Class: CMSC203 CRN 20931

 Program: Assignment 2

Instructor: Khandan Monshi

 Summary of Description: Create 3 classes.

 Due Date: 10/04/2024

 Integrity Pledge: I pledge that I have completed the programming assignment independently.

 I have not copied the code from a student or any source.

CLASS Patient:

PRIVATE VARIABLES:

String firstName

String middleName

String lastName

String streetAddress

String city

String state

String zipCode

String emergencyContactName

String emergencyContactPhone

String phoneNumber

// No-arg constructor

METHOD Patient():

DO NOTHING (empty constructor)

// Constructor to initialize first, middle, and last name

METHOD Patient(fName, mName, lName):

SET firstName = fName

SET middleName = mName

SET lastName = lName

// Constructor to initialize all attributes

METHOD Patient(fName, mName, lName, address, cityName, stateName, zip, emergencyName, emergencyPhone):

SET firstName = fName

SET middleName = mName

SET lastName = lName

SET streetAddress = address

SET city = cityName

SET state = stateName

SET zipCode = zip

SET emergencyContactName = emergencyName

SET emergencyContactPhone = emergencyPhone

// Accessor (Getter) methods

METHOD getFirstName():

RETURN firstName

METHOD getMiddleName():

RETURN middleName

METHOD getLastName():

RETURN lastName

METHOD getStreetAddress():

RETURN streetAddress

METHOD getCity():

RETURN city

METHOD getState():

RETURN state

METHOD getZipCode():

RETURN zipCode

METHOD getPhoneNumber():

RETURN phoneNumber

METHOD getEmergencyContactName():

RETURN emergencyContactName

METHOD getEmergencyContactPhone():

RETURN emergencyContactPhone

// Mutator (Setter) methods

METHOD setFirstName(fName):

SET firstName = fName

METHOD setMiddleName(mName):

SET middleName = mName

METHOD setLastName(lName):

SET lastName = lName

METHOD setStreetAddress(address):

SET streetAddress = address

METHOD setCity(cityName):

SET city = cityName

METHOD setState(stateName):

SET state = stateName

METHOD setZipCode(zip):

SET zipCode = zip

METHOD setPhoneNumber(phone):

SET phoneNumber = phone

METHOD setEmergencyContactName(emergencyName):

SET emergencyContactName = emergencyName

METHOD setEmergencyContactPhone(emergencyPhone):

SET emergencyContactPhone = emergencyPhone

// Method to build the full name of the patient

METHOD buildFullName():

RETURN firstName + " " + middleName + " " + lastName

// Method to build the full address of the patient

METHOD buildAddress():

RETURN streetAddress + " " + city + " " + state + " " + zipCode

// Method to build the emergency contact information

METHOD buildEmergencyContact():

RETURN emergencyContactName + " " + emergencyContactPhone

// toString method to display patient information

METHOD toString() OVERRIDE:

RETURN "Name: " + buildFullName() + "\n" +

"Address: " + buildAddress() + "\n" +

"EmergencyContact: " + buildEmergencyContact()

CLASS Procedure:

PRIVATE VARIABLES:

String procedureName

String procedureDate

String practitionerName

double procedureCharges

// No-arg constructor

METHOD Procedure():

DO NOTHING (empty constructor)

// Constructor to initialize procedureName and procedureDate

METHOD Procedure(procName, procDate):

SET procedureName = procName

SET procedureDate = procDate

// Constructor to initialize all attributes

METHOD Procedure(procName, procDate, practitioner, charges):

SET procedureName = procName

SET procedureDate = procDate

SET practitionerName = practitioner

SET procedureCharges = charges

// Accessor (Getter) methods

METHOD getProcedureName():

RETURN procedureName

METHOD getProcedureDate():

RETURN procedureDate

METHOD getPractitionerName():

RETURN practitionerName

// Accessor for charges

METHOD getProcedureCharges():

RETURN procedureCharges

// Mutator (Setter) methods

METHOD setProcedureName(procName):

SET procedureName = procName

METHOD setProcedureDate(procDate):

SET procedureDate = procDate

METHOD setPractitionerName(practitioner):

SET practitionerName = practitioner

METHOD setProcedureCharges(charges):

SET procedureCharges = charges

// toString method to display procedure information

METHOD toString() OVERRIDE:

RETURN "Procedure: " + procedureName + "\n" +

"ProcedureDate=" + procedureDate + "\n" +

"Practitioner=" + practitionerName + "\n" +

"Charge=" + FORMAT(procedureCharges TO 2 decimal places)

CLASS PatientDriverApp:

MAIN METHOD:

DECLARE input AS new Scanner (for keyboard input)

// Step 1: Gather Patient Information from User

PRINT "Enter First Name: "

READ firstName

PRINT "Enter Middle Name: "

READ middleName

PRINT "Enter Last Name: "

READ lastName

PRINT "Enter Street Address: "

READ streetAddress

PRINT "Enter City: "

READ city

PRINT "Enter State: "

READ state

PRINT "Enter Zip Code: "

READ zipCode

PRINT "Enter Emergency Contact Name: "

READ emergencyContactName

PRINT "Enter Emergency Contact Phone: "

READ emergencyContactPhone

// Step 2: Create Patient Object

DECLARE patient AS new Patient (firstName, middleName, lastName, streetAddress, city, state, zipCode, emergencyContactName, emergencyContactPhone)

// Step 3: Create Procedure Objects using different constructors and user input

DECLARE procedure1 AS result of calling gatherProcedureData(input, 1) // Full data constructor

DECLARE procedure2 AS result of calling gatherProcedureDataPartial(input, 2) // Partial data constructor

DECLARE procedure3 AS result of calling gatherProcedureDataNoArg(input, 3) // No-arg constructor with manual setting

// Step 4: Display Patient Information

CALL displayPatient(patient)

// Step 5: Display Procedure Information for all procedures

CALL displayProcedure(procedure1)

CALL displayProcedure(procedure2)

CALL displayProcedure(procedure3)

// Step 6: Calculate and display total charges

DECLARE totalCharges AS result of calling calculateTotalCharges(procedure1, procedure2, procedure3)

PRINT "Total Charges: $" formatted with commas and two decimal places (totalCharges)

// Step 7: Display Student Information

PRINT "Student Name: Ugonna Umunna"

PRINT "МС#: M21189134"

PRINT "Due Date: 10/04/2024"

METHOD displayPatient(patient):

PRINT "Patient info:"

PRINT result of calling patient.toString()

METHOD displayProcedure(procedure):

PRINT procedure.toString()

METHOD calculateTotalCharges(procedure1, procedure2, procedure3):

RETURN procedure1.getProcedureCharges() + procedure2.getProcedureCharges() + procedure3.getProcedureCharges()

METHOD gatherProcedureData(input, procedureNumber):

PRINT "Enter details for Procedure " + procedureNumber

PRINT "Enter Procedure Name: "

READ procedureName

PRINT "Enter Procedure Date (MM/DD/YYYY): "

READ procedureDate

PRINT "Enter Practitioner Name: "

READ practitionerName

PRINT "Enter Procedure Charges: "

READ procedureCharges

RETURN new Procedure (procedureName, procedureDate, practitionerName, procedureCharges)

METHOD gatherProcedureDataPartial(input, procedureNumber):

PRINT "Enter details for Procedure " + procedureNumber

PRINT "Enter Procedure Name: "

READ procedureName

PRINT "Enter Procedure Date (MM/DD/YYYY): "

READ procedureDate

DECLARE procedure AS new Procedure (procedureName, procedureDate)

PRINT "Enter Practitioner Name: "

READ practitionerName

CALL procedure.setPractitionerName(practitionerName)

PRINT "Enter Procedure Charges: "

READ procedureCharges

CALL procedure.setProcedureCharges(procedureCharges)

RETURN procedure

METHOD gatherProcedureDataNoArg(input, procedureNumber):

PRINT "Enter details for Procedure " + procedureNumber

DECLARE procedure AS new Procedure() // No-arg constructor

PRINT "Enter Procedure Name: "

READ procedureName

CALL procedure.setProcedureName(procedureName)

PRINT "Enter Procedure Date (MM/DD/YYYY): "

READ procedureDate

CALL procedure.setProcedureDate(procedureDate)

PRINT "Enter Practitioner Name: "

READ practitionerName

CALL procedure.setPractitionerName(practitionerName)

PRINT "Enter Procedure Charges: "

READ procedureCharges

CALL procedure.setProcedureCharges(procedureCharges)

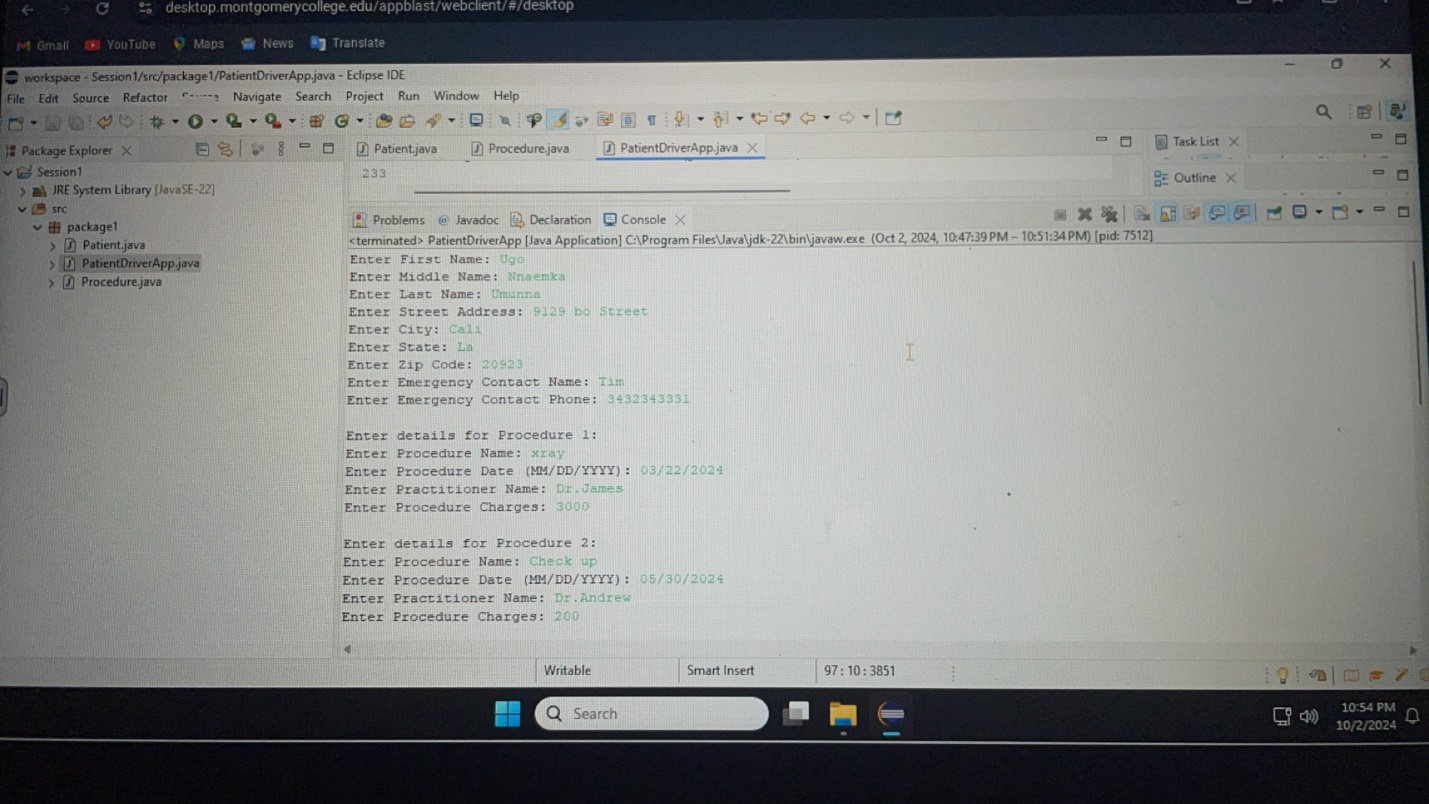
RETURN procedure

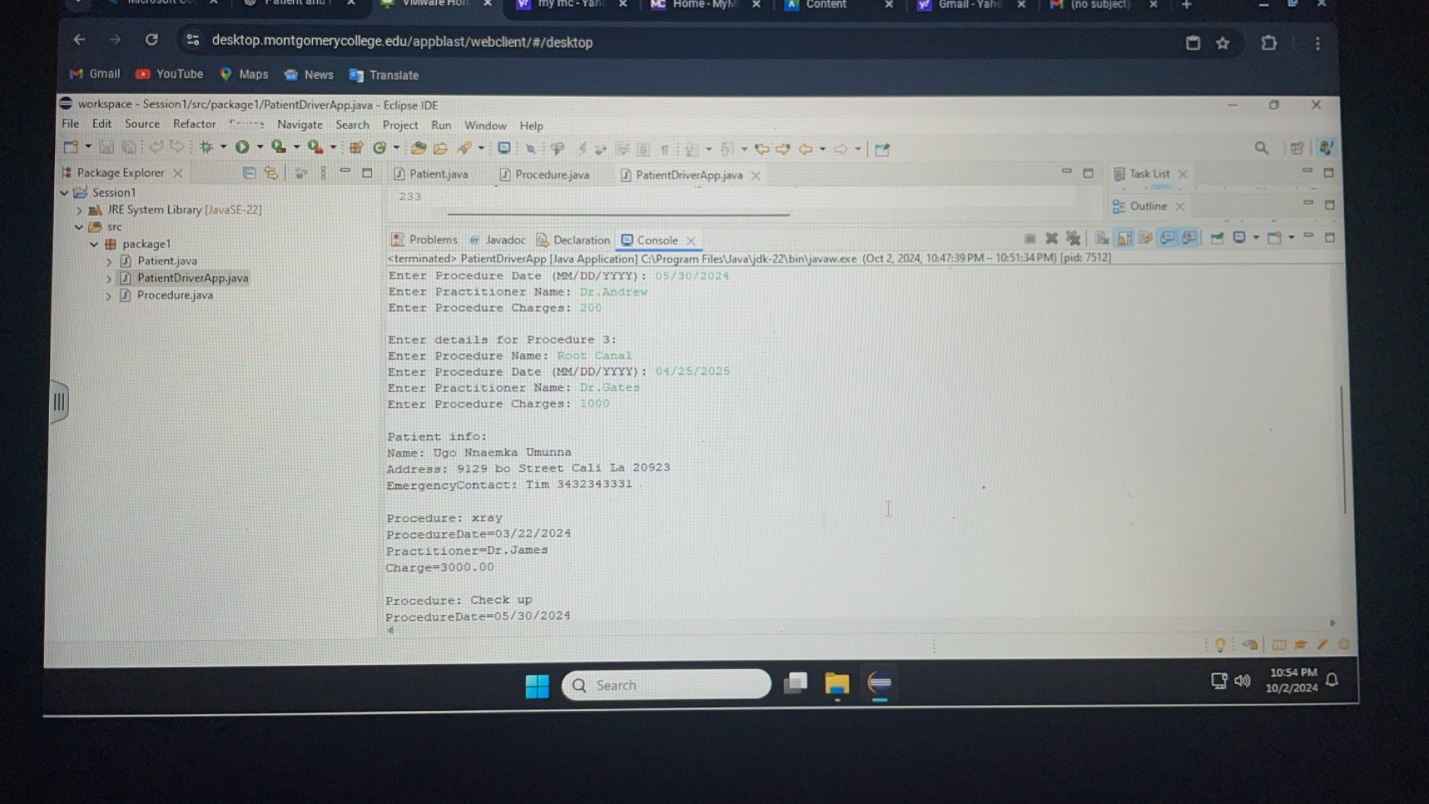
|  |
| --- |
| Patient |
| -firstName: String  -middleName: String  -lastName: String  -streetAddress: String  -city: String  -state: String  -zipCode: String  -phoneNumber: String  -emergencyContactName: String  -emergencyContactPhone: String |
| +Patient()  +Patient(firstName: String, middleName: String, lastName: String)  +Patient(firstName: String, middleName: String, lastName: String, streetAddress: String, city: String, state: String, zipCode: String, phoneNumber: String, emergencyContactName: String, emergencyContactPhone: String)  +getFirstName(): String  +setFirstName(firstName: String): void  +getMiddleName(): String  +setMiddleName(middleName: String): void  +getLastName(): String  +setLastName(lastName: String): void  +getStreetAddress(): String  +setStreetAddress(streetAddress: String): void  +getCity(): String  +setCity(city: String): void  +getState(): String  +setState(state: String): void  +getZipCode(): String  +setZipCode(zipCode: String): void  +getPhoneNumber(): String  +setPhoneNumber(phoneNumber: String): void  +getEmergencyContactName(): String  +setEmergencyContactName(emergencyContactName: String): void  +getEmergencyContactPhone(): String  +setEmergencyContactPhone(emergencyContactPhone: String): void  +buildFullName(): String  +buildAddress(): String  +buildEmergencyContact(): String  +toString(): String |

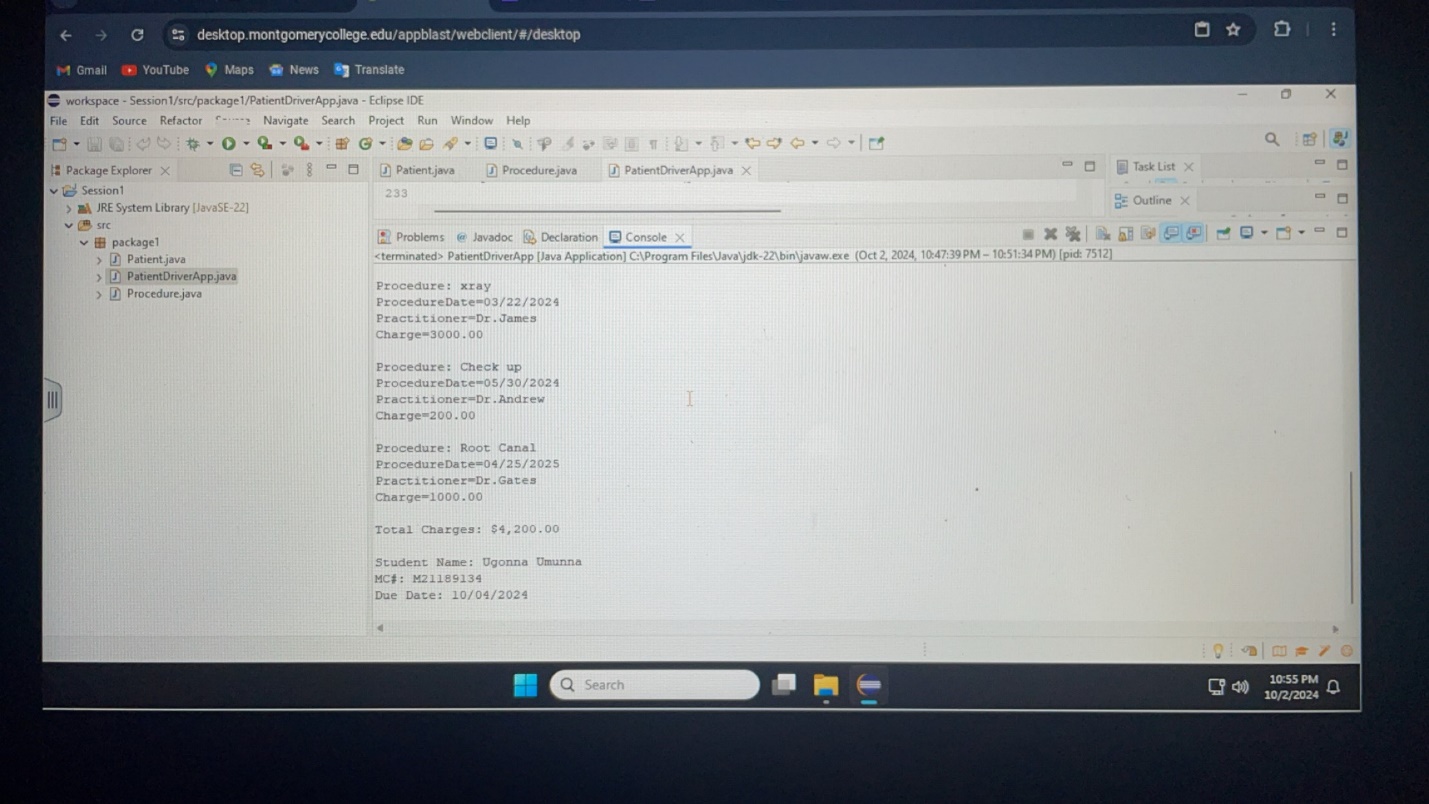
|  |
| --- |
| Procedure |
| -procedureName: String  -procedureDate: String  -practitionerName: String  -procedureCharges: double |
| +Procedure()  +Procedure(procedureName: String, procedureDate: String)  +Procedure(procedureName: String, procedureDate: String, practitionerName: String, procedureCharges: double)  +getProcedureName(): String  +setProcedureName(procedureName: String): void  +getProcedureDate(): String  +setProcedureDate(procedureDate: String): void  +getPractitionerName(): String  +setPractitionerName(practitionerName: String): void  +getProcedureCharges(): double  +setProcedureCharges(procedureCharges: double): void  +toString(): String |

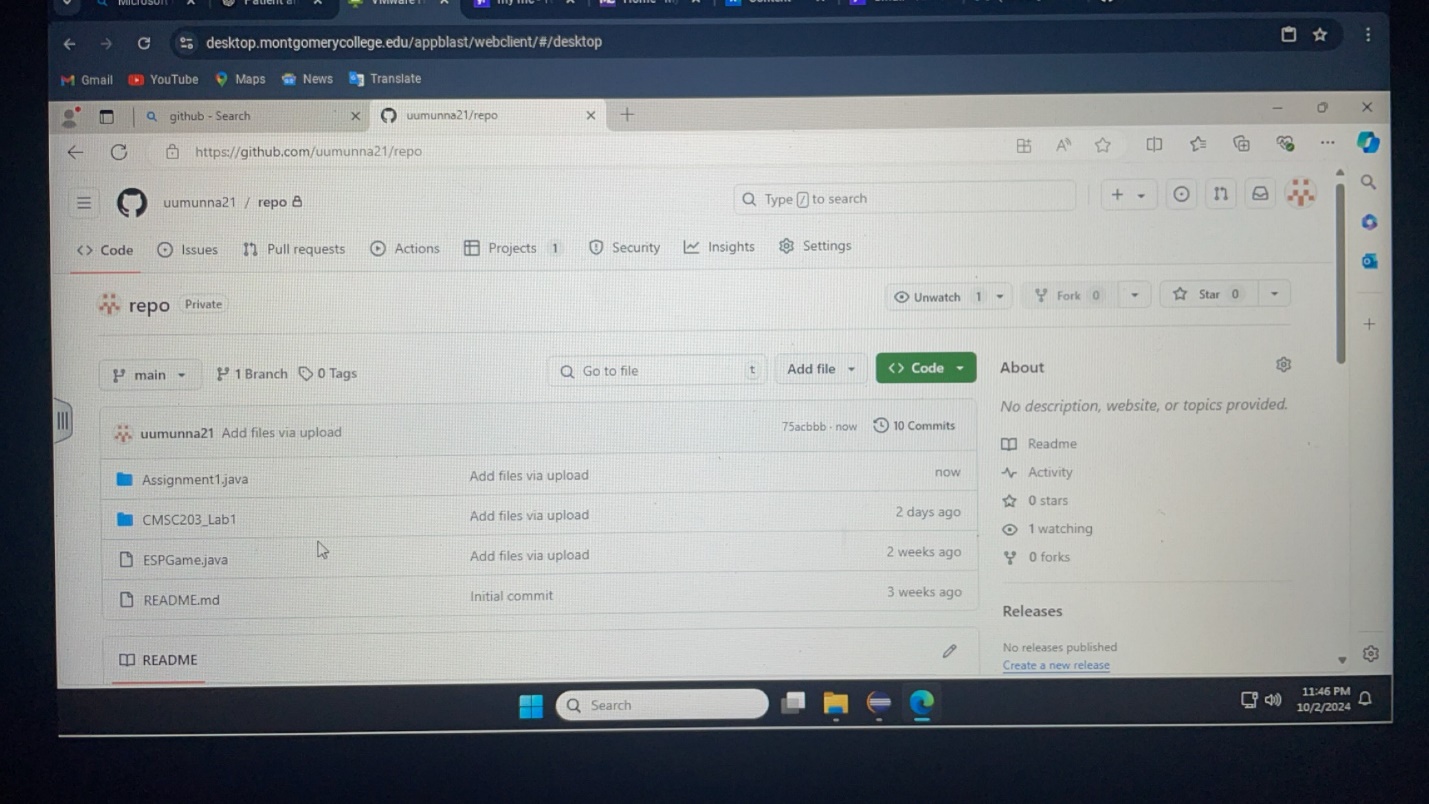
|  |
| --- |
| PatientDriverApp |
|  |
| +main(args: String[]): void  +displayPatient(patient: Patient): void  +displayProcedure(procedure: Procedure): void  +calculateTotalCharges(procedure1: Procedure, procedure2: Procedure, procedure3: Procedure): double |

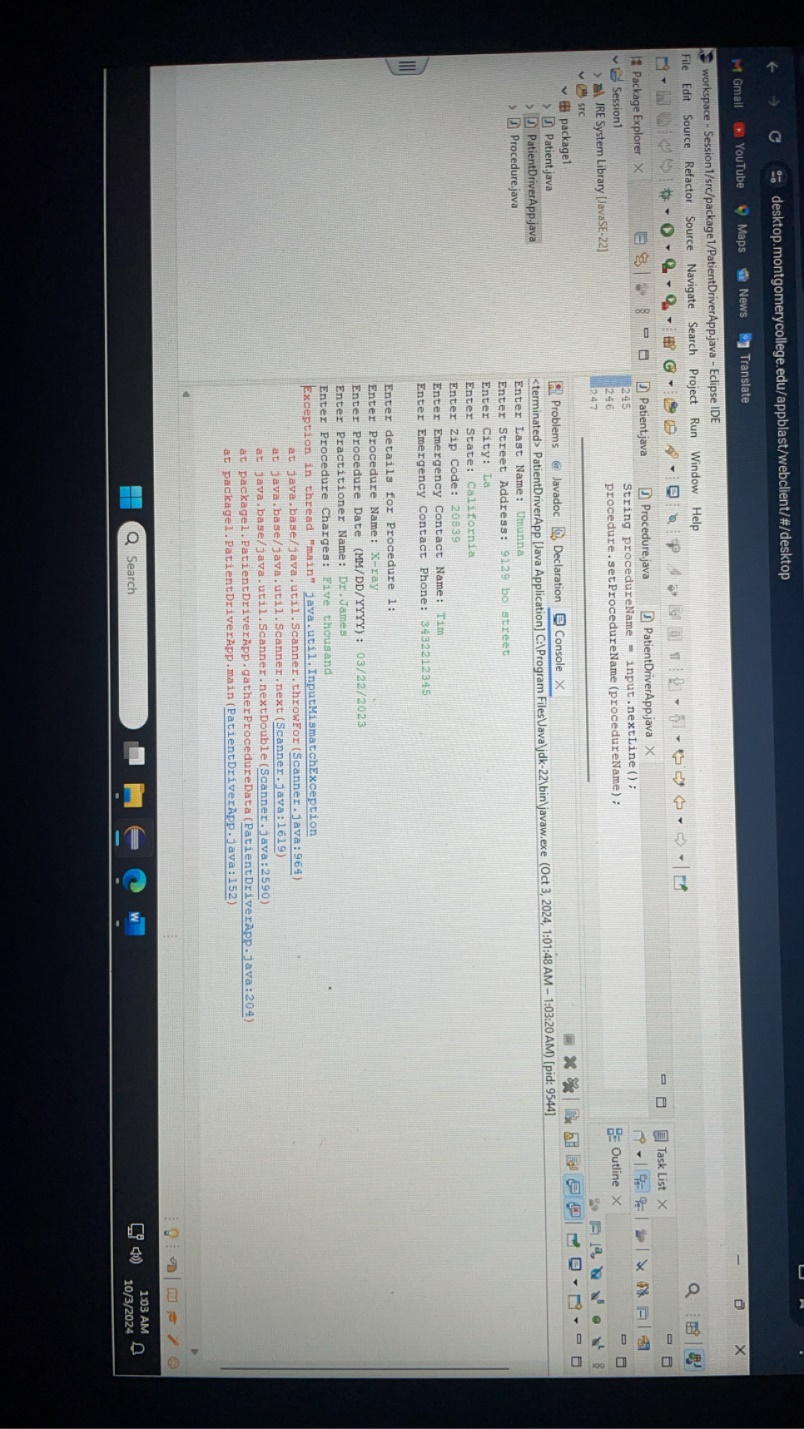
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cases | Input | Expected Output | Actual Output | Did Test Pass? |
| Case 1 | Ugo | Enter Middle Name | Enter Middle Name | Yes |
| Case 2 | Five thousand | Enter details for procedure 2 | Program crahsed | No |
| Case 3 | La | Enter State | Enter State | Yes |
| Case 4 | California | Enter Zip Code | Enter Zip Code | Yes |

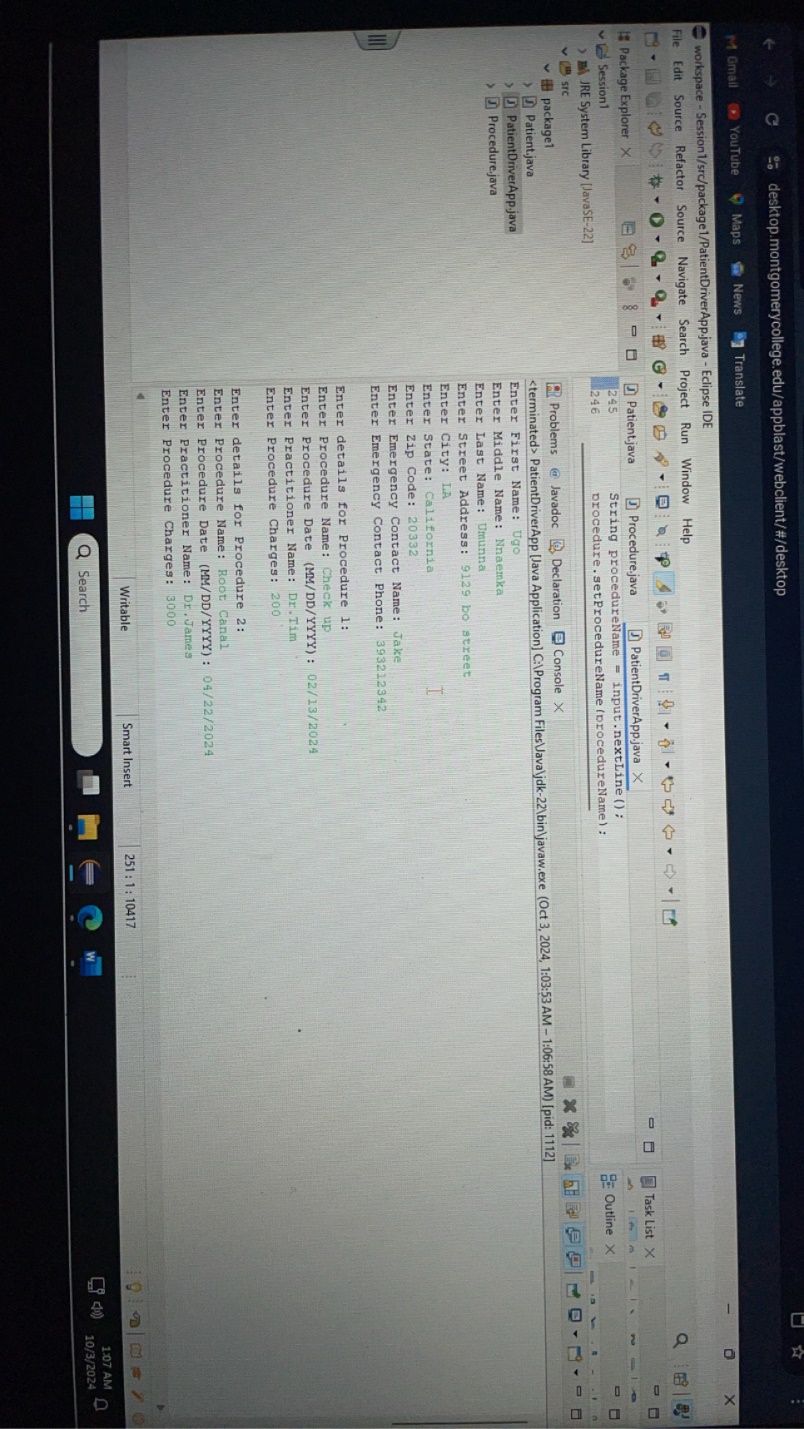


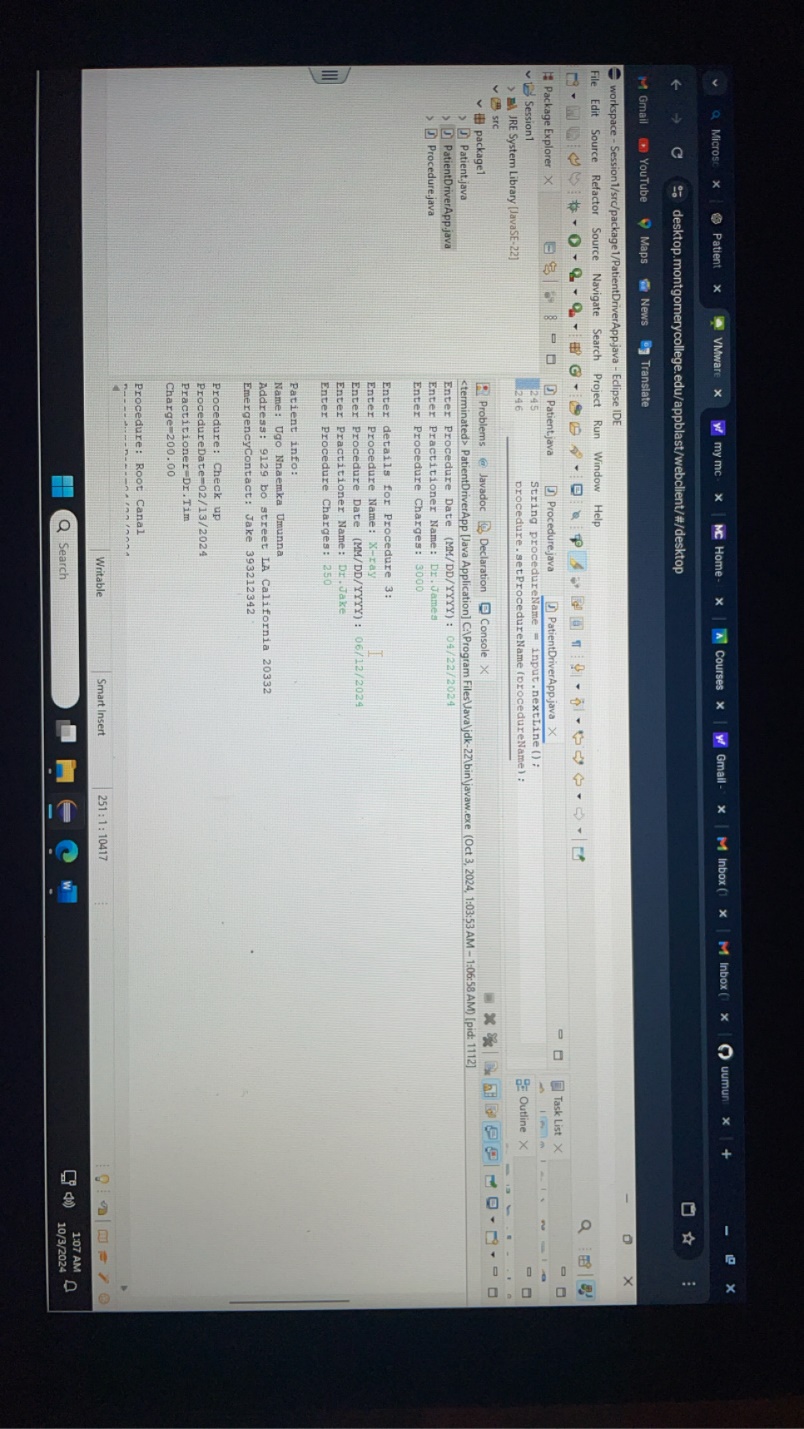


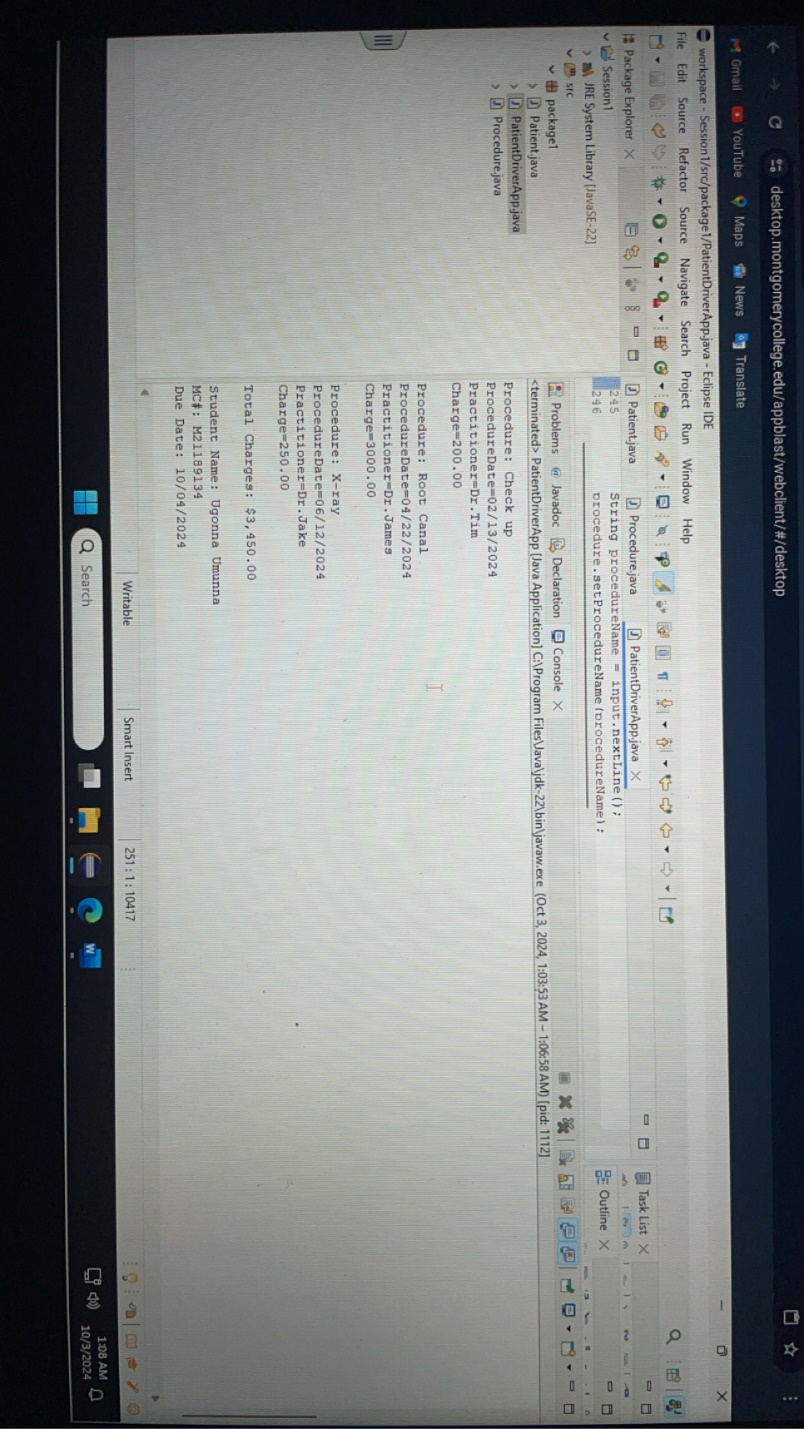












Write about your Learning Experience, highlighting your lessons learned and learning experience from working on this project.

What have you learned? I have learnt how to create classes

What did you struggle with? Loading the main class package

What would you do differently on your next project? Work more organized

What parts of this assignment were you successful with, and what parts (if any) were you not successful with? I was successful in hiding data.

Provide any additional resources/links/videos you used to while working on this assignment/project.

**Check List:** <Provide answers to the column Y/N or N/A >**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** |  | **Y/N** | **Comments** |
|  | **Assignment files:** | Y |  |
|  | * FirstInitialLastName\_ Assignment#\_Moss.zip | Y |  |
|  | * FirstInitialLastName\_Assignment#.docx/.pdf | Y |  |
|  | * Source java files | Y |  |
|  | **Program compiles** | Y |  |
|  | **Program runs with desired outputs related to a Test Plan** | Y |  |
|  | **Documentation file:** | Y |  |
|  | * Comprehensive Test Plan | Y |  |
|  | * Screenshots related to the Test Plan | Y |  |
|  | * Screenshots of your GitHub account with submitted Assignment# (if required) | Y |  |
|  | * UML Diagram (if required) | Y |  |
|  | * Algorithms/Pseudocode (if required) | Y |  |
|  | * Flowchart (if required) | Y |  |
|  | * Lessons Learned | Y |  |
|  | * Checklist is completed and included in the Documentation | Y |  |