\_number (7) ONCE. Hence, the winning amount for this set is 10 (bet\_amount of 10 times 1, which is the number of occurrences of the lucky\_number).*
* *○ The third number set (7, 4, 7) contains lucky\_number (7) TWICE. Hence, the winning amount for this set is 20 (bet\_amount of 10 times 2).*
* *○ Finally, the Total Winning Amount is calculated to be 30 as it is the sum of all winning amounts.*
* *HINT: Explore the use of PHP implode function for displaying numbers in “Number Set” column in the above table*
* **

Basically, the sample output says it all that we need to do.

I read the documentation for implode() function. I find the example easier to understand this time.

Example #1 implode() example

<?php

$array = array('lastname', 'email', 'phone');

$comma\_separated = implode(",", $array);

echo $comma\_separated; // lastname,email,phone

…

|  |
| --- |
| …  <html>  <body>  <h3>Lucky Number: <?=$lucky\_number?> </h3>  <h3>Bet Amount: <?=$bet\_amount?> </h3>  <table border='1'>  <tr>  <th>Number Set</th>  <th>Number of Matches</th>  <th>Winning Amount</th>  </tr>  <?php  $total = 0;  for ($i = 0; $i < $quantity; $i++) {  // From pt 3 below: Determine values of variables before the echo  $set = $random\_sets[$i];  $num\_set = implode(',', $set);  $matches = $result[$i];  $amt = $bet\_amount \* $matches;    $total += $amt;  // 1. Display HTML required  // 1a. Parts that repeat, use a loop  // 2. Introduce variables for the parts that change  // 3. Determine values of variables before the echo  echo "  <tr>  <td>$num\_set</td>  <td>$matches</td>  <td>$amt</td>  </tr>  ";  }  ?>  <tr>  <th colspan='2'>  Total Winning Amount  </th>  <th>  <?=$total?>  </th>  </tr>  </table>  </body>  </html> |

**Test**. When actual values are random, create your own hardcoded values for testing. Don’t let the bugs accumulate.

# Q3

3-star questions tend to require a lot of time (tedious or complicated) to solve but the marks are little. Low ROI. My approach is to solve as many little parts as I can when I am sure I have solved all the other questions that I’m confident in.

## Q3 Part A

* *Complete q3.php such that it displays people’s names in an HTML table format as shown below:*
* *● Do NOT HARD-CODE the names.*
* *● Make use of $people associative array provided in the resource file.*

Same as Q1 part B. Free marks!

|  |
| --- |
| <?php  /\*  Name: Wilson Alexander Daniel the First  Email: wad1@smu.edu.sg  \*/  ...  ?>  <!DOCTYPE html>  <html>  <body>  <form method='post' action='q3.php'>    <table border='1'>  <tr>  <th>Person</th>  </tr>  <?php  foreach ($people as $name => $display) {  echo "  <tr>  <td>  <label>  <input type='checkbox' name='persons[]' value='$name'>$display  </label>  </td>  </tr>  ";  }  ?>  <tr>  <td>  <input type='submit'>  </td>  </tr>  </table>  </form>  </body>  </html> |

Hmmm… I don’t even need to change the code.

**Test**. Don’t let the bugs accumulate.

## Q3 Part B

* *After the user makes selections and clicks the SUBMIT button in q3.php, the form submits back to q3.php.*
* *Complete q3.php such that it displays appropriate messages at the top of the page using Heading-1.*
* *● If no one is selected, display "You didn’t select anyone! Select at least THREE (3) people!"*
* *● If one or two people are selected, display "Select at least THREE (3) people!"*

|  |
| --- |
| ...  // default  $msg = '';  $persons = [];  // process form  if ( isset( $\_POST['btnSubmit'])) {  /\*  I named the submit button so that I can use it to check  is this page accessed for the first time or  is there a form submission.  \*/  if ( isset($\_POST['persons'])) { // 1 or more checkboxes ticked    $persons = $\_POST['persons'];  $num\_persons = count($persons);  if ( $num\_persons < 3) {  $msg = "Select at least THREE (3) people!";  } // else {  // $msg = 'OK'; // for testing  // }  } else { // no checkbox ticked  $msg = "You didn’t select anyone! Select at least THREE (3) people!";  }  }  ?>  <!DOCTYPE html>  <html>  <body>  <h1><?=$msg?></h1>  ...  <input type='submit' name='btnSubmit'>  ... |

**Test**. Don’t let the bugs accumulate.

* *● Page q3.php must remember the user’s original selections (if any) and display it correctly.*

|  |
| --- |
| ...  <?php  foreach ($people as $name => $display) {  $checked = in\_array($name, $persons) ? 'checked' : '';  // 2. Introduce variables for the parts that change  // 3. Determine values of variables before the echo  echo "  <tr>  <td>  <label>  <input type='checkbox' name='persons[]' value='$name'  $checked>$display  </label>  </td>  </tr>  ";  }  ?>  ... |

**Test**. Don’t let the bugs accumulate.

## Q3 Part C

* *If the user selects THREE (3) or more people in q3.php as shown below:*
* *and clicks on the SUBMIT button, q3.php must display:*
* *● An HTML table listing all people where the user’s selections are checked.*
* *● An HTML table showing the selected people’s images.*
* *○ It is an N by N matrix.*
* *○ The displayed images are selected randomly with an equal probability of each person being selected.*
* *HINT: Explore the use of PHP rand function*

|  |
| --- |
| ...  // default  ...  $num\_persons = 0;  // process form  ...  $persons = $\_POST['persons'];  $num\_persons = count($persons);  ...  <table border='1'>  <?php  // 1. Display HTML required  // 1a. Parts that repeat, use a loop  // 2. Introduce variables for the parts that change  // 3. Determine values of variables before the echo  for ($r = 0; $r < $num\_persons; $r++) {  echo '  <tr>  ';  for ($c = 0; $c < $num\_persons; $c++) {  $name = $persons[ rand(0, $num\_persons-1)];  echo "  <td>  <img src='$name.jpg' />  </td>  ";  }  echo '  </tr>  ';  }  ?>  </table>  ... |

**Test**. Don’t let the bugs accumulate.

* *Continuing from Part C, modify q3.php to check if all of the displayed images on the diagonal are the same. Specifically,*
* *1) All images on the diagonal from the Top Left Corner to the Bottom Right Corner are the same (showing the same person), OR*
* *2) All images on the diagonal from the Top Right Corner to the Bottom Left Corner are the same (showing the same person)*
* *Display the following messages if 1) or 2) are satisfied, using HTML Heading-1.*
* *● If 1) is satisfied, display "Top Left to Bottom Right Diagonal FOUND".*
* *● If 2) is satisfied, display "Top Right to Bottom Left Diagonal FOUND".*

|  |
| --- |
| ...  <table border='1'>  <?php  // default values  $topLeftName = '';  $topRightName = '';  $topLeftBottomRight = true;  $topRightBottomLeft = true;  for ($r = 0; $r < $num\_persons; $r++) {  echo '  <tr>  ';  for ($c = 0; $c < $num\_persons; $c++) {  $name = $persons[ rand(0, $num\_persons-1)];  // $name = 'trump'; // testing  echo "  <td>  <img src='$name.jpg' />  </td>  ";    // check diagonals  if ( $r == 0 ) {  // 1st row; remember the corners  if ( $c == 0 ) {  $topLeftName = $name;  }  if ( $c == $num\_persons-1) {  $topRightName = $name;  }  } else {  // 2nd row onwards  // if this is on the diagonal, check against the remembered corner  if ( $topLeftBottomRight && $r == $c) {  $topLeftBottomRight = $topLeftName == $name;  }  if ( $topRightBottomLeft && $r == ($num\_persons-1 - $c) ) {  $topRightBottomLeft = $topRightName == $name;  }  }  }  echo '  </tr>  ';  }  ?>  </table>  <?php  // display  if ( $topLeftBottomRight ) {  echo '  <h1>Top Left to Bottom Right Diagonal FOUND</h1>  ';  }  if ( $topRightBottomLeft ) {  echo '  <h1>Top Right to Bottom Left Diagonal FOUND</h1>  ';  }  ?>  ... |

Note, there are other ways to solve this.

# Summary

1. Read through the whole paper first.
2. Aim to get as many marks as possible within the time limits.
3. Break down a complex problem into smaller parts.
4. Solve one small part at a time.
5. Test a lot and frequently. Don’t let the bugs accumulate.