```
public class Motorcycle extends Vehicle implements Rentable {
    private double engineCapacity;

    public Motorcycle(String vehicleId, String model, double baseRentalRate, double
engineCapacity) {
        super(vehicleId, model, baseRentalRate);
        this.engineCapacity = engineCapacity;
    }

    public double getEngineCapacity() {
        return engineCapacity;
    }

    public void setEngineCapacity(double engineCapacity) {
        this.engineCapacity = engineCapacity;
    }

    @Override
    public double calculateRentalCost(int days) {
        return getBaseRentalRate() * days;
    }

    @Override
    public boolean isAvailableForRental() {
        return isAvailable();
    }

    @Override
    public void rent(Customer customer, int days) {
        if (!isAvailableForRental()) throw new IllegalStateException("Motorcycle is not
available.");
        setAvailable(false);
        System.out.println("Motorcycle rented by " + customer.getName() + " for " + days + "
days.");
    }

    @Override
    public void returnVehicle() {
        setAvailable(true);
        System.out.println("Motorcycle returned.");
    }
}
```