**Computer Networking- Assignment 3**

**Q:** *Surf the web and find the difference between encapsulation and abstraction (any viewpoint-java,c++,etc.)*

Encapsulation and abstraction are two fundamental concepts in object-oriented programming (OOP) that are supported in the Java programming language. Encapsulation refers to the practice of hiding the internal implementation details of an object from the outside world and providing access to it only through a well-defined interface. In Java, this is often achieved by using access modifiers such as public, private, and protected to control access to instance variables and methods. Encapsulation allows objects to maintain their state and behavior and prevents outside code from modifying or accessing them directly, which can lead to unintended consequences. Abstraction, on the other hand, refers to the practice of focusing on the essential features of an object while ignoring the irrelevant or non-essential details. This is often achieved by defining abstract classes or interfaces that define a set of methods that must be implemented by any class that inherits from them. Abstraction provides a way to create generalized classes that can be used in a variety of contexts and allows programmers to work at a higher level of abstraction without worrying about the implementation details. To summarize, encapsulation is about protecting data by controlling its access, while abstraction is about hiding implementation details by defining an abstract interface. Encapsulation is a fundamental concept in Java, whereas abstraction is a broader concept that includes encapsulation as one of its key components.

Example of encapsulation in Java:

public class BankAccount {

private double balance;

public BankAccount(double initialBalance) {

balance = initialBalance;

}

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) {

if (balance >= amount) {

balance -= amount;

} else {

System.out.println("Insufficient funds");

}

}

public double getBalance() {

return balance;

}

}

In this example, the BankAccount class encapsulates the balance variable and provides public methods to deposit, withdraw, and retrieve the balance. This allows outside code to interact with the BankAccount object without directly modifying or accessing its internal state.

Example of abstraction in Java:

public interface Shape {

public double getArea();

public double getPerimeter();

}

In this example, the Shape interface defines two abstract methods that must be implemented by any class that implements the interface. This allows programmers to work with shapes at a high level of abstraction, without worrying about the specific implementation of each shape. For example, a program could create an array of Shape objects and use the getArea and getPerimeter methods to calculate the properties of each shape, without knowing or caring about the specific type of shape being used.