Overview of Object-Oriented Programming (OOP)

Learning Objectives

- Define what Object-Oriented Programming (OOP) is
- Differentiate procedural and object-oriented approaches
- Identify the 4 main principles of OOP
- Recognize the benefits and purpose of OOP in software development

What is Object-Oriented Programming?

What is Object-Oriented Programming?

Object-Oriented Programming (OOP) is a programming paradigm based on the concept of **objects**, which represent real-world entities. These objects contain data in the form of **attributes** (also called properties) and behavior in the form of **methods** (functions).

Classes

A **class** is a blueprint for creating objects, which are specific instances of the class. It defines properties (attributes) and behaviors (methods) that the objects of the class will have.

OBJECT

Objects are instances of a class in object-oriented programming (OOP). They represent real-world entities or abstract concepts that contain both **data** (attributes) and **behavior** (methods).

ACCESS MODIFIERS

Access modifiers are keywords in object-oriented programming (OOP) that control the visibility and accessibility of class members (attributes and methods). They define whether other parts of the program can access or modify these members.

ACCESS MODIFIERS

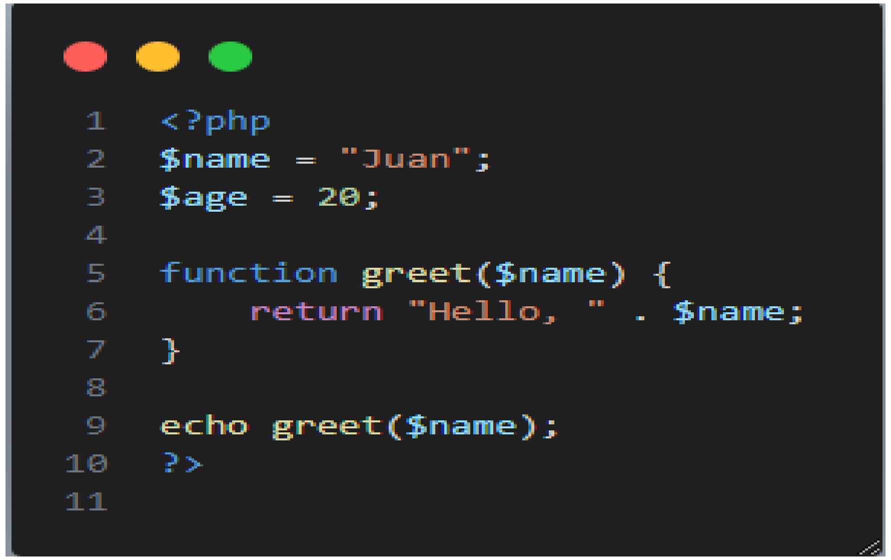
MODIFIERS	EXAMPLE	VISIBILITY
public	public $name;	Accessible from anywhere
private	private $age;	Accessible only within the class (name mangling)
protected	protected $salary	Accessible within the class and its subclasses

Differentiate procedural and object-oriented approaches

Procedural vs. Object-Oriented Programming

Procedural Programming

•Uses functions and steps in order
•Data is separate from logic
•Harder to reuse and maintain



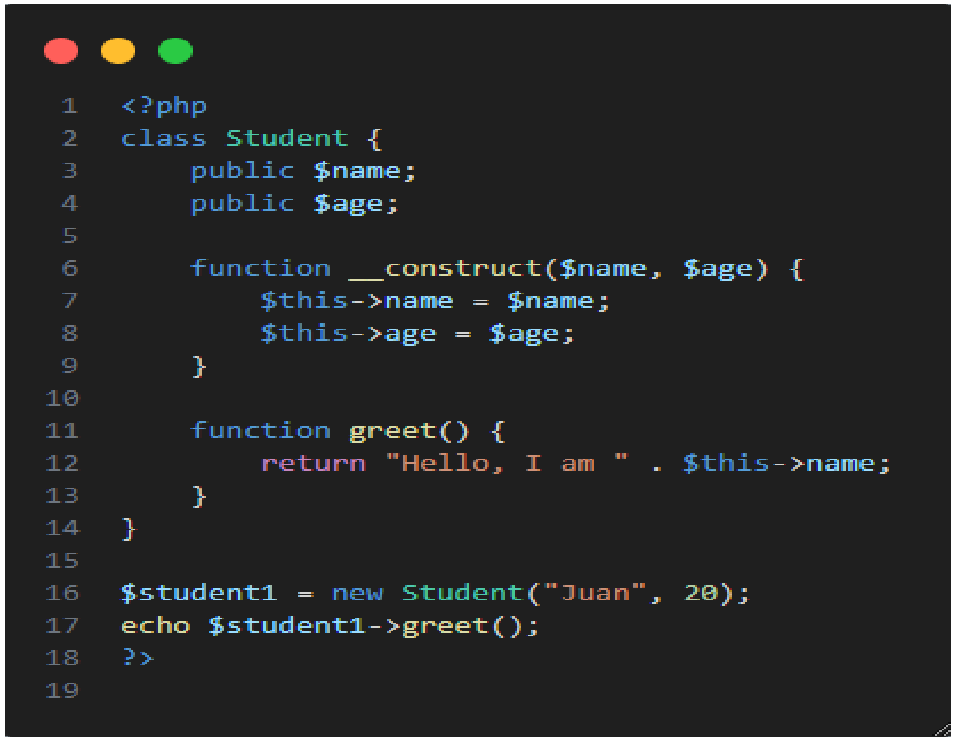
```php
<?php
$name = "Juan";
$age = 20;

function greet($name) {
    return "Hello, " . $name;
}

echo greet($name);
?>
```

Object-Oriented Programming

•Uses classes and objects
•Combines data and actions into reusable code blocks
•Easier to update and organize



```php
<?php
class Student {
    public $name;
    public $age;

    function __construct($name, $age) {
        $this->name = $name;
        $this->age = $age;
    }

    function greet() {
        return "Hello, I am " . $this->name;
    }
}

$student1 = new Student("Juan", 20);
echo $student1->greet();
?>
```

The 4 Pillars of OOP

Encapsulation

-Data and functions are kept inside the class, and data is protected from direct access.

Abstraction

-Providing simple interfaces to interact with complex systems without exposing how they work inside.

Inheritance

-A class can get the properties and behaviors of another class.

Polymorphism

-Different classes can use the same function name, but each one behaves differently.

Benefits of Object-Oriented Programming (OOP)

1. Reusable Code

- Once a class is created, it can be reused anywhere in your program or in future projects.
- You can make many objects from a single class without rewriting code.
- Example: A User class can be used for login, profile, and account management without rewriting functions.

2. Easier to Maintain and Debug

- Since OOP keeps code organized into classes, it's easier to find and fix problems.
- Each class handles its own part, so bugs are easier to locate and fix without affecting the whole system.
- Example: If your login system has an error, you only check the Authentication class, not the whole application.

3. Scalable and Extendable

- OOP makes it easy to add new features without changing existing code too much.
- You can extend classes or add new ones as the system grows.
- Example: Start with a User class, and later add Admin or Customer classes using inheritance.

4. Easier to Maintain and Debug

- Classes help group related data and actions together.
- Code becomes more readable, especially in big projects.
- Developers can understand the system faster because it's well-structured.

Thank you