**Report about Abstract class and Interface**

**Created by: William George William Botros**

Track: PHP-ITP Alex Q2, 2023/2024-Intake3

In object-oriented programming (OOP), both abstract classes and interfaces are mechanisms that allow to define contracts for classes to adhere to, promoting code reusability and abstraction. However, they serve different purposes and have some key differences:

**Abstract Class:**

**1- Definition:**

-An abstract class is a class that cannot be instantiated on its own and is meant to be subclassed by other classes.

-It can have both abstract (unimplemented) methods and concrete (implemented) methods.

**2-Abstract Methods:**

-Abstract classes can have abstract methods, which are methods without a body.

-Subclasses must provide concrete implementations for all abstract methods defined in the abstract class.

**3- Constructor:**

-Abstract classes can have constructors, and they are usually used to initialize common properties or perform common tasks.

**4- Inheritance:**

-A class can extend only one abstract class.

**5- Access Modifiers:**

Abstract classes can have access modifiers for methods and fields, allowing control over their visibility.

**Interface:**

**1- Definition:**

-An interface is a collection of abstract methods (methods without a body) that a class can choose to implement.

-It is a way to achieve multiple inheritance in languages that support interfaces.

**2- Methods:**

-All methods in an interface are public and abstract by default.

-A class implementing an interface must provide concrete implementations for all methods declared in that interface.

**3- Fields:**

-Interfaces can only have constants (public, static, and final fields) but not instance variables.

**4- Inheritance:**

-A class can implement multiple interfaces, allowing for multiple inheritance of method signatures.

**5- Constructor:**

-Interfaces do not have constructors.