

### Lab 6: The Right Rectangular Pyramid Application


#### General Description

Note: Due to the client-side script involved in this application, live demo is not available.

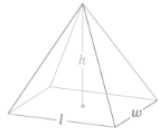
The focus of this lab is OOP inheritance and code reuse. WHEN DESIGNING A CLASS, YOU SHOULD NEVER REPEAT YOURSELF FOR CODE THAT IS ALREADY AVAILABLE IN A PARENT CLASS.

In this lab, you will create a PHP application that calculates the base area, volume, lateral surface, and surface area of a right rectangular pyramid. A user enters the width and length of the base and the height of the pyramid, the application displays all details of the pyramid. Since AJAX is enabled, the width, length, and height are submitted to the server automatically when a user is entering them, and the browser updates the Web page when server's response has arrived. The width, length, and height of a pyramid can only accept positive numbers. If any value is invalid, the Web page clears the details of the pyramid.

Please see the following screenshots for sample output. The screenshot on the left-hand side shows an output when valid width, length, and height are entered and the screenshot on the right-hand side shows an output when invalid values are entered.

The Right Rectangular Pyramid Application	
Enter width, length, and height:	
Width (w):	<input type="text" value="10"/>
Length (l):	<input type="text" value="8"/>
Height (h):	<input type="text" value="12"/>
	
Base Area:	<input type="text" value="80.00"/>
Volume:	<input type="text" value="320.00"/>
Lateral Surface:	<input type="text" value="230.49"/>
Surface Area:	<input type="text" value="310.49"/>

The Right Rectangular Pyramid Application	
Enter width, length, and height:	
Width (w):	<input type="text" value="10"/>
Length (l):	<input type="text" value="8"/>
Height (h):	<input type="text" value="-12"/>
	
Base Area:	<input type="text"/>
Volume:	<input type="text"/>
Lateral Surface:	<input type="text"/>
Surface Area:	<input type="text"/>

#### The Composition

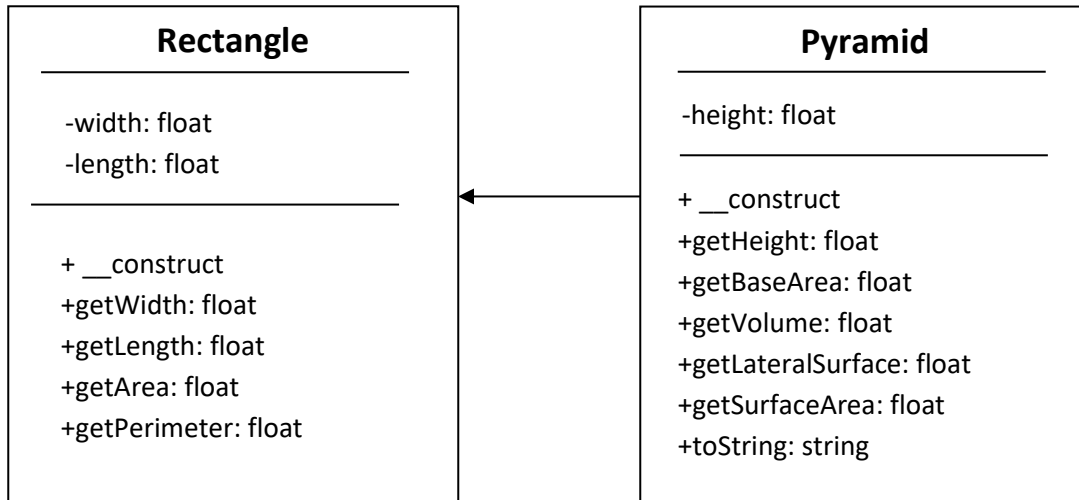
The application is composed of the following script files:

- *index.php* presents the interface of the application. Pay attention to the ids of the three text boxes that accept user inputs and the four div blocks that display details of a pyramid.
- *rectangle.class* file defines the base class named **Rectangle**.
- *pyramid.class.php* file defines the class named **Pyramid**.
- *pyramid\_do.php* handles server-side tasks including creating OOP objects and sending server's responses.

- *main.js* handles client-side tasks including sending AJAX requests and handling server's responses.
- *pyramid.png* is the image file.

### UML Diagrams of the OOP Classes

The following shows the UML diagrams of the two classes: **Circle** and **Pyramid**.



### Step-by-Step Instructions

1. Download the data files from Canvas and extract the **Lab06** folder into **htdocs/I211** folder.

#### Creating the *rectangle.class.php* file and defining the **Circle** class

2. Inside the **Lab06** folder, create a new PHP file named *rectangle.class.php* and then define a new class named **Rectangle**. Define the **Rectangle** class according to the UML diagram above.
3. The constructor accepts two parameters and assigns them to the data members of the class.
4. The four **get** methods return the width, length, area, and perimeter of a **Rectangle** object. Use the following formulas to calculate the area and perimeter.
  - Area: width \* length.
  - Perimeter: 2 \* (width + length).

#### Creating the *pyramid.class.php* file and defining the **Pyramid** class

5. Create the **Pyramid** class and save it in the file named *pyramid.class.php*. This class should inherit from the **Rectangle** class. Define the **Pyramid** class according to the UML diagram above.
6. The constructor of the class accepts three parameters. It also overrides the constructor inherited from the **Rectangle** class. Remember to use the keyword **parent** and scope resolution operator (**::**) to access code defined in the parent class.

7. The five **get** methods return the height, base area, volume, lateral surface, and surface area of the Pyramid. Use the following explanation and formulas for calculation. If code is already available in the parent class, you should reuse it and never repeat yourself.
- Base area: the area of a base, which is a rectangle.
  - Volume: base area \* height / 3.
  - Lateral surface area: the sum of the four triangles that form the side portions of the shape. Use the following formula to determine the lateral surface area. Please note you may use **sqrt()** and **pow()** functions in PHP to determine square root and square.
- $$A_L = l\sqrt{\left(\frac{w}{2}\right)^2 + h^2} + w\sqrt{\left(\frac{l}{2}\right)^2 + h^2}$$
- Surface area: the total area of the surface, lateral surface + base area.
8. The **toString** method returns a string representation of a JSON object that contains details of the **Pyramid** object. Use these names in the JSON object: **Width**, **Height**, **Base**, **Volume**, **Lateral**, and **Surface**. To format a number in PHP, you may use the **number\_format** function. The following is an example of a JSON string:

```
{ "Width": "6.20", "Length": "10.00", "Height": "8.60", "Base": "62.00", "Volume": "177.73", "Lateral": "153.09", "Surface": "215.09" }
```

### Modifying the *main.js* file to complete the **calculate()** function

The code provided to you in the file creates an **XMLHttpRequest** object when the window's load event occurs. You only need to complete the **calculate()** function. The function gets called whenever a keyup event occurs in anyone of the three input textboxes. The function carries out following tasks in sequence:

9. Retrieve user's inputs for width, length, and height from the three textboxes.
10. If any one of the three user inputs is invalid (i.e. non-numeric or negative value), clear the four div blocks that display a pyramid's details (see screenshots on page 1); if all inputs are valid (i.e. positive numbers), send an asynchronous AJAX request and handle server's responses.
  - a. Use the **GET** method to send the AJAX request.
  - b. The server-side script that accepts the request is called *pyramid\_do.php*.
  - c. Send the width, length, and height you retrieved previously along with the AJAX request.
  - d. To handle the server's response, parse the JSON object returned from the server and display a pyramid's base area, volume, lateral surface, and surface area with the four div blocks.

### Creating the *pyramid\_do.php* file

This file handles the server-side scripting of the AJAX request. It carries out three tasks in sequence: retrieve the width, length, and height in query string variables sent in a AJAX request, create a **Pyramid** object, and echo the output by calling the **toString** method with the **Pyramid** object.

11. Create a new PHP file and save it as *pyramid\_do.php*.
12. Add your code that carries out the tasks described above.

### Code style guidelines:

1. Code style is as important as the code itself. Code style includes enough comments, adequate space between code blocks, and proper indentation, etc. Read details and view sample code from <http://pear.php.net/manual/en/standards.php>.
2. Please provide sufficient comments in your source code. Source code is a language for people, not just for computers. Comment as you go and don't wait for later. Ask yourself: "How will the next person know that?" Commenting code shows your professionalism, but also helps your grader understand your code.
3. Every file should contain a header in this format:

```
/*  
 * Author: your name  
 * Date: today's date  
 * Name: file name  
 * Description: short paragraph that describes and explains the file  
*/
```
4. Indent your code. Leave enough space between code blocks.

### Turning in your lab

Your work will be evaluated on completeness and correctness. Thoroughly test your code before you turn it in. It is your responsibility to ensure you turn in the correct files. You will NOT receive any credit if you turn in the wrong files whether or not you've completed the lab.

1. Zip the entire **Lab06** folder and save it as *Lab06.zip*.
2. Upload the *Lab06.zip* file in Canvas before the lab's deadline.

### Grading rubric

Your TAs will assess your lab according to the following grading rubric. You should very closely follow the instructions in this handout when working on the lab. Small deviations may be fine, but you should avoid large deviations. You will not receive credits if your deviation does not satisfy an item of the grading rubric. Whether a deviation is small or large and whether it satisfies the requirement are at your TAs' discretion. Here is the breakdown of the scoring:

Creating the **Rectangle** class (2 points)

Activities	Points
Define the private data	0.5
Define the constructor	0.5

## I211 Information Infrastructure II

---

Create the four <b>get</b> methods	1
------------------------------------	---

Creating the **Pyramid** class (8 points)

Activities	Points
Define the private data	0.5
Define the constructor	0.5
Create the five <b>get</b> methods	5
Create the <b>toString</b> method	2

Creating the **pyramid\_do.php** file (2 points)

Activities	Points
Retrieve width, length, and height from query string variables	0.5
Create a <b>Pyramid</b> object	0.5
Call the <b>toString</b> method and echo the output	1

Modifying the main.js file to complete the **calculate** function (3 points)

Activities	Points
Define (open) and send an asynchronous AJAX request	0.5
Handle server's responses to display details of a pyramid	2
Clear the div blocks	0.5

Programming style (5 points)

Activities	Points
Comment your code	3
Use white spaces to separate code sections	1
Indent and line up code	1