College of Engineering, Construction & Living Sciences
Bachelor of Information Technology
IN607: Introductory Application Development Concepts
Level 6, Credits 15
In-Class Activity: PHP Basics 2

Instructions

The purpose of this in-class activity to familiarise yourself with classes, inheritance & interfaces in **PHP**.

Code Review

You must submit all program files via **GitHub Classroom**. Here is the URL to the repository you will use for your code review – https://classroom.github.com/a/P656imf2. Checkout from the **main** branch to the **02-in-class-activity** branch by running the command - **git checkout 02-in-class-activity**. This branch will be your development branch for this activity. Once you have completed this activity, create a pull request & assign the **GitHub** user **grayson-orr** to a reviewer. **Do not** merge your own pull request.

Problem 1:

Create a **Cat** class with the private attributes name, age & breed. For each attribute, create a getter & setter. Also, create a __**toString()** special method which returns the following:

```php
<?php
    "My " . $this->breed . " 's name is "
        . $this->name. ". S/he is " . $this->age . " year(s) old.";
?>
```

Create two **Cat** objects called **cat_one** & **cat_two**. Using setters, change **cat_one's** name to Fido & age to 10. Again, using a setter, change **cat_two's** breed to **American Bobtail**. For each **Cat** object, print its string representation.

```php
<?php
    // Write your solution here
?>
```

Problem 2:

Create a **SoftwareDeveloper** & **AgileCoach** class which inherits from **Employee** class. **SoftwareDeveloper** class has one additional class attribute called **prog_lang**. **AgileCoach** also has one additional class attribute called **employees** & three class methods which **add**, **remove**, **search** & **show_all** employees managed by **AgileCoach**. **Note:** employees is a list of **SoftwareDeveloper** objects.

Use the three **SoftwareDeveloper** objects & **AgileCoach** object provided to display the expected output.

```php
<?php
    class Employee {
        protected $first_name;
        protected $last_name;
        protected $salary;

        public function __construct($first_name, $last_name, $salary) {
            $this->first_name = $first_name;
            $this->last_name = $last_name;
            $this->salary = $salary;
        }

        public function __toString() {
            return $this->first_name . " " . $this->last_name;
        }
    }

    $sft_dev_one = new SoftwareDeveloper("Alfredo", "Boyle", 50000, "C#");
    $sft_dev_two = new SoftwareDeveloper("Malik", "Martin", 55000, "JavaScript");
    $sft_dev_three = new SoftwareDeveloper("Livia", "Martin", 75000, "Kotlin");
    $agile_coach = new AgileCoach("Lillian", "Cunningham", 100000, array($sft_dev_one, $sft_dev_two));

    // Write your solution here

    // Expected output:
    // Malik Martin
    // Livia Martin
    // Alfredo Boyle not found
    // Livia Martin found
?>
```

Problem 3:

Language class has no class attributes, but a class method called **good_morning**. **Maori**, **Spanish** & **German** class inherit from **Language** class. When you run the following code, what is happening & why is it happening? Refactor the code to display the expected output.

```php
<?php
    class Language {
        public function good_morning() {
            throw new Exception("good_morning() function not implemented");
        }
    }

    class Maori extends Language {
        public function good_morning() {
            echo "Morena";
        }
```

```php
    }

    class Spanish extends Language {

    }

    class German extends Language {
        public function good_morning() {
            echo "Guten Morgen";
        }
    }

    $maori = new Maori();
    $spanish = new Spanish();
    $german = new German();
    $maori->good_morning();
    $spanish->good_morning();
    $german->good_morning();

    // Expected output:
    // Morena
    // Hola
    // Guten Morgen
?>
```

Problem 4:

Implement the **push()**, **pop()**, **peek()**, **size()** & **show_all()** methods in the **Stack** class. **Note: size()** method returns the length of a **stack** & **show_all()** method returns the items in a **stack**.

You are probably wondering what a **stack** is. A **stack** is an **Abstract Data Type (ADT)** that is used to store elements in a **Last In First Out (LIFO)** manner. A **stack** is a collection of elements where the addition of elements is performed at the end & the removal of elements is performed at the beginning. A **stack** has two methods: **push()** & **pop()**. The push method adds an element to the **stack**. The pop method removes the last element added to the **stack**.

Use the **Stack** object provided to display the expected output.

```php
<?php
    class Stack {
        private $stack;

        public function __construct() {
            $this->stack = array();
        }

        public function push($item) {
            // Write your code here
        }

        public function pop() {
            // Write your code here
        }

        public function peek() {
            // Write your code here
```

```php
        }

        public function is_empty() {
            // Write your code here
        }

        public function size() {
            // Write your code here
        }

        public function show_all() {
            // Write your code here
        }

        public function __toString() {
            return $this->stack;
        }
    }

    $stack = new Stack();
    $stack->push("Introductory App Dev Concepts");
    $stack->push("Intermediate App Dev Concepts");
    $stack->push("Advanced App Dev Concepts");

    // Write your solution here

    // Expected output:
    // ["Introductory App Dev Concepts", "Intermediate App Dev Concepts"]
    // Intermediate App Dev Concepts is at the top of the stack
    // There are 2 item(s) in the stack
?>
```

Problem 5:

Create an interface called **App**. This interface has three methods, **login()**, **register()** & **logout()**. The **login()** method accepts two arguments, email & password, & the **register()** method accepts three arguments, email, password & username.

Create a class called **Facebook** which implements **App**. For each method implemented, echo the following:

```php
// login() - Logged in with the email - $email.
// register() - Registered with the email and username - $email and $username.
// logout() - User logged out.

<?php
    // Write your solution here
?>
```