**CHUKWU DANIEL NONSO**

**BHU/20/04/05/0010**

**CMP 325**

QUESTIONS & ANSWERS

1. With code examples explain polymorphism & its different types?

**ANSWER**

What is Polymorphism?

Polymorphism is a concept in object-oriented programming (OOP) that allows objects of different types to be treated as objects of a common type. This means that a single piece of code can operate on multiple types of objects in a consistent way, without the need for explicit type checking.

There are 2 main types of polymorphism in OOP:

**a. Static polymorphism** (also known as compile-time polymorphism) is achieved through method overloading, which is when a class has multiple methods with the same name but different parameter lists.

Code Example:

class Calculator {

int add(int a, int b) {

return a + b;

}

int add(int a, int b, int c) {

return a + b + c;

}

}

**b. Dynamic polymorphism** (also known as run-time polymorphism) is achieved through method overriding, which is when a subclass provides a specific implementation for a method that is already defined in its superclass.

Code Example:

Class Countries {

void welcome() {

System.out.println("Welcome to Country");

}

}

class Nigeria extends Countries {

void welcome() {

System.out.println("Welcome to Nigeria");

}

}

2. Explain in details with code snippets, abstraction & encapsulation?

**ANSWER**

**a. Abstraction** refers to the process of representing the essential features of an object or system, while hiding the details that are not relevant or necessary. In other words, abstraction allows you to simplify complex systems by focusing on the most important aspects of the problem at hand.

Code Example:

abstract class Human() {

abstract void sayHello();

}

class Circle extends Human() {

void sayHello() {

System.out.println("Hello World!");

}

}

With the above example we can see that abstraction allows us to focus on the properties of the class Human rather than its implementation

**b. Encapsulation** refers to the process of hiding the internal details of an object and making them inaccessible to other objects. By encapsulating the data and behavior of an object, you can protect the integrity of the objects state and ensure that the object is used correctly.

Code Example:

class Student {

private double gpa;

public double getGpa() { return gpa; } // getters

public void setGpa(double value) { gpa = value; } // setters

}

In the above example the attribute “gpa” is encapsulated within the Student class, and can only be accessed and updated using our setters and getters

3. Explain in detail the Java Technology?

**ANSWER**

Java is a popular general-purpose programming language that is widely used for developing a wide range of applications, including web applications, mobile apps, desktop applications, and enterprise systems. Developed by Sun Microsystems (now owned by Oracle) in the early 1990s, Java was designed to be easy to learn, write, and maintain, and to be platform-independent, meaning that code written in Java can run on any platform that supports the Java Virtual Machine (JVM).

One of the key features of Java is its use of the object-oriented programming (OOP) paradigm, which allows developers to create complex systems by breaking them down into smaller, reusable pieces (objects). Java also supports encapsulation, inheritance, and polymorphism, which are three key concepts of OOP.

Java supports multiple programming paradigms such as functional, procedural, and concurrent programming. Java also has a rich set of libraries and frameworks for various purposes including web development, database connectivity, and machine learning.