NAME : SOHAM SAHA

SEC : CS2A

ROLL : 61

ENROLLMENT : 12019009001389

SUB : OOP WITH JAVA LAB

1. **Write a Java program to implement the concept of inheritance.**

**Code:**

class Employee {

float salary = 40000;

}

class Programmer extends Employee {

int bonus = 10000;

public static void main(String args[]) {

Programmer p = new Programmer();

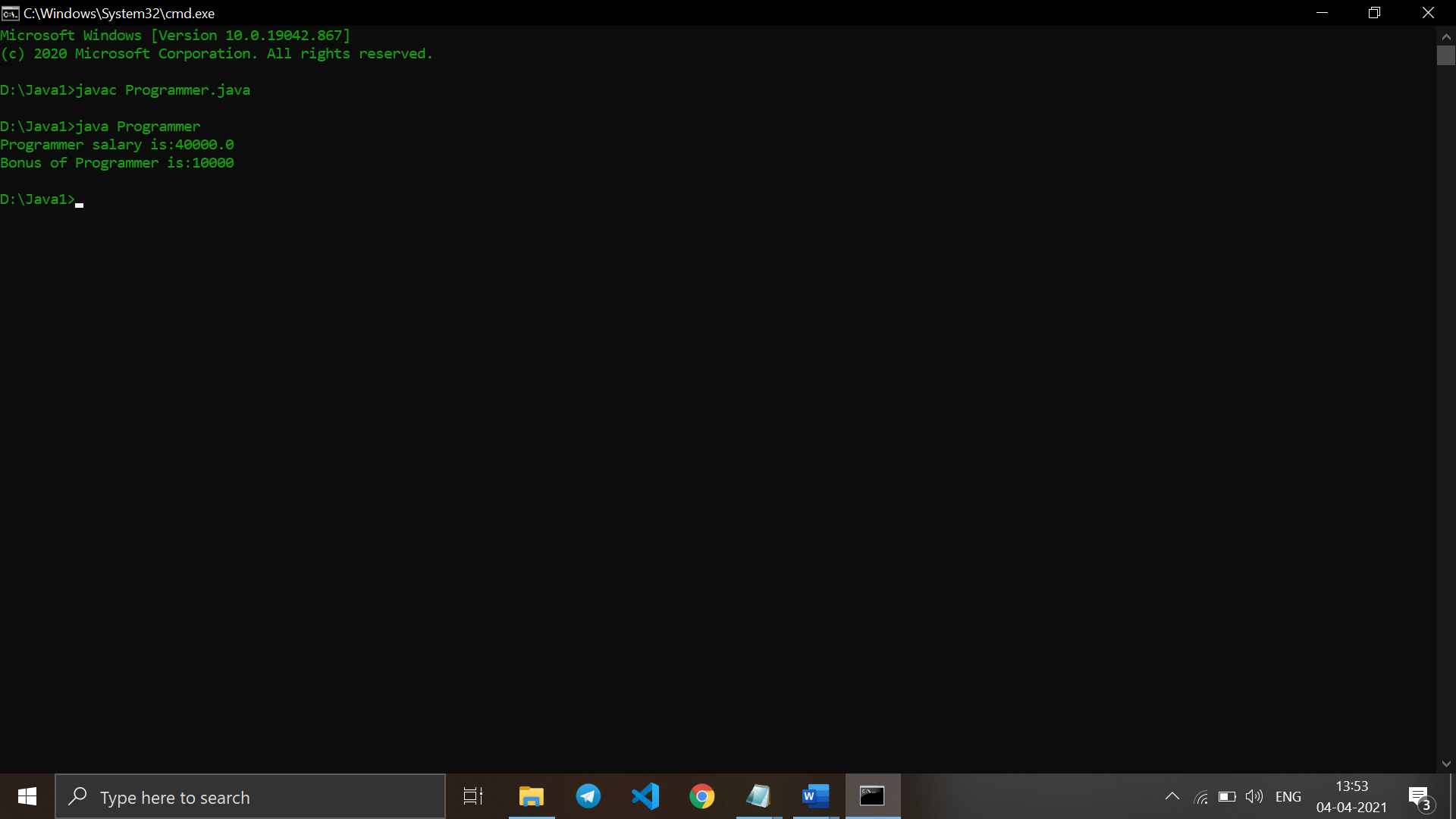
System.out.println("Programmer salary is:" + p.salary);

System.out.println("Bonus of Programmer is:" + p.bonus);

}

}

**Output:**



1. **Write a Java program to show method overloading.**

**Code:**

public class Sum {

public int sum(int x, int y) {

return (x + y);

}

public int sum(int x, int y, int z) {

return (x + y + z);

}

public double sum(double x, double y) {

return (x + y);

}

public static void main(String args[]) {

Sum s = new Sum();

System.out.println(s.sum(10, 20));

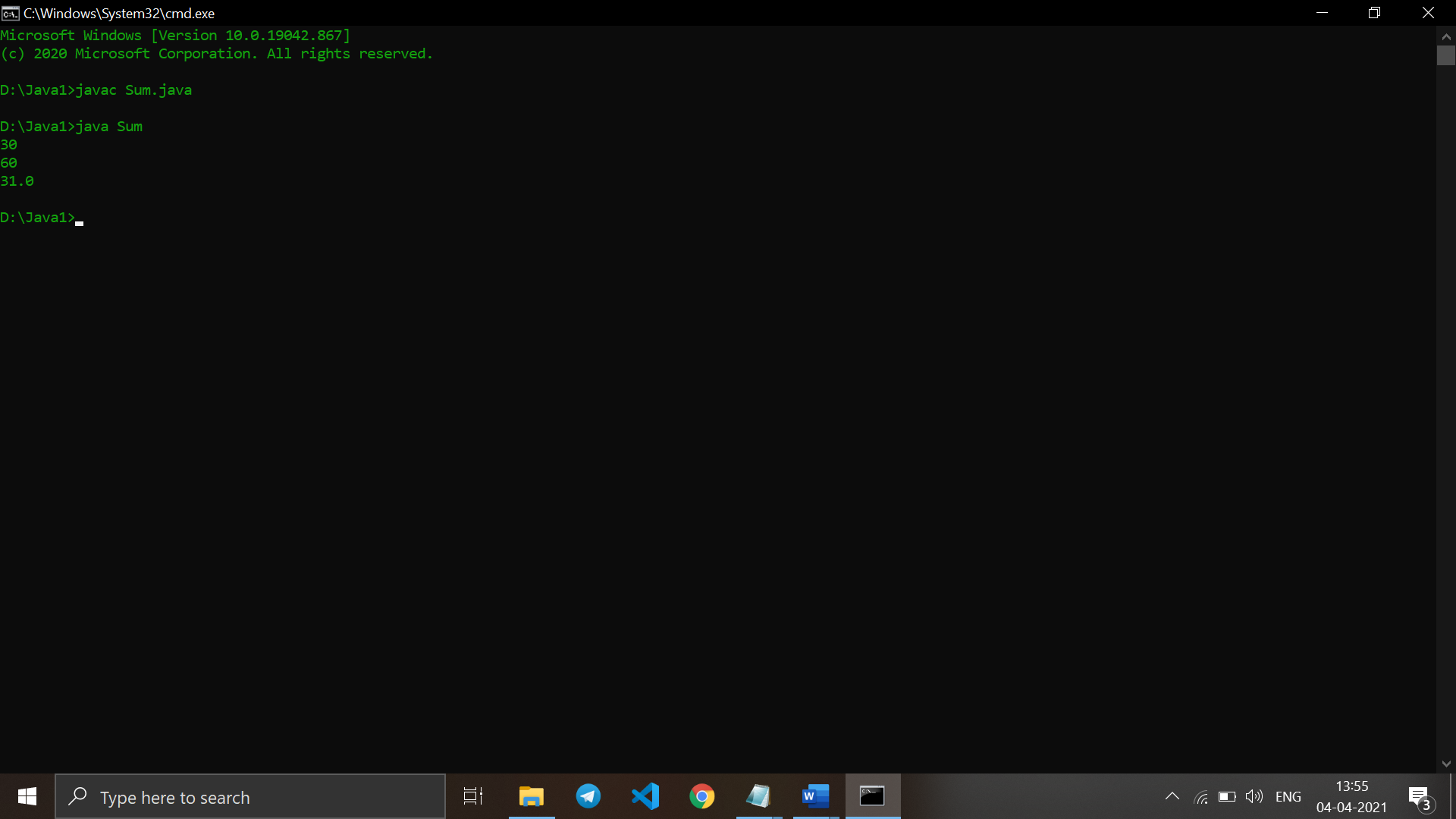
System.out.println(s.sum(10, 20, 30));

System.out.println(s.sum(10.5, 20.5));

}

}

**Output:**



1. **Write a Java program to show method overriding.**

**Code:**

class Parent {

void show() {

System.out.println("Parent's show()");

}

}

class Child extends Parent {

@Override

void show() {

System.out.println("Child's show()");

}

}

class Main {

public static void main(String[] args) {

Parent obj1 = new Parent();

obj1.show();

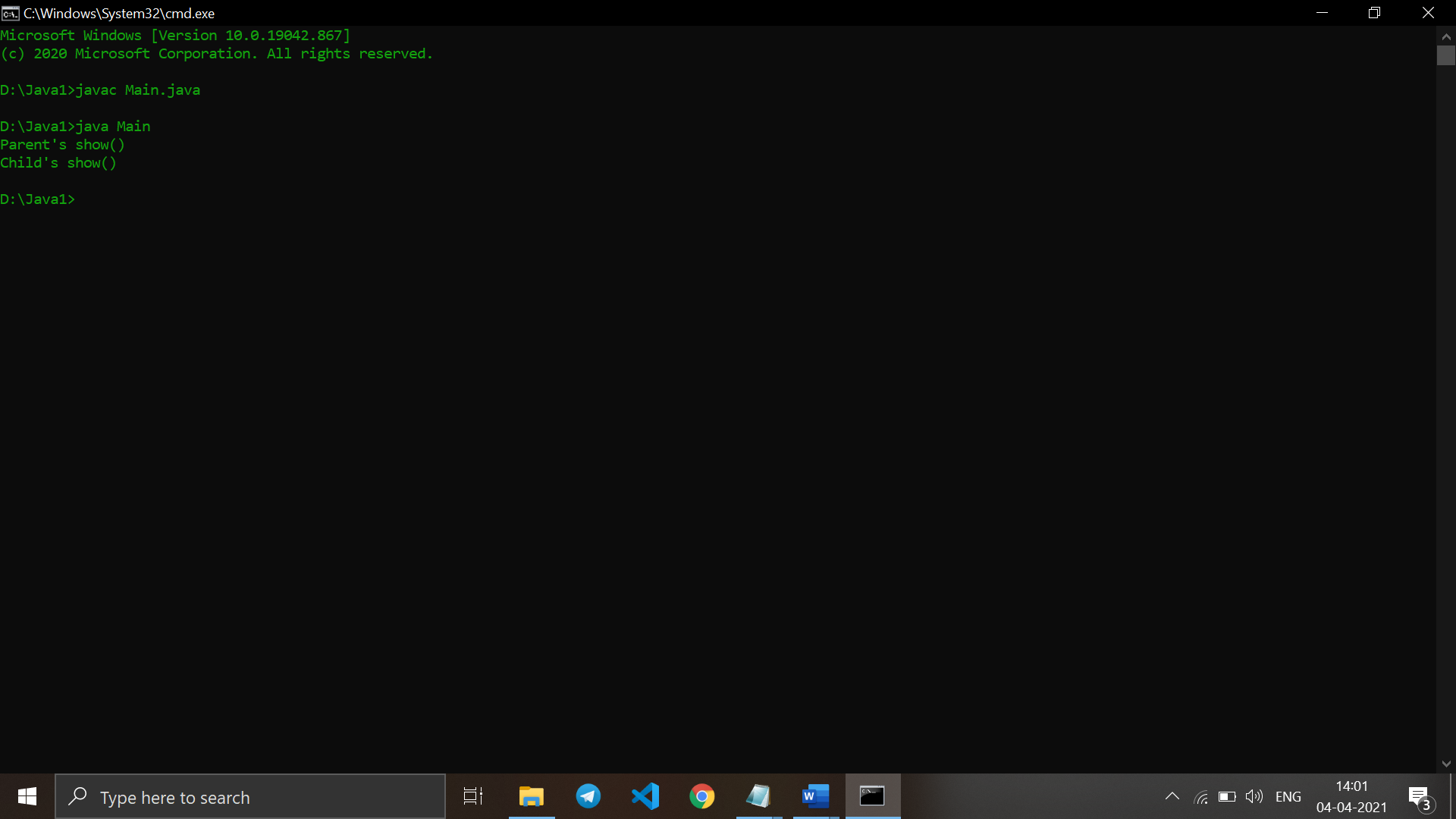
Parent obj2 = new Child();

obj2.show();

}

}

**Output:**



1. **Create a general class ThreeDObject and derive the classes Box, Cube, Cylinder and Cone from it. The class ThreeDObject has methods wholeSurfaceArea ( ) and volume ( ). Override these two methods in each of the derived classes to calculate the volume and whole surface area of each type of three-dimensional objects. The dimensions of the objects are to be taken from the users and passed through the respective constructors of each derived class. Write a main method to test these classes.**

**Code:**

class ThreeDObject

{

double wholeSurfaceArea (){

return 0;

}

double volume(){

return 0;

}

double wholeSurfaceArea (int a,int b,int c){

return 0;

}

double volume(int a,int b,int c){

return 0;

}

double wholeSurfaceArea (int a,int b){

return 0;

}

double volume(int a,int b){

return 0;

}

double wholeSurfaceArea (int a){

return 0;

}

double volume(int a){

return 0;

}

}

class Box extends ThreeDObject

{

double wholeSurfaceArea (int l,int b,int h ){

double sa = 2\*(l+b)\*h;

return sa;

}

double volume(int l,int b,int h){

return l\*b\*h;

}

}

class Cube extends ThreeDObject

{

double wholeSurfaceArea (int a){

double sa = 4\*a\*a;

return sa;

}

double volume(int a){

return a\*a\*a;

}

}

class Cylinder extends ThreeDObject

{

double wholeSurfaceArea (int r,int h ){

double sa = 2\*3.14\*r\*(r+h);

return sa;

}

double volume(int r,int h){

return 3.14\*r\*r\*h;

}

}

class Cone extends ThreeDObject

{

double wholeSurfaceArea (int r,int h ){

double sa = 3.14\*r\*(r+Math.sqrt((r\*r)+(h\*h)));

return sa;

}

double volume(int r,int h){

double ar=3.14\*r\*r\*h/3;

return ar;}}

public class MainClass

{

public static void main(String args[])

{

ThreeDObject box = new Box();

ThreeDObject cube = new Cube();

ThreeDObject cylinder = new Cylinder();

ThreeDObject cone = new Cone();

System.out.println("The Whole Surface Area of Box is: "+box.wholeSurfaceArea(6,8,4));

System.out.println("The Whole Surface Area of Cube is: "+cube.wholeSurfaceArea(7));

System.out.println("The Whole Surface Area of Cone is: "+cone.wholeSurfaceArea(5,9));

System.out.println("The Whole Surface Area of Cylinder is: "+cylinder.wholeSurfaceArea(4,8));

System.out.println("The volume of Box is: "+box.volume(6,8,4));

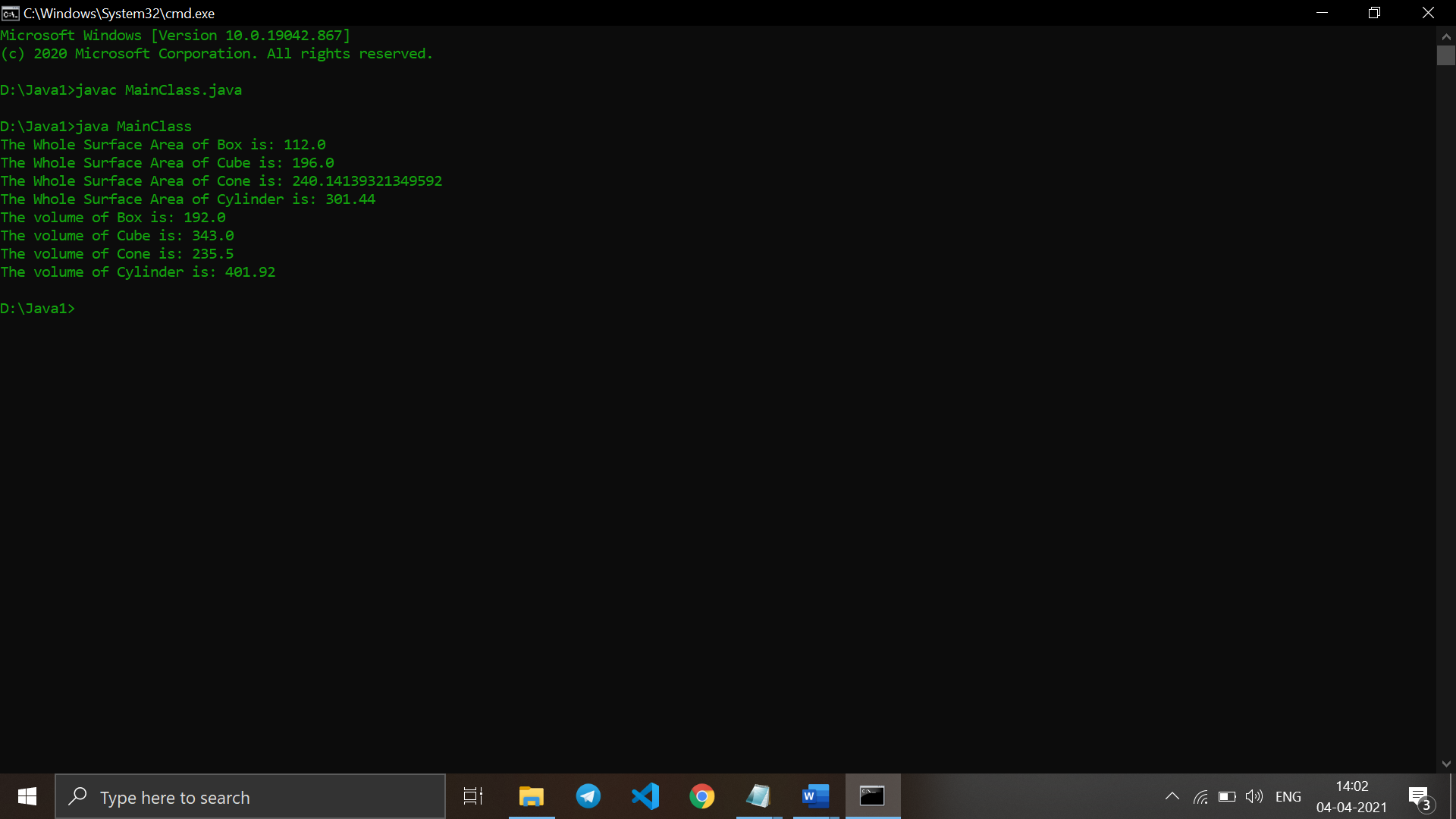
System.out.println("The volume of Cube is: "+cube.volume(7));

System.out.println("The volume of Cone is: "+cone.volume(5,9));

System.out.println("The volume of Cylinder is: "+cylinder.volume(4,8));

}}

**Output:**



1. **Write a program to create a class named Vehicle having protected instance variables regnNumber, speed, color, ownerName and a method showData ( ) to show “This is a vehicle class”. Inherit the Vehicle class into subclasses named Bus and Car having individual private instance variables routeNumber in Bus and manufacturerName in Car and both of them having showData ( ) method showing all details of Bus and Car respectively with content of the super class’s showData ( ) method.**

**Code:**

class Vehicle {

protected int regnNumber, speed;

protected String color, ownerName;

Vehicle(int regnNo, int speed, String color, String ownerName) {

this.regnNumber = regnNo;

this.speed = speed;

this.color = color;

this.ownerName = ownerName;

}

protected void showData() {

System.out.println("This is a vehicle class");

System.out.println("Regn. Number = " + this.regnNumber);

System.out.println("Speed = " + this.speed);

System.out.println("Color = " + this.color);

System.out.println("Owner Name = " + this.ownerName);

}

}

class Bus extends Vehicle {

private int routeNumber;

Bus(int routeNumber, int regnNo, int speed, String color, String ownerName) {

super(regnNo, speed, color, ownerName);

this.routeNumber = routeNumber;

}

public void showData() {

super.showData();

System.out.println("The route number = " + this.routeNumber);

}

}

public class Car extends Vehicle {

private String manufacName;

Car(String manufacName, int regnNo, int speed, String color, String ownerName) {

super(regnNo, speed, color, ownerName);

this.manufacName = manufacName;

}

public void showData() {

super.showData();

System.out.println("The Manufacturer Name = " + this.manufacName);

}

public static void main(String args[]) {

System.out.println("For Bus");

System.out.println("----------------------------------------------");

Bus ob1 = new Bus(222, 20213, 45, "blue", "James Bond");

ob1.showData();

System.out.println("For Car");

System.out.println("----------------------------------------------");

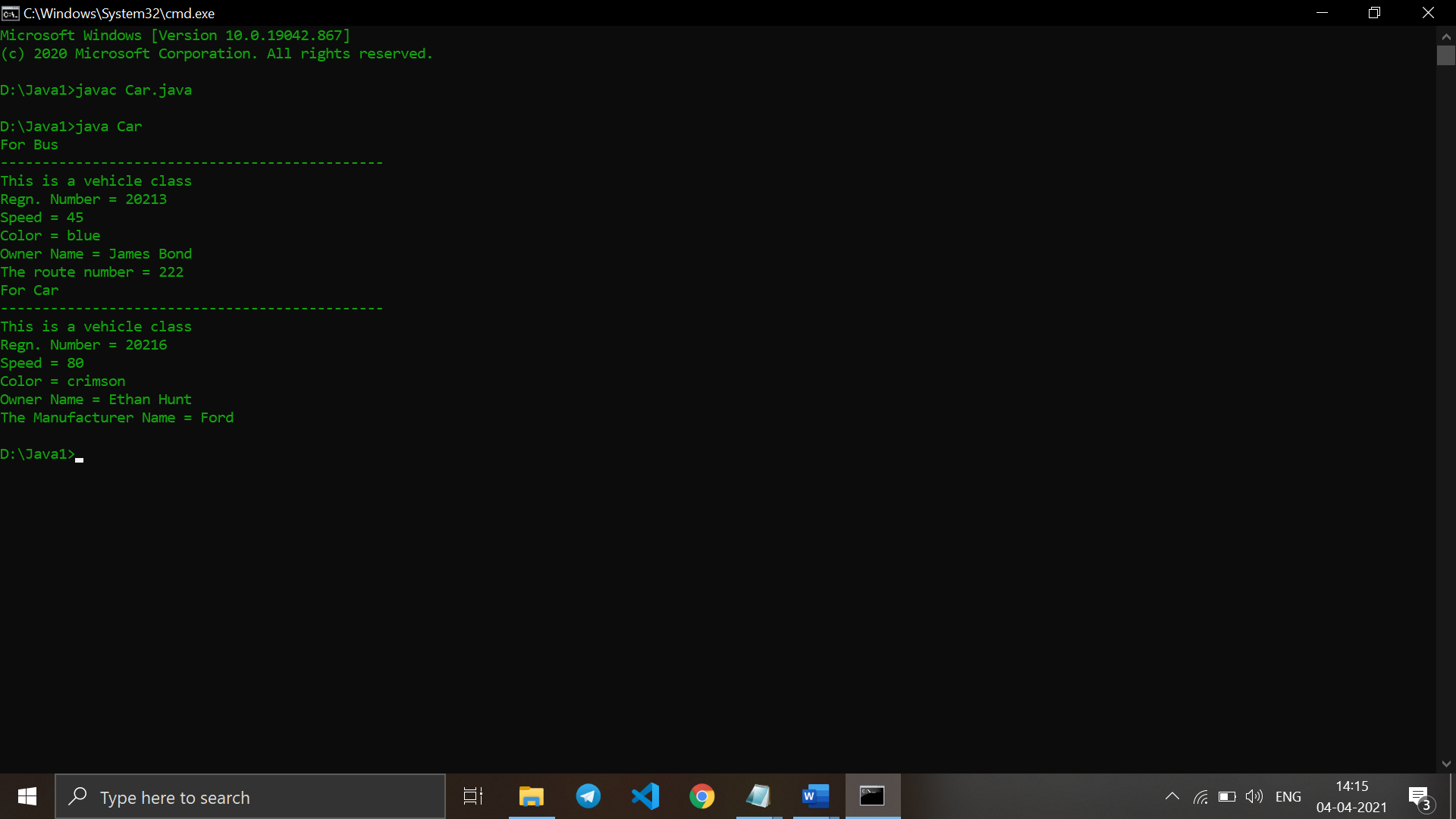
Car ob2 = new Car("Ford", 20216, 80, "crimson", "Ethan Hunt");

ob2.showData();

}

}

**Output:**



1. **An educational institution maintains a database of its employees. The database is divided into a number of classes whose hierarchical relationships are shown below. Write all the classes and define the methods to create the database and retrieve individual information as and when needed. Write a driver program to test the classes.**

***Staff* (code, name) *Teacher* (subject, publication) is a Staff**

***Officer* (grade) is a Staff *Typist* (speed) is a Staff**

***RegularTypist* (remuneration) is a Typist *CasualTypist* (daily wages) is a Typist.**

**class Staff {**

**int code;**

**String name;**

**Staff(int code, String name) {**

**this.code = code;**

**this.name = name;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + this.code);**

**System.out.println("Name = " + this.name);**

**}**

**}**

**class Teacher extends Staff {**

**String subject, publication;**

**Teacher(int code, String name, String sub, String publ) {**

**super(code, name);**

**this.subject = sub;**

**this.publication = publ;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + super.code);**

**System.out.println("Name = " + super.name);**

**System.out.println("Subject = " + this.subject);**

**System.out.println("Publication = " + this.publication);**

**}**

**}**

**class Officer extends Staff {**

**String grade;**

**Officer(int code, String name, String grade) {**

**super(code, name);**

**this.grade = grade;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + super.code);**

**System.out.println("Name = " + super.name);**

**System.out.println("Grade = " + this.grade);**

**}**

**}**

**class Typist extends Staff {**

**int speed;**

**Typist(int code, String name, int speed) {**

**super(code, name);**

**this.speed = speed;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + super.code);**

**System.out.println("Name = " + super.name);**

**System.out.println("Speed = " + this.speed);**

**}**

**}**

**class RegularTypist extends Typist {**

**int remuneration;**

**RegularTypist(int code, String name, int speed, int remuneration) {**

**super(code, name, speed);**

**this.remuneration = remuneration;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + super.code);**

**System.out.println("Name = " + super.name);**

**System.out.println("Speed = " + super.speed + " wpm");**

**System.out.println("Remuneration = " + this.remuneration + "$");**

**}**

**}**

**class CasualTypist extends Typist {**

**int dailyWages;**

**CasualTypist(int code, String name, int speed, int dailyWages) {**

**super(code, name, speed);**

**this.dailyWages = dailyWages;**

**}**

**public void getDetails() {**

**System.out.println("Code = " + super.code);**

**System.out.println("Name = " + super.name);**

**System.out.println("Speed = " + super.speed + " wpm");**

**System.out.println("Daily Wages = " + this.dailyWages + "$");**

**}**

**}**

**public class DriverClass {**

**public static void main(String[] args) {**

**Staff ob1 = new Teacher(100, "John Doe", "Maths", "AAC");**

**ob1.getDetails();**

**Staff ob2 = new Officer(101, "Jane Doe", "A");**

**ob2.getDetails();**

**Staff ob3 = new Typist(102, "James Doe", 250);**

**ob3.getDetails();**

**Staff ob4 = new RegularTypist(103, "James Depp", 250, 500);**

**ob4.getDetails();**

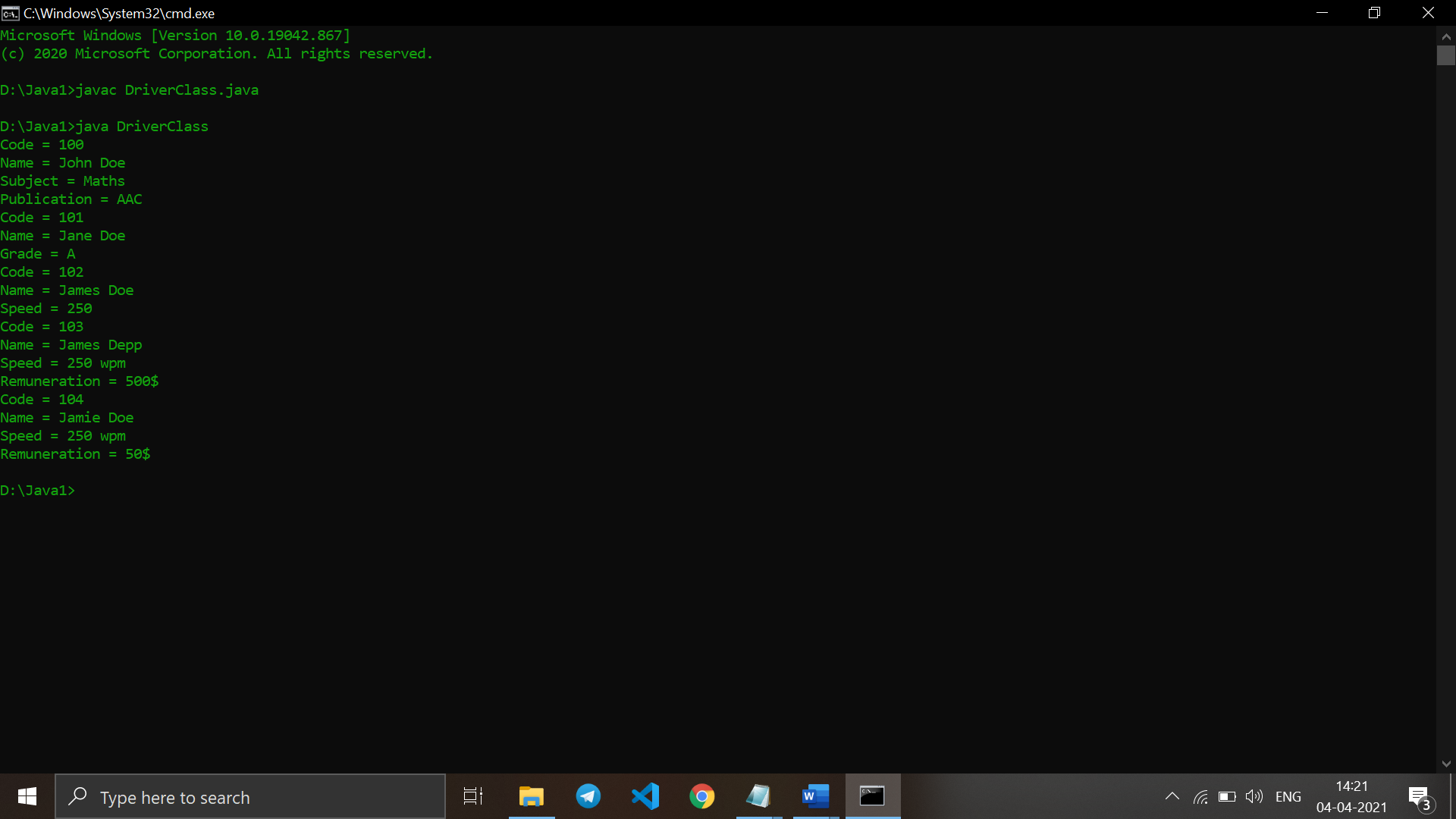
**Staff ob5 = new RegularTypist(104, "Jamie Doe", 250, 50);**

**ob5.getDetails();**

**}**

**}**

**Output:**



1. **Create a base class Building that stores the number of floors of a building, number of rooms and it’s total footage. Create a derived class House that inherits Building and also stores the number of bedrooms and bathrooms. Demonstrate the working of the classes.**

**class Building {**

**int floors, rooms, footage;**

**Building(int floors, int rooms, int footage) {**

**this.floors = floors;**

**this.rooms = rooms;**

**this.footage = footage;**

**}**

**void show() {**

**System.out.println("Number of Floors: " + this.rooms);**

**System.out.println("Number of Rooms: " + this.floors);**

**System.out.println("Total Footage: " + this.footage);**

**}**

**}**

**public class House extends Building {**

**int bedrooms, bathrooms;**

**House(int floors, int rooms, int footage, int bedrooms, int bathrooms) {**

**super(floors, rooms, footage);**

**this.bedrooms = bedrooms;**

**this.bathrooms = bathrooms;**

**}**

**void show() {**

**super.show();**

**System.out.println("Number of Bedrooms: " + this.bedrooms);**

**System.out.println("Number of Bathrooms: " + this.bathrooms);**

**}**

**public static void main(String[] args) {**

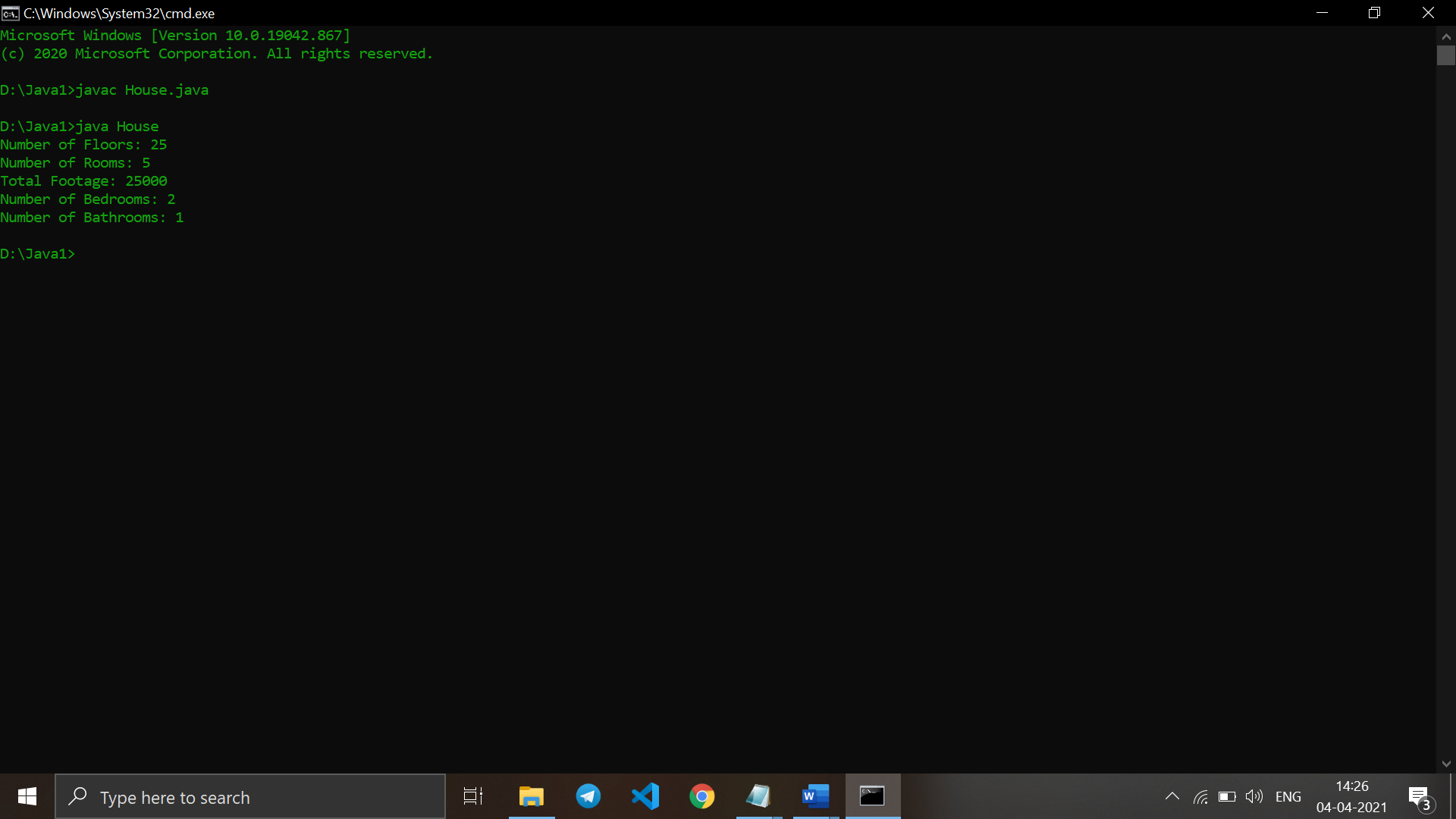
**House ob = new House(5, 25, 25000, 2, 1);**

**ob.show();**

**}**

**}**

**Output:**



1. **In the earlier program, create a second derived class Office that inherits Building and stores the number of telephones and tables. Now demonstrate the working of all three classes.**

**Code:  
class Building {**

**int floors, rooms, footage;**

**Building(int floors, int rooms, int footage) {**

**this.floors = floors;**

**this.rooms = rooms;**

**this.footage = footage;**

**}**

**void show() {**

**System.out.println("Number of Floors: " + this.rooms);**

**System.out.println("Number of Rooms: " + this.floors);**

**System.out.println("Total Footage: " + this.footage);**

**}**

**}**

**class House extends Building {**

**int bedrooms, bathrooms;**

**House(int floors, int rooms, int footage, int bedrooms, int bathrooms) {**

**super(floors, rooms, footage);**

**this.bedrooms = bedrooms;**

**this.bathrooms = bathrooms;**

**}**

**void show() {**

**super.show();**

**System.out.println("Number of Bedrooms: " + this.bedrooms);**

**System.out.println("Number of Bathrooms: " + this.bathrooms);**

**}**

**}**

**public class Office extends Building {**

**int telephones, tables;**

**Office(int floors, int rooms, int footage, int telephones, int tables) {**

**super(floors, rooms, footage);**

**this.telephones = telephones;**

**this.tables = tables;**

**}**

**void show() {**

**super.show();**

**System.out.println("Number of Telephones: " + this.telephones);**

**System.out.println("Number of Tables: " + this.tables);**

**}**

**public static void main(String[] args) {**

**Building ob = new House(5, 25, 25000, 2, 1);**

**Building ob1 = new Office(5, 25, 25000, 50, 50);**

**System.out.println("House Details:");**

**ob.show();**

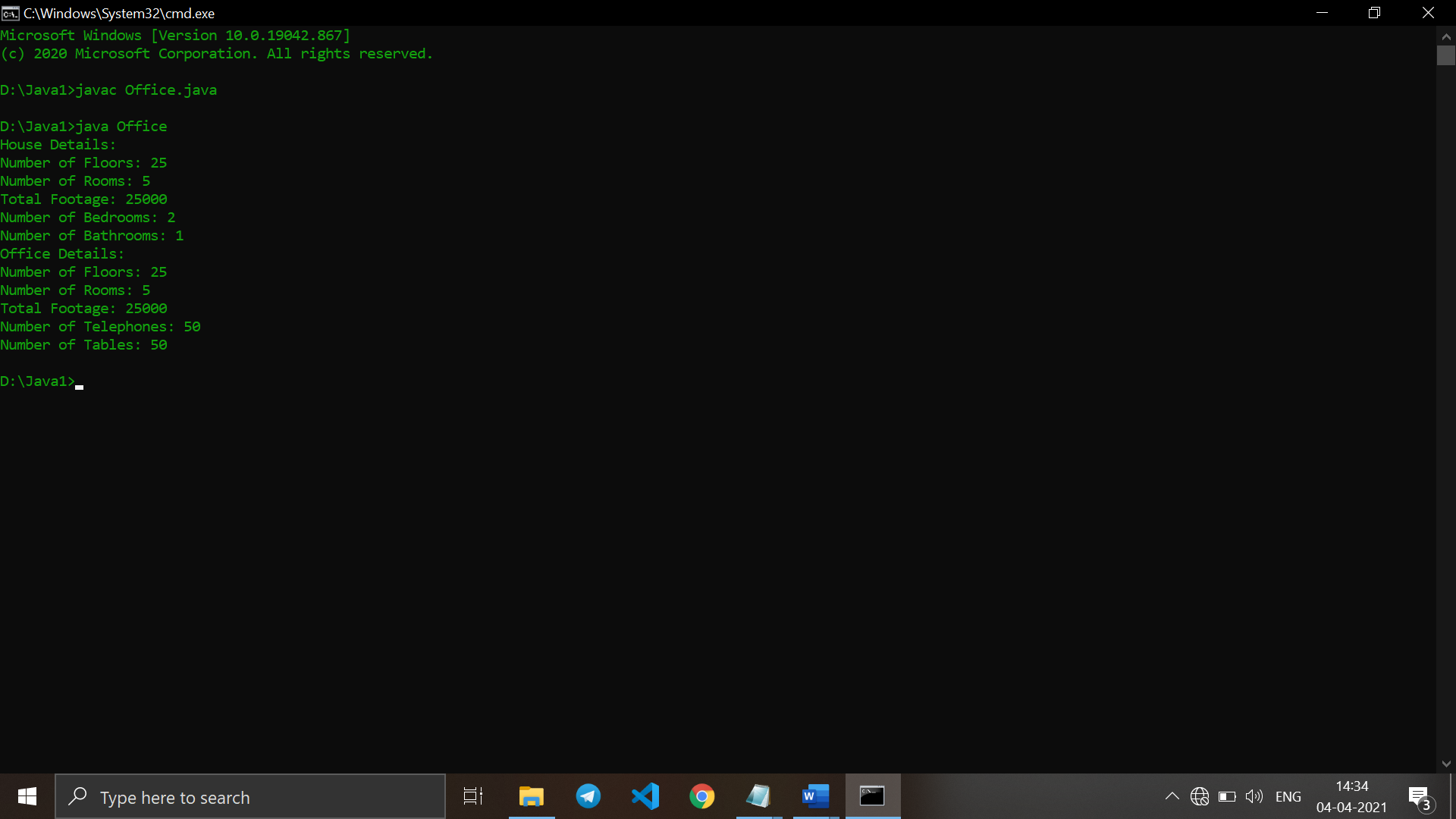
**System.out.println("Office Details:");**

**ob1.show();**

**}**

**}**

**Output:**



1. **Write a Java program which creates a base class Num and contains an integer number along with a method shownum() which displays the number. Now create a derived class HexNum which inherits Num and overrides shownum() which displays the hexadecimal value of the number. Demonstrate the working of the classes.**

**Code:**

**import java.util.\*;**

**class Num {**

**int num;**

**Num(int num) {**

**this.num = num;**

**}**

**public void showNum() {**

**System.out.println("Number = " + this.num);**

**}**

**}**

**public class HexNum extends Num {**

**HexNum(int num) {**

**super(num);**

**}**

**public void showNum() {**

**System.out.println("Number = " + super.num);**

**System.out.println("Hex Value = " + Integer.toHexString(super.num));**

**}**

**public static void main(String args[]) {**

**Scanner sc = new Scanner(System.in);**

**int n;**

**System.out.println("Enter the number ");**

**n = sc.nextInt();**

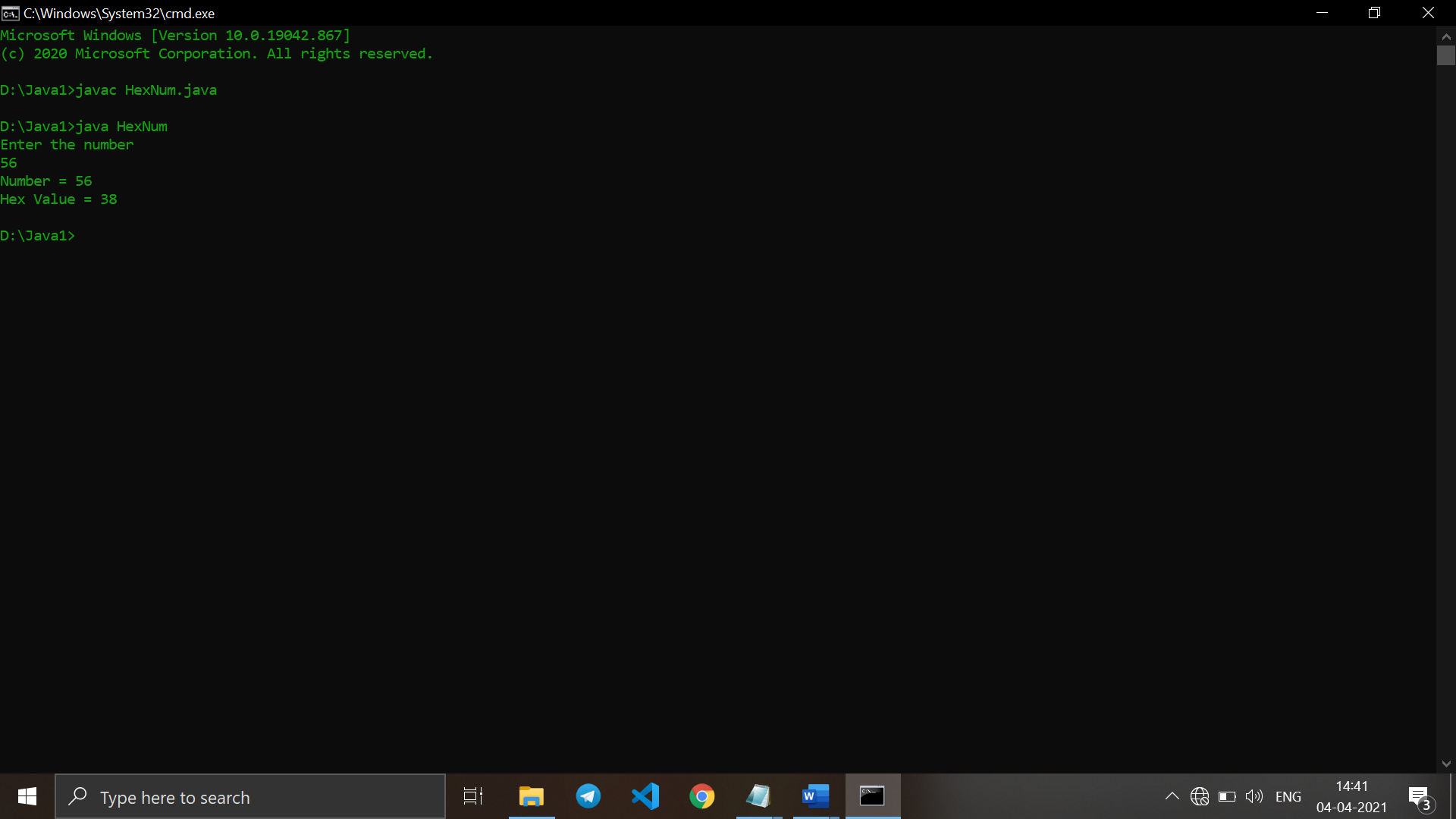
**HexNum ob = new HexNum(n);**

**ob.showNum();**

**sc.close();**

**}}**

**Output:**



1. **Write a Java program which creates a base class Num and contains an integer number along with a method shownum() which displays the number. Now create a derived class OctNum which inherits Num and overrides shownum() which displays the octal value of the number. Demonstrate the working of the classes.**

**Code:**

**import java.util.\*;**

**class Num {**

**int num;**

**Num(int num) {**

**this.num = num;**

**}**

**public void showNum() {**

**System.out.println("Number = " + this.num);**

**}**

**}**

**public class OctNum extends Num {**

**OctNum(int num) {**

**super(num);**

**}**

**public void showNum() {**

**System.out.println("Number = " + super.num);**

**System.out.println("Octal Value = " + Integer.toOctalString(super.num));**

**}**

**public static void main(String args[]) {**

**Scanner sc = new Scanner(System.in);**

**int n;**

**System.out.println("Enter the number ");**

**n = sc.nextInt();**

**OctNum ob = new OctNum(n);**

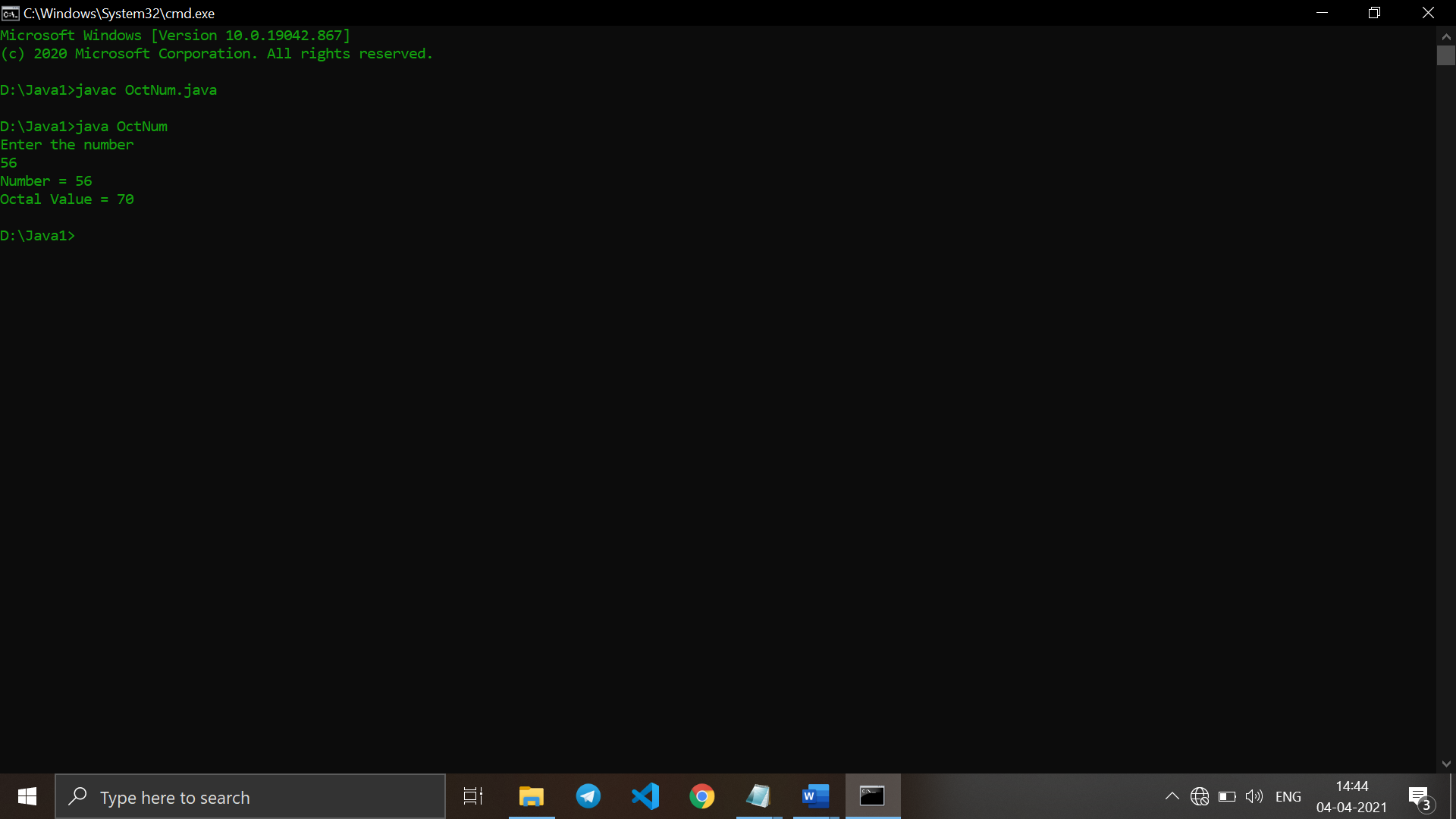
**ob.showNum();**

**sc.close();**

**}**

**}**

**Output:**



1. **Combine Question number 10 and 11 and have all the three classes together. Now describe the working of all classes.**

**Code:**

**import java.util.\*;**

**class Num {**

**int num;**

**Num(int num) {**

**this.num = num;**

**}**

**public void showNum() {**

**System.out.println("Number = " + this.num);**

**}**

**}**

**class HexNum extends Num {**

**HexNum(int num) {**

**super(num);**

**}**

**public void showNum() {**

**System.out.println("Number = " + super.num);**

**System.out.println("Hex Value = " + Integer.toHexString(super.num));**

**}**

**}**

**public class OctNum extends Num {**

**OctNum(int num) {**

**super(num);**

**}**

**public void showNum() {**

**System.out.println("Number = " + super.num);**

**System.out.println("Octal Value = " + Integer.toOctalString(super.num));**

**}**

**public static void main(String args[]) {**

**Scanner sc = new Scanner(System.in);**

**int n;**

**System.out.println("Enter the number ");**

**n = sc.nextInt();**

**HexNum ob1 = new HexNum(n);**

**ob1.showNum();**

**OctNum ob = new OctNum(n);**

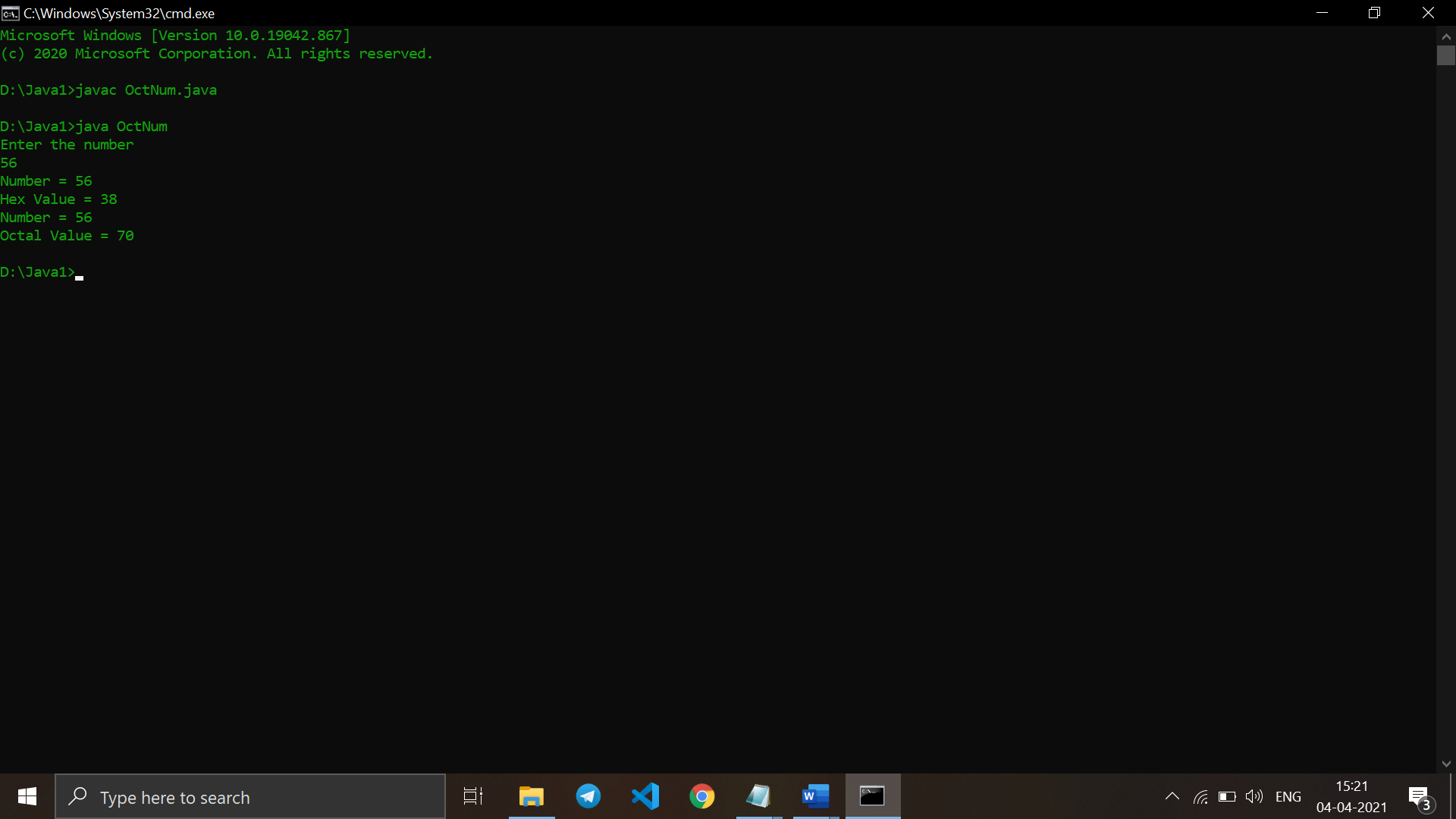
**ob.showNum();**

**sc.close();**

**}**

**}**

**Output:**



1. **Create a base class Distance which stores the distance between two locations in miles and a method travelTime(). The method prints the time taken to cover the distance when the speed is 60 miles per hour. Now in a derived class DistanceMKS, override travelTime() so that it prints the time assuming the distance is in kilometers and the speed is 100 km per second. Demonstrate the working of the classes.**

Code:

class Distance {

double distanceinmiles, speed;

Distance() {

distanceinmiles = 0.0;

speed = 60.0;

}

Distance(double distanceinmiles) {

this.distanceinmiles = distanceinmiles;

speed = 60.0;

}

double travelTime() {

return (distanceinmiles / speed);

}

}

public class DistanceMKS extends Distance {

double speed;

DistanceMKS(double distanceinmiles) {

super(distanceinmiles);

speed = 100.0;

}

double travelTime() {

return (distanceinmiles / speed);

}

public static void main(String[] args) {

Distance ob = new DistanceMKS(150);

System.out.println("Time taken to cover 150 miles at 100 miles/hour: " + ob.travelTime() + " hours");

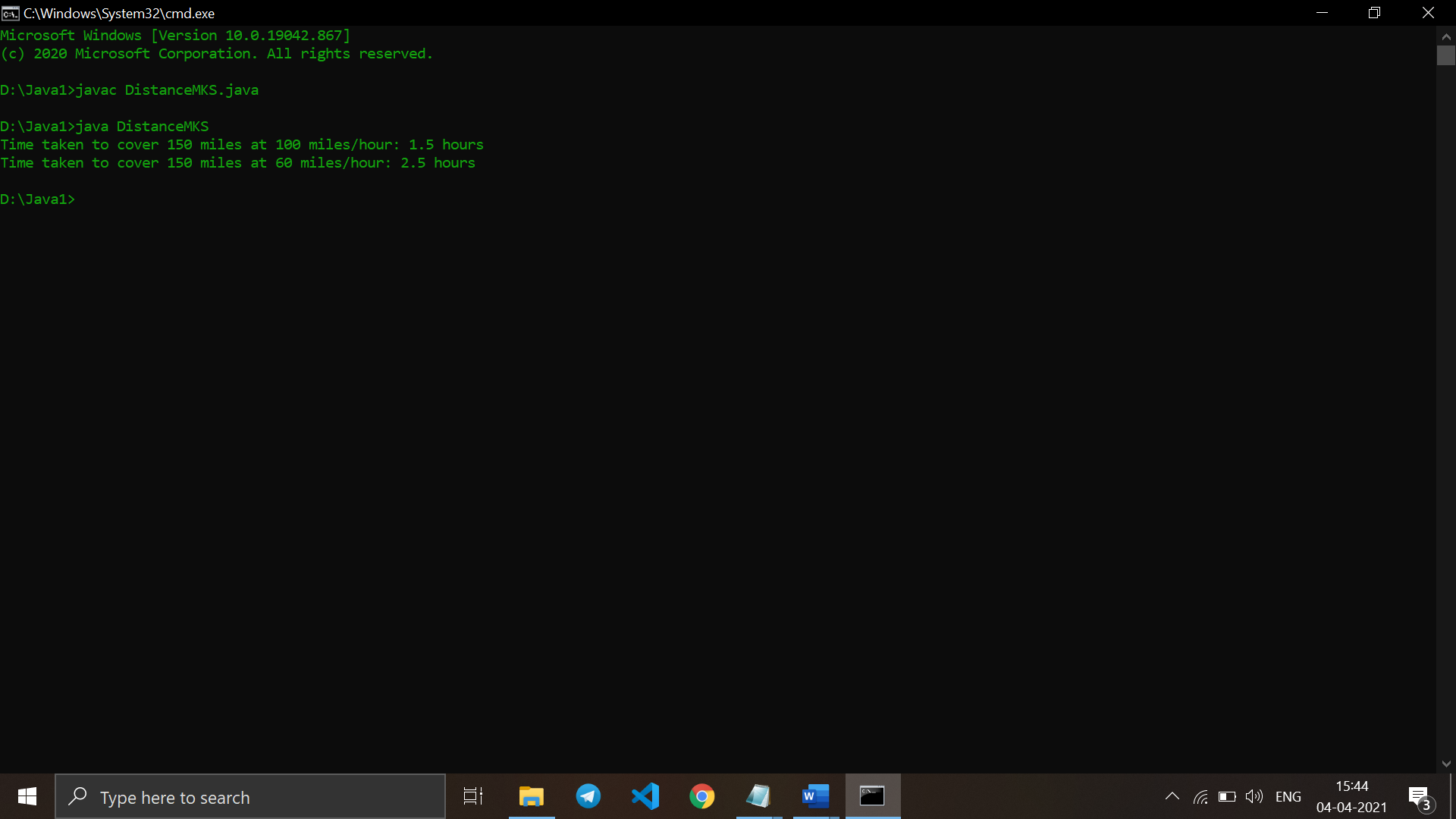
Distance ob1 = new Distance(150);

System.out.println("Time taken to cover 150 miles at 60 miles/hour: " + ob1.travelTime() + " hours");

}

}

Output:



1. **Create a base class called “vehicle” that stores number of wheels and speed.**

**Create the following derived classes –**

**“car” that inherits “vehicle” and also stores number of passengers.**

**“truck” that inherits “vehicle” and also stores the load limit.**

**Write a main function to create objects of these two derived classes and display all**

**the information about “car” and “truck”. Also compare the speed of these two**

**vehicles - car and truck and display which one is faster.**

**Code:**

class Vehicle {

int wheels;

double speed;

Vehicle(int wheels, double speed) {

this.wheels = wheels;

this.speed = speed;

}

double showspeed() {

return (this.speed);

}

int showwheels() {

return (this.wheels);

}

void display() {

System.out.println("Speed: " + this.speed);

System.out.println("No. of Wheels: " + this.wheels);

}

}

class Car extends Vehicle {

int passengers;

Car(int wheels, double speed, int passengers) {

super(wheels, speed);

this.passengers = passengers;

}

void display() {

System.out.println("Car Details:");

super.display();

System.out.println("No. of Passengers: " + this.passengers);

}

}

public class Truck extends Vehicle {

double loadlimit;

Truck(int wheels, double speed, double loadlimit) {

super(wheels, speed);

this.loadlimit = loadlimit;

}

void display() {

System.out.println("Truck Details:");

super.display();

System.out.println("Load Limit: " + this.loadlimit);

}

public static void main(String[] args) {

Vehicle ob = new Car(4, 200.0, 10);

Vehicle ob1 = new Truck(8, 95, 1100.0);

if (ob.showspeed() > ob1.showspeed())

System.out.println("The Car is faster than the Truck");

else if (ob.showspeed() < ob1.showspeed())

System.out.println("The truck is faster than the car");

else

System.out.println("The truck is as fast as the car");

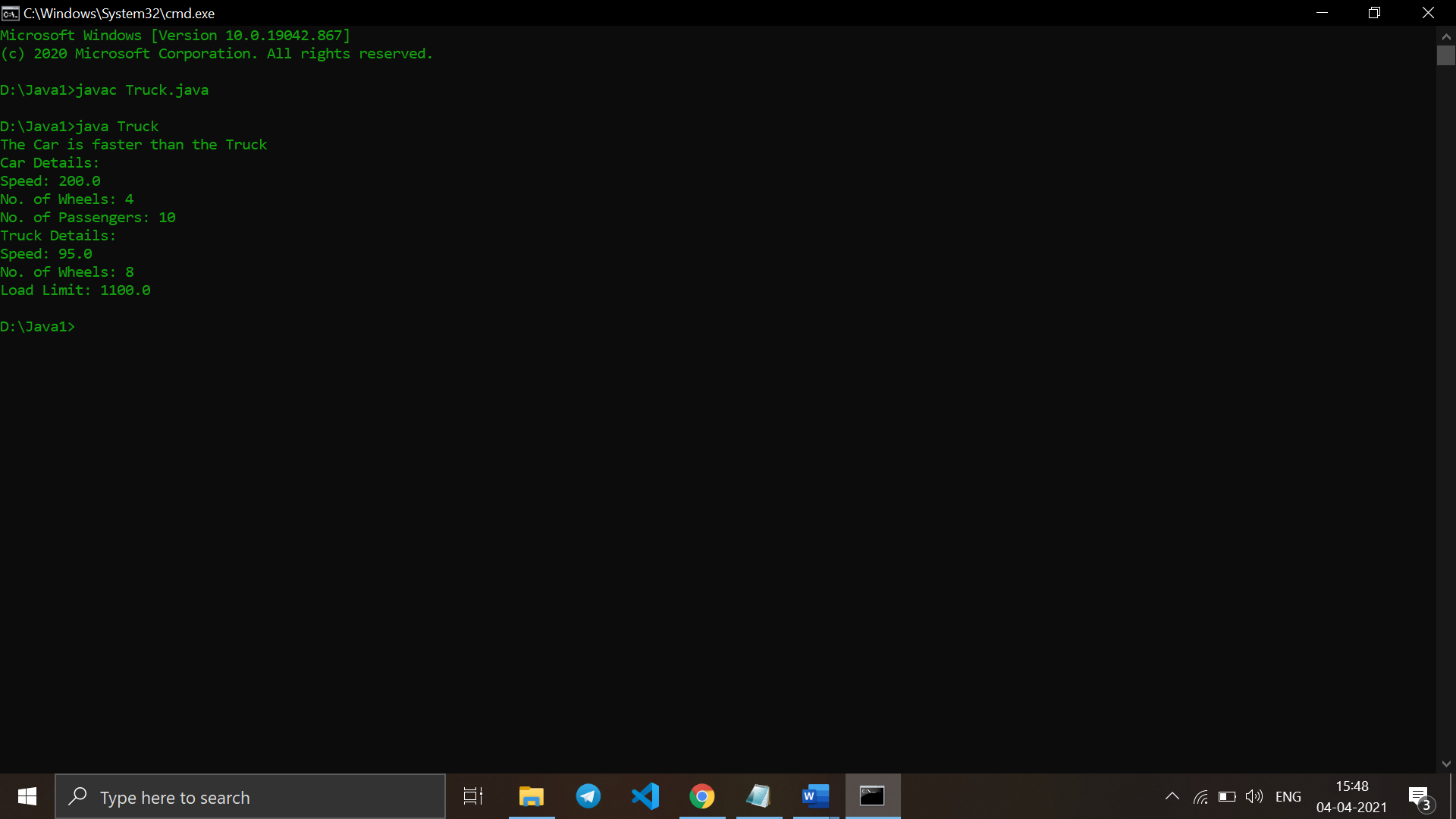
ob.display();

ob1.display();

}

}

Output:



1. **Write a Java program to explain “multilevel inheritance.”**

**Code:**

class Car {

public Car() {

System.out.println("Class Car");

}

public void vehicleType() {

System.out.println("Vehicle Type: Car");

}

}

class Maruti extends Car {

public Maruti() {

System.out.println("Class Maruti");

}

public void brand() {

System.out.println("Brand: Maruti");

}

public void speed() {

System.out.println("Max: 90Kmph");

}

}

public class Maruti800 extends Maruti {

public Maruti800() {

System.out.println("Maruti Model: 800");

}

public void speed() {

System.out.println("Max: 80Kmph");

}

public static void main(String args[]) {

Maruti800 obj = new Maruti800();

obj.vehicleType();

obj.brand();

obj.speed();

}

}

Output:

