Experiment No. 8
Implement a program on multiple inheritance with interface.
Date of Performance:
Date of Submission:

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 with classes. In java, we can achieve multiple inheritance only through Interfaces.
- 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:

```
1. public class Demo {
public static void main(String args[]) {
 Animal a = new Animal();
```

```
a.eat();
a.travel();
}
}
interface AnimalEat {
void eat();
}
interface AnimalTravel {
void travel();
} class Animal implements AnimalEat, AnimalTravel {
public void eat() {
System.out.println("Animal is eating");
} public void travel() {
System.out.println("Animal is travelling");
}
}
```

OUTPUT:

Conclusion:

Comment on how interface are useful and implemented using java.

- Interfaces in Java provide a blueprint for classes, specifying a set of methods that implementing classes must define.
- They ensure code consistency and reusability by enforcing a common contract for implementing classes.

- Interfaces enable multiple classes to share a common set of methods without requiring a shared parent class, promoting flexibility in code design.

Implementing Interfaces in Java:

- To implement an interface, a class in Java uses the `implements` keyword followed by the interface name.
- The implementing class must provide concrete definitions (method bodies) for all the methods declared in the interface.
- Once implemented, objects of the class can be treated as instances of the interface, allowing for polymorphism and standardized method invocation.