**CDAC Mumbai PG-DAC August 24**

**Assignment No- 5**

1. Create a base class BankAccount with methods like deposit() and withdraw(). Derive a class SavingsAccount that overrides the withdraw() method to impose a limit on the withdrawal amount. Write a program that demonstrates the use of overridden methods and proper access modifiers & return the details.

class BankAccount {

protected double balance;

private String accountNumber;

public BankAccount(String accountNumber, double initialBalance) {

this.accountNumber = accountNumber;

this.balance = initialBalance;

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("Deposited: $" + amount);

} else {

System.out.println("Deposit amount must be positive.");

}

}

public void withdraw(double amount) {

if (amount <= balance) {

balance -= amount;

System.out.println("Withdrew: $" + amount);

} else {

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

}

}

public void displayAccountDetails() {

System.out.println("Account Number: " + accountNumber);

System.out.println("Current Balance: $" + balance);

}

}

class SavingsAccount extends BankAccount {

private double withdrawalLimit;

public SavingsAccount(String accountNumber, double initialBalance, double withdrawalLimit) {

super(accountNumber, initialBalance);

this.withdrawalLimit = withdrawalLimit;

}

public void withdraw(double amount) {

if (amount > withdrawalLimit) {

System.out.println("Withdrawal failed: Amount exceeds withdrawal limit of $" + withdrawalLimit);

} else {

super.withdraw(amount);

}

}

public void displayAccountDetails() {

super.displayAccountDetails();

System.out.println("Withdrawal Limit: $" + withdrawalLimit);

}

}

public class Main {

public static void main(String[] args) {

// Create a BankAccount object

BankAccount account1 = new BankAccount("BA123", 5000);

account1.deposit(1000);

account1.withdraw(1500);

account1.displayAccountDetails();

System.***out***.println();

// Create a SavingsAccount object with a withdrawal limit

SavingsAccount account2 = new SavingsAccount("SA456", 3000, 1000);

account2.deposit(500);

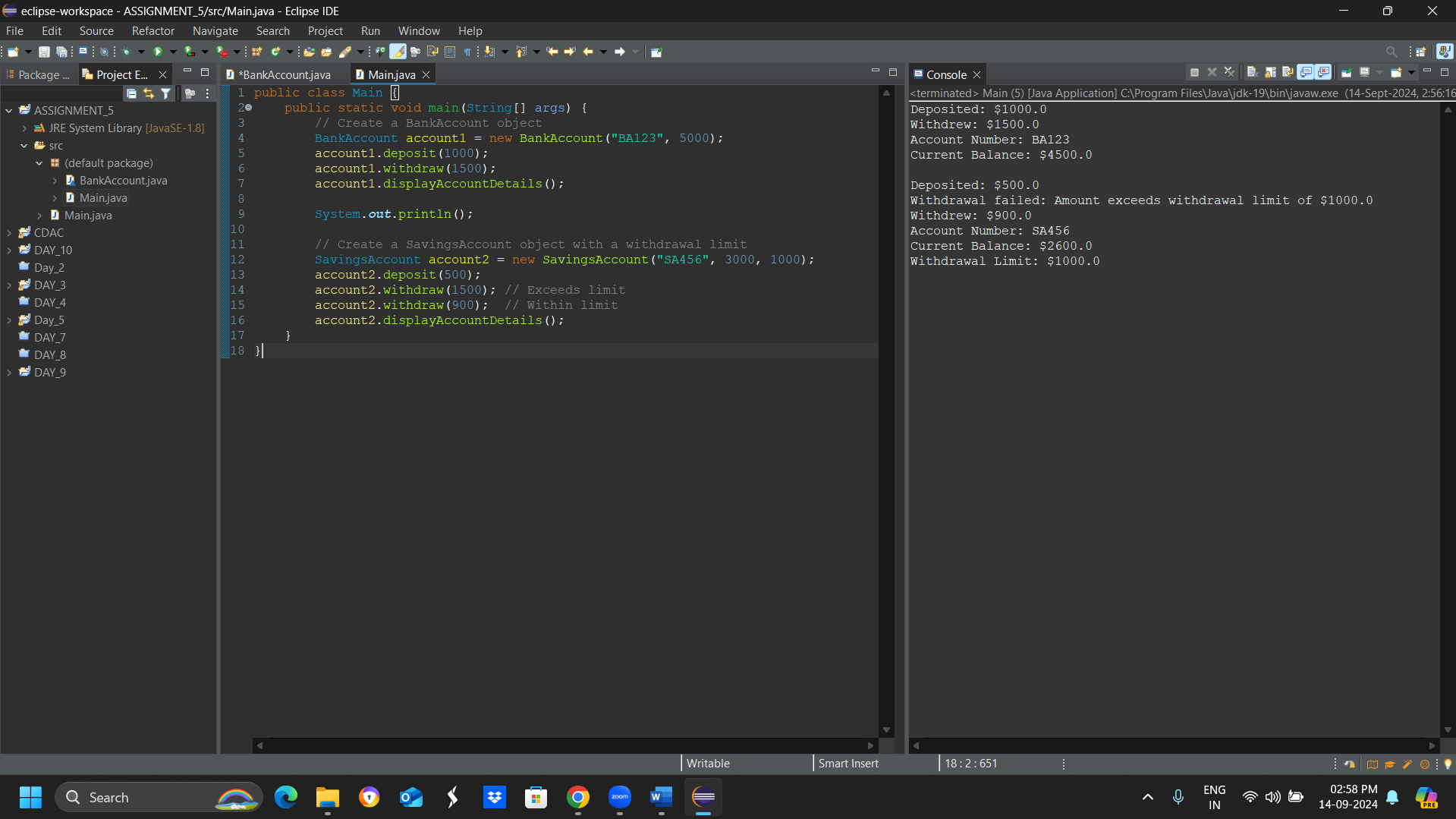
account2.withdraw(1500); // Exceeds limit

account2.withdraw(900); // Within limit

account2.displayAccountDetails();

}

}



1. Create a base class Vehicle with attributes like make and year. Provide a constructor in Vehicle to initialize these attributes. Derive a class Car that has an additional attribute model and write a constructor that initializes make, year, and model. Write a program to create a Car object and display its details.

public class Vehicle {

private String make;

private int year;

public Vehicle(String make, int year) {

this.make = make;

this.year = year;

}

public String getMake() {

return make;

}

public int getYear() {

return year;

}

}

// Car.java

public class Car extends Vehicle {

private String model;

public Car(String make, int year, String model) {

super(make, year);

this.model = model;

}

public String getModel() {

return model;

}

}

public class Main {

public static void main(String[] args) {

Car myCar = new Car("Toyota", 2022, "Corolla");

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

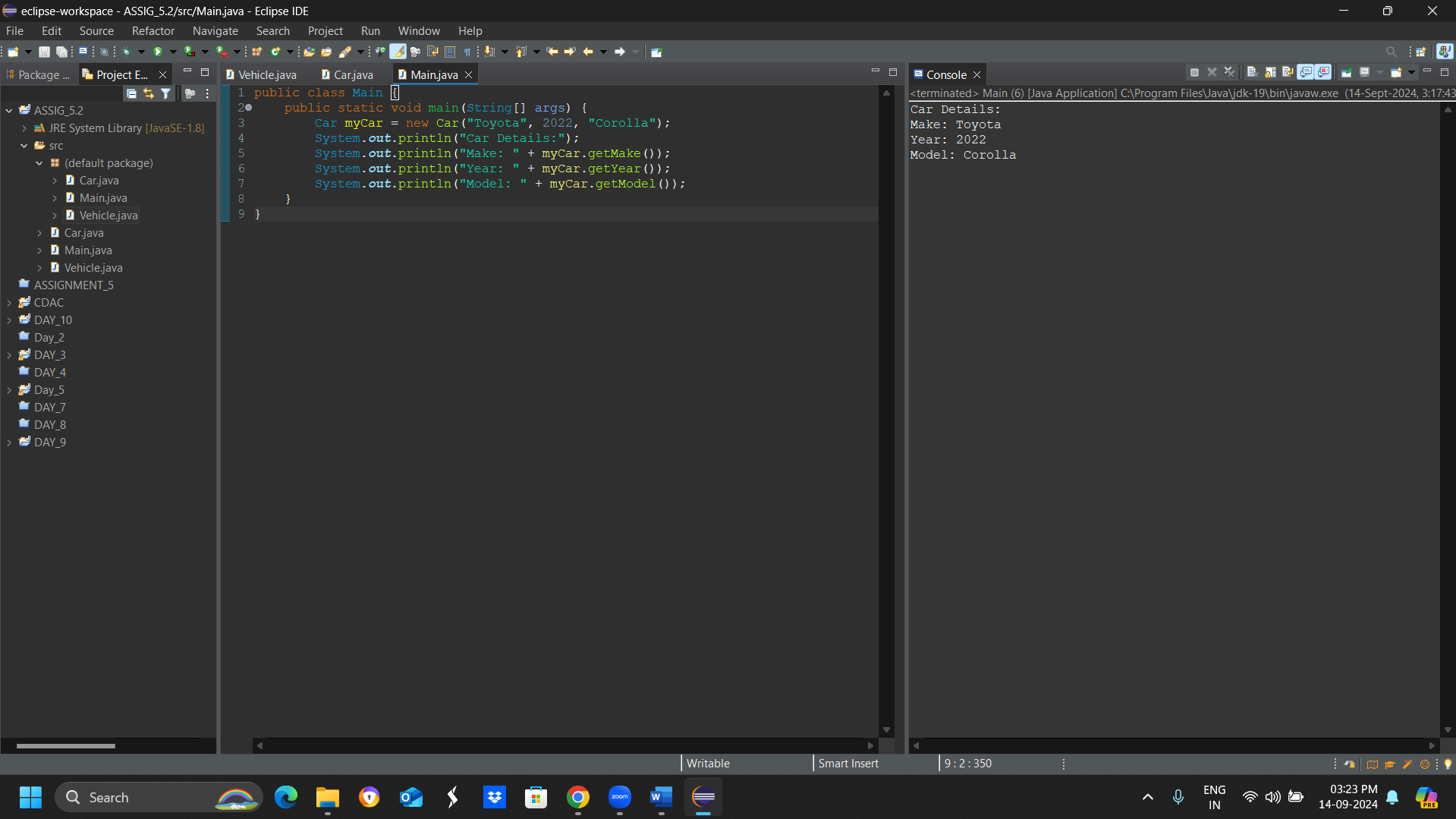
System.***out***.println("Make: " + myCar.getMake());

System.***out***.println("Year: " + myCar.getYear());

System.***out***.println("Model: " + myCar.getModel());

}

}



1. Create a base class Animal with attributes like name, and methods like eat() and sleep(). Create a subclass Dog that inherits from Animal and has an additional method bark(). Write a program to demonstrate the use of inheritance by creating objects of Animal and Dog and calling their methods.

package com.Animal;

class Animal {

// Attributes

String name;

// Constructor to initialize the name

public Animal(String name) {

this.name = name;

}

// Method eat

public void eat() {

System.***out***.println(name + " is eating.");

}

// Method sleep

public void sleep() {

System.***out***.println(name + " is sleeping.");

}

}

package com.Animal;

class Dog extends Animal {

// Constructor to initialize name using the parent class constructor

public Dog(String name) {

super(name); // Calls the constructor of Animal

}

// Additional method specific to Dog

public void bark() {

System.***out***.println(name + " is barking.");

}

}

package com.Animal;

public class Main {

public static void main(String[] args) {

// Create an object of Animal

Animal animal = new Animal("Generic Animal");

animal.eat();

animal.sleep();

// Create an object of Dog

Dog dog = new Dog("Buddy");

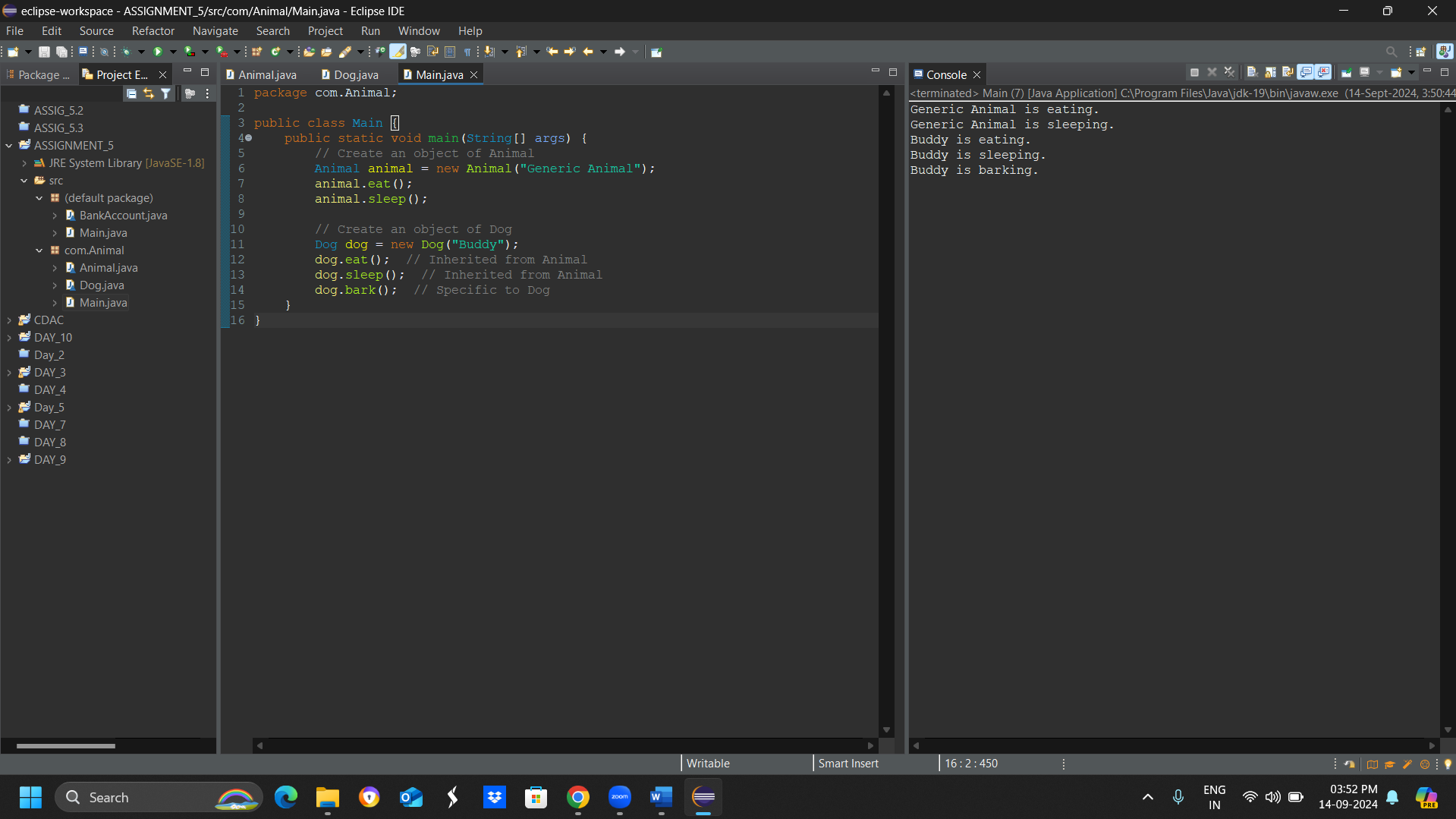
dog.eat(); // Inherited from Animal

dog.sleep(); // Inherited from Animal

dog.bark(); // Specific to Dog

}

}



1. Build a class Student which contains details about the Student and compile and run its

instance.

// Class to represent a Student

class Student {

// Attributes of the student

String name;

int rollNumber;

String course;

// Constructor to initialize the Student object

public Student(String name, int rollNumber, String course) {

this.name = name;

this.rollNumber = rollNumber;

this.course = course;

}

// Method to display student details

public void displayDetails() {

System.out.println("Student Name: " + name);

System.out.println("Roll Number: " + rollNumber);

System.out.println("Course: " + course);

}

}

// Main class to create and run an instance of the Student class

public class Main {

public static void main(String[] args) {

// Create an instance of the Student class

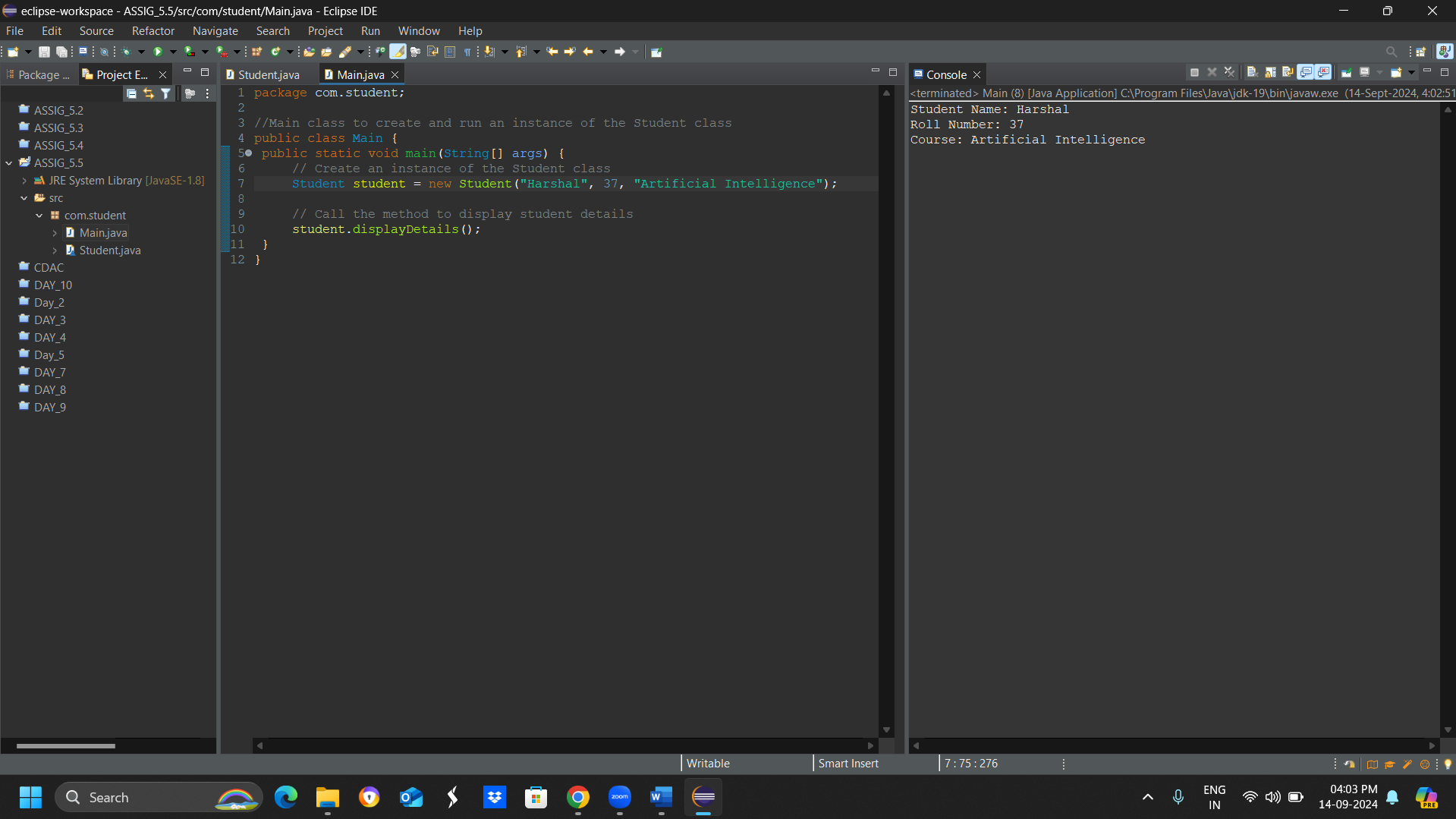
Student student = new Student("Alice", 101, "Computer Science");

// Call the method to display student details

student.displayDetails();

}

}



1. Write a Java program to create a base class Vehicle with methods startEngine() and stopEngine(). Create two subclasses Car and Motorcycle. Override the startEngine() and stopEngine() methods in each subclass to start and stop the engines differently.

// Base class Vehicle

class Vehicle {

// Method to start engine

public void startEngine() {

System.out.println("The vehicle's engine is starting.");

}

// Method to stop engine

public void stopEngine() {

System.out.println("The vehicle's engine is stopping.");

}

}

// Subclass Car inheriting from Vehicle

class Car extends Vehicle {

// Override startEngine method

@Override

public void startEngine() {

System.out.println("The car's engine is starting with a roar.");

}

// Override stopEngine method

@Override

public void stopEngine() {

System.out.println("The car's engine is stopping smoothly.");

}

}

// Subclass Motorcycle inheriting from Vehicle

class Motorcycle extends Vehicle {

// Override startEngine method

@Override

public void startEngine() {

System.out.println("The motorcycle's engine is starting with a loud rev.");

}

// Override stopEngine method

@Override

public void stopEngine() {

System.out.println("The motorcycle's engine is stopping quickly.");

}

}

// Main class to test the Vehicle, Car, and Motorcycle classes

public class Main {

public static void main(String[] args) {

// Create an object of Car

Vehicle car = new Car();

car.startEngine(); // Calls the overridden method in Car

car.stopEngine(); // Calls the overridden method in Car

// Create an object of Motorcycle

Vehicle motorcycle = new Motorcycle();

motorcycle.startEngine(); // Calls the overridden method in Motorcycle

motorcycle.stopEngine(); // Calls the overridden method in Motorcycle

}

}

