Theory Assignment Report

Only for course Teacher						
		Needs Improvement	Developing	Sufficient	Above Average	Total Mark
Allocate mark & Percentage		25%	50%	75%	100%	5
Clarity	1					
Content Quality	2					
Spelling & Grammar	1					
Organization and Formatting	1					
					Total obtained mark	
Comments						

Semester: Spring 25

Student Name: Nur Ahmed

Student ID: 222-35-1111

Batch: 38th **Section:** A

Course Code: SE321

Course Name: Software Engineering Web Application

Course Teacher Name: Mujahidul Islam

Designation: Lecturer

Submission Date: 09/04/2025

Contents

PHP OOP Concepts

What is Object-Oriented Programming?

The programming model Object-Oriented Programming (OOP) defines objects as encapsulated entities which merge attributes (also referred to as properties or fields) and procedures (also known as methods). Through OOP programmers attempt to achieve real-world programming entities which include inheritance along with hiding and polymorphism.

Importance of OOP in PHP

The original procedural PHP language acquired OOP abilities with version 4 while version 5 delivered major improvements to this functionality. PHP implementations of OOP lead developers to produce reusable code that enables better maintenance as well as the ability to construct sophisticated scalable web applications.

Classes and Objects

Class: A class is a blueprint for objects. It defines a type of object according to the methods and properties it has.

Object: An object is an instance of a class.

Coding Example:

main.php

```php
<?php
class Car {
    public $color;
    public $model;

    public function __construct($color, $model) {
        $this->color = $color;
        $this->model = $model;
    }

    public function message() {
        return "My car is a " . $this->color . " " . $this
            ->model . ".";
    }
}

$myCar = new Car("black", "Volvo");
echo $myCar->message();
?>
```

Output

```
My car is a black Volvo.

=== Code Execution Successful ===
```

Explanation:

This example defines a Car class with properties $color and $model, a constructor, and a method to return a string about the car.

Inheritance

Inheritance allows a class to inherit the properties and methods of another class.

Coding Example:

main.php

```php
<?php
class Vehicle {
    public $brand = "Generic";

    public function honk() {
        return "Beep beep!";
    }
}

class Truck extends Vehicle {
    public $type = "Heavy-Duty";
}

$myTruck = new Truck();
echo $myTruck->brand; // Outputs: Generic
echo $myTruck->honk(); // Outputs: Beep beep!
?>
```

Output

```
GenericBeep beep!

=== Code Execution Successful ===
```

Encapsulation

Encapsulation restricts access to some of an object's components.

Coding Example:

main.php

```php
<?php
class BankAccount {
    private $balance = 0;

    public function deposit($amount) {
        $this->balance += $amount;
    }

    public function getBalance() {
        return $this->balance;
    }
}

$account = new BankAccount();
$account->deposit(500);
echo $account->getBalance();
?>
```

Output

```
500

=== Code Execution Successful ===
```

Polymorphism

Polymorphism allows methods to do different things based on the object that is calling them.

Coding Example:

main.php

```php
<?php
class Animal {
    public function makeSound() {
        return "Some sound";
    }
}

class Dog extends Animal {
    public function makeSound() {
        return "Bark";
    }
}

class Cat extends Animal {
    public function makeSound() {
        return "Meow";
    }
}

$animals = [new Dog(), new Cat()];
foreach ($animals as $animal) {
    echo $animal->makeSound() . "<br>";
}
?>
```

Output

```
Bark<br>Meow<br>

=== Code Execution Successful ===
```

Abstraction

Abstraction means hiding the complex reality while exposing only the necessary parts.

Coding Example:

main.php

```php
<?php
abstract class Shape {
    abstract public function getArea();
}

class Square extends Shape {
    private $length;

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

    public function getArea() {
        return $this->length * $this->length;
    }
}

$square = new Square(4);
echo $square->getArea();
?>
```

Output

```
16

=== Code Execution Successful ===
```

Benefits of OOP in PHP

- Codes are reusable through the inheritance feature.
- Programs become simpler to scale because of better application control.
- The design of PHP code becomes easier to keep supporting and altering in operation.
- The more secure architecture emerges through the encapsulation method which prevents unauthorized data access.
- Code receives logical partition within separate modules called objects.

Conclusion

The PHP programming language implements Object-Oriented Programming to generate effective application system structures. Through combination of inheritance alongside encapsulation and abstraction and polymorphism developers can produce code which maintains straightforward design and also sustains growth and continues to be maintainable. By adopting OOP principles in PHP development applications will achieve higher quality alongside enhanced robustness.