**Aim:** Implement a program on multiple inheritance with interface.

**Objective:** Implement multiple inheritance in a program to perform addition, multiplication and transpose operations on a matrix. Create an interface to hold prototypes of these methods and create a class input to read input. Inherit a new class from this interface and class. In main class create object of this child class and invoke required methods.

## Theory:

* In Multiple inheritance, one class can have more than one superclass and inherit features from all parent classes. Java does not support [multiple inheritance](https://www.geeksforgeeks.org/java-and-multiple-inheritance/) with classes. In java, we can achieve multiple inheritance only through [Interfaces](http://quiz.geeksforgeeks.org/interfaces-in-java/).
* An interface contains variables and methods like a class but the methods in an interface are abstract by default unlike a class. If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.
* However, Java supports multiple interface inheritance where an interface extends more than one super interfaces.
* A class implements an interface, but one interface extends another interface.Multiple inheritance by interface occurs if a class implements multiple interfaces or also if an interface itself extends multiple interfaces.
* The following is the syntax used to extend multiple interfaces in Java:

access\_specifier interface subinterfaceName extends superinterface1, superinterface2, …… {

// Body  
}

**Code:**

class MultInherit{

public static void main(String args[])

{

Pig a=new Pig();

a.animalsound();

a.sleep();

}

}

interface Animal{

public void animalsound();

public void sleep();

}

class Pig implements Animal{

public void animalsound(){

System.out.println("The Pig says: wee-wee");

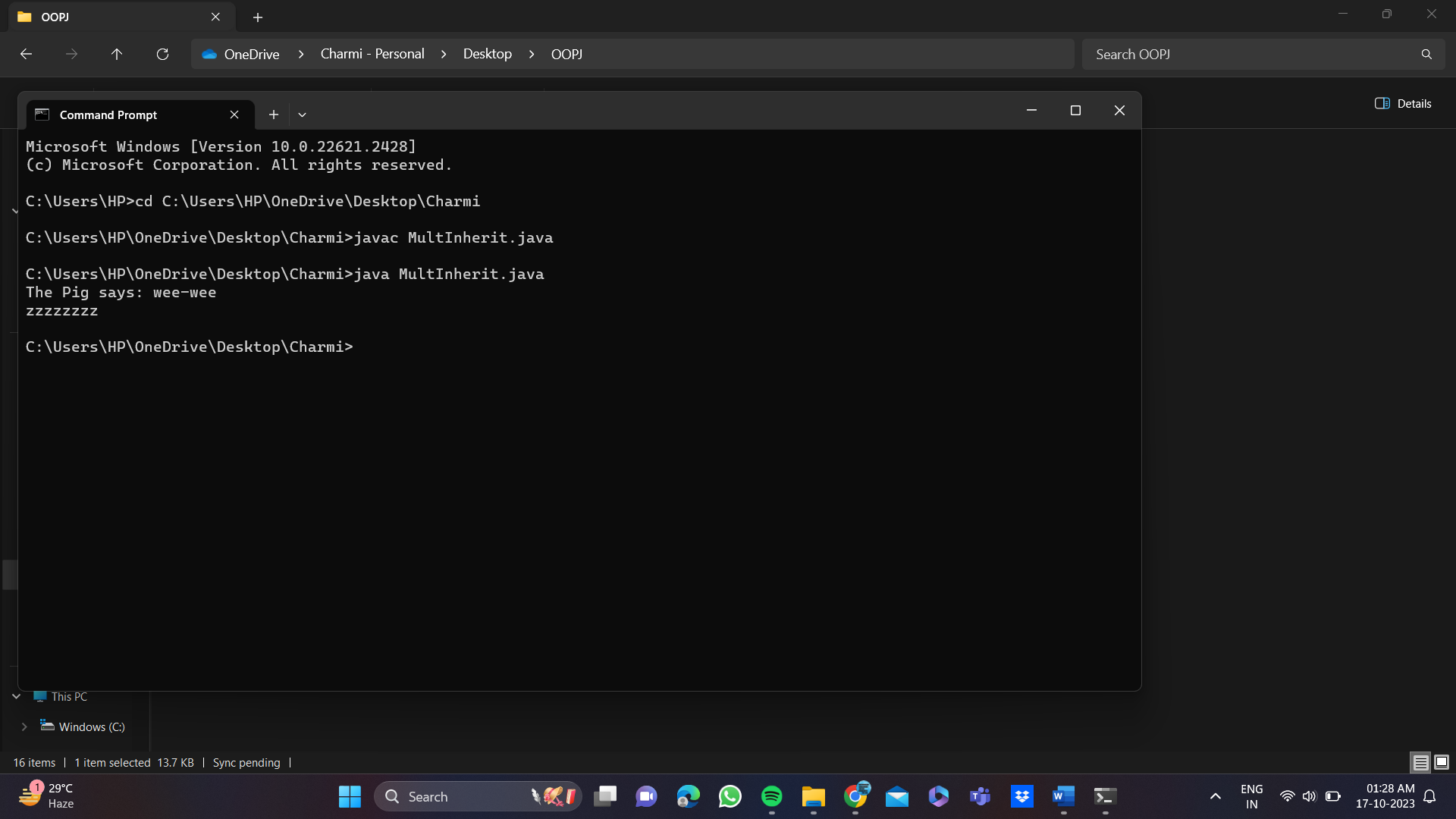
}

public void sleep(){

System.out.println("zzzzzzzz");

}

}



## Conclusion:

Comment on how interface are useful and implemented using java.

Interfaces in Java are a fundamental concept that allows you to define a contract specifying a set of methods that implementing classes must adhere to.

Abstraction: Interfaces allow you to define a contract or a set of methods without specifying the implementation. This promotes abstraction, enabling you to focus on what a class should do rather than how it should do it.