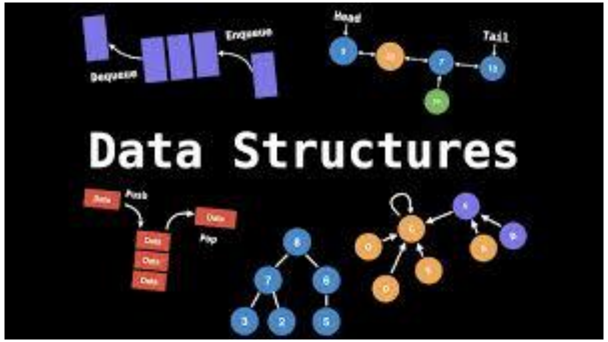
Data Structures

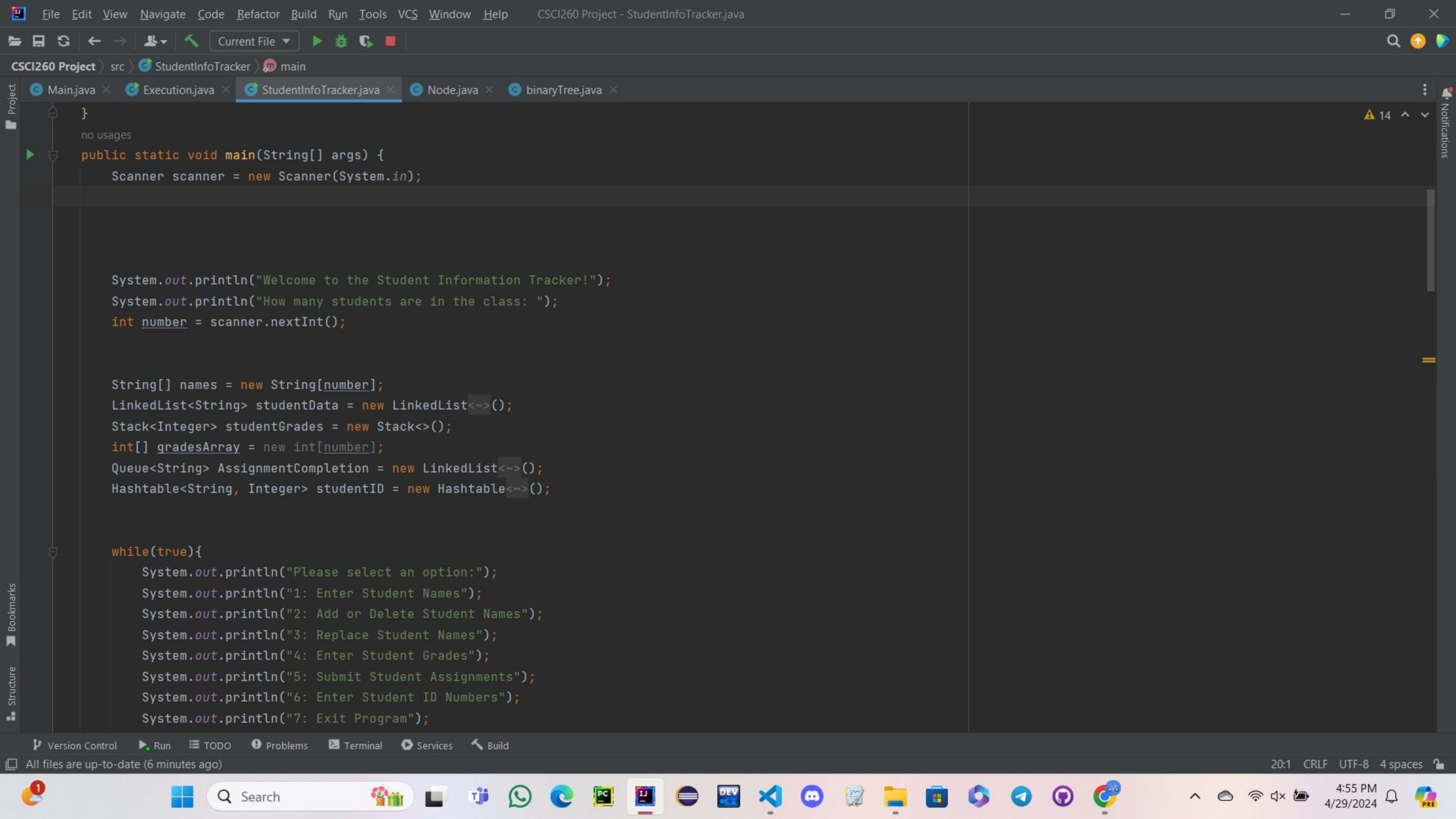
Spring 2024 W01

Zeal Patel and Justin Sewnarine

Team Name: Programming City Thunder



Problem statement: Our software is designed to ease management of educational data in classroom settings. This software will help with efficient tracking of student data in classes. Data that is kept track of can include things such as student names, student ID's, student grades etc. This software, if implemented in educational institutions, can help lower mistakes that teachers or professors might make on occasions. The software also brings along with itself the ability to track student assignments. The software was created with the assistance of different algorithms such as Arrays, Linked lists, Stacks, Queues, Hash Tables, Bubble Sort, and Binary Tree.

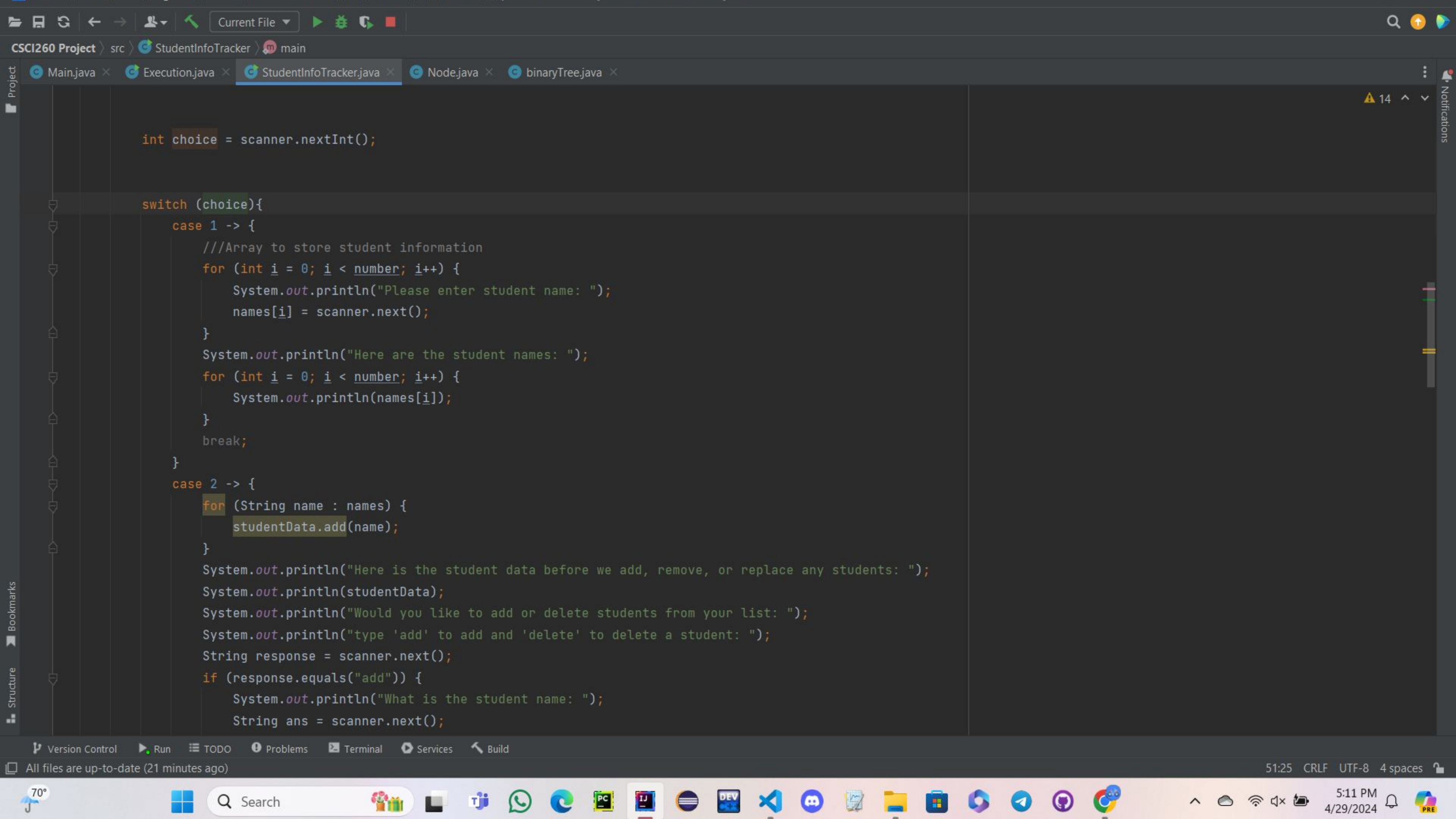


```java
    }
    no usages
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);




        System.out.println("Welcome to the Student Information Tracker!");
        System.out.println("How many students are in the class: ");
        int number = scanner.nextInt();


        String[] names = new String[number];
        LinkedList<String> studentData = new LinkedList<~>();
        Stack<Integer> studentGrades = new Stack<>();
        int[] gradesArray = new int[number];
        Queue<String> AssignmentCompletion = new LinkedList<~>();
        Hashtable<String, Integer> studentID = new Hashtable<~>();


        while(true){
            System.out.println("Please select an option:");
            System.out.println("1: Enter Student Names");
            System.out.println("2: Add or Delete Student Names");
            System.out.println("3: Replace Student Names");
            System.out.println("4: Enter Student Grades");
            System.out.println("5: Submit Student Assignments");
            System.out.println("6: Enter Student ID Numbers");
            System.out.println("7: Exit Program");
```

CSCI260 Project - StudentInfoTracker.java
CSCI260 Project 〉 src 〉 StudentInfoTracker 〉 main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
All files are up-to-date (6 minutes ago)
20:1 CRLF UTF-8 4 spaces
4:55 PM
4/29/2024

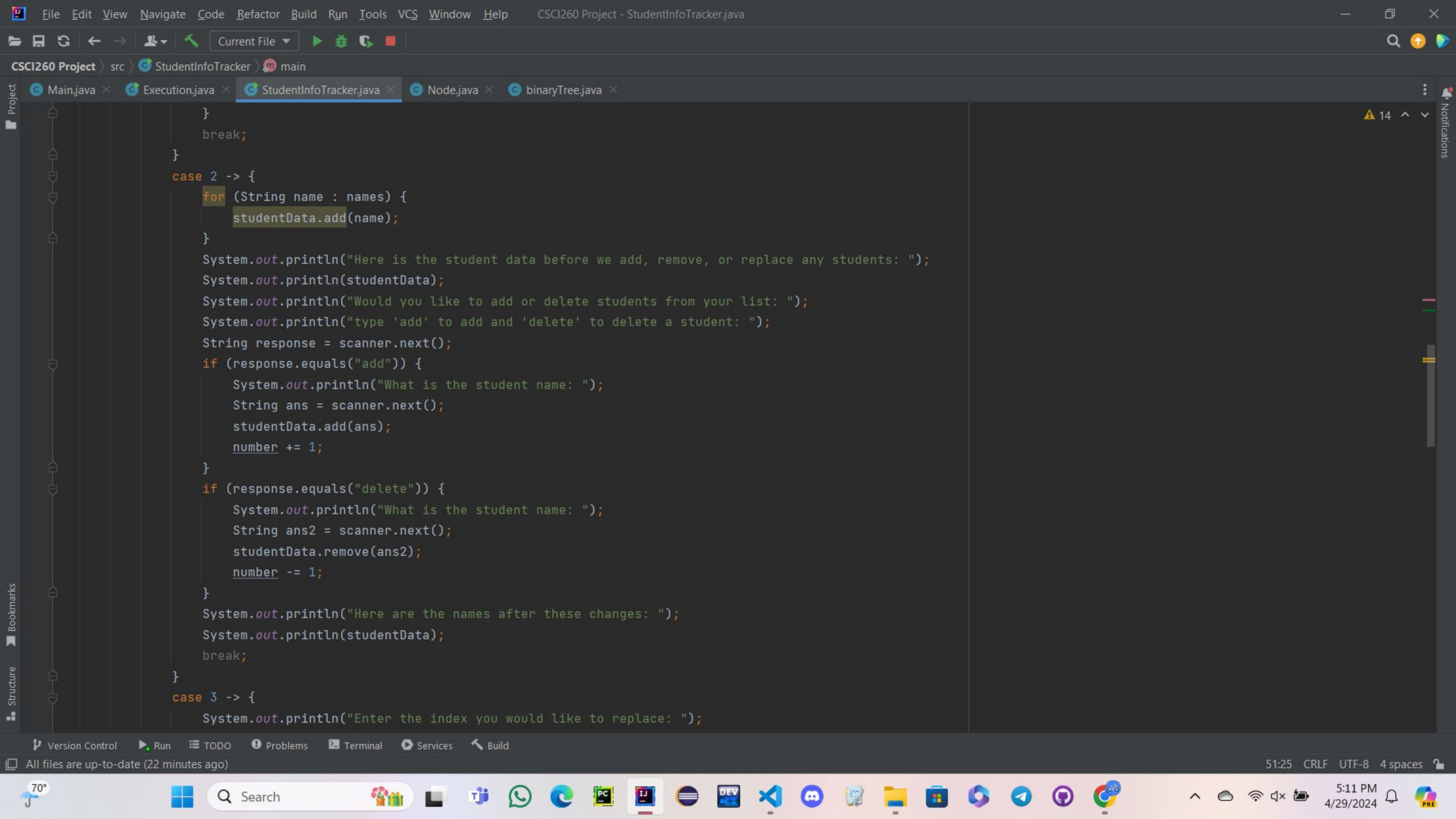
- Code Explanation

➔ Scanner class is used in order to get input from the user; the user is asked to enter the number of students. Several Data structures are initialized and these include an array to store student names, linked list to store other student information, Stack to store grades, Queues to store student assignments, an array to store student grades/ sort them from least to greatest using bubble sort, and a hashtable top store student ID numbers. A while loop is used to display to the user a menu of options such as add student name, grade, number, assignment, ID and waits for the user to select a option.



```
        int choice = scanner.nextInt();


        switch (choice){
            case 1 -> {
                ///Array to store student information
                for (int i = 0; i < number; i++) {
                    System.out.println("Please enter student name: ");
                    names[i] = scanner.next();
                }
                System.out.println("Here are the student names: ");
                for (int i = 0; i < number; i++) {
                    System.out.println(names[i]);
                }
                break;
            }
            case 2 -> {
                for (String name : names) {
                    studentData.add(name);
                }
                System.out.println("Here is the student data before we add, remove, or replace any students: ");
                System.out.println(studentData);
                System.out.println("Would you like to add or delete students from your list: ");
                System.out.println("type 'add' to add and 'delete' to delete a student: ");
                String response = scanner.next();
                if (response.equals("add")) {
                    System.out.println("What is the student name: ");
                    String ans = scanner.next();
```

All files are up-to-date (21 minutes ago)


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help CSCI260 Project - StudentInfoTracker.java

CSCI260 Project 〉 src 〉 StudentInfoTracker 〉 main

Main.java | Execution.java | StudentInfoTracker.java | Node.java | binaryTree.java

```java
                }
                break;
            }
            case 2 -> {
                for (String name : names) {
                    studentData.add(name);
                }
                System.out.println("Here is the student data before we add, remove, or replace any students: ");
                System.out.println(studentData);
                System.out.println("Would you like to add or delete students from your list: ");
                System.out.println("type 'add' to add and 'delete' to delete a student: ");
                String response = scanner.next();
                if (response.equals("add")) {
                    System.out.println("What is the student name: ");
                    String ans = scanner.next();
                    studentData.add(ans);
                    number += 1;
                }
                if (response.equals("delete")) {
                    System.out.println("What is the student name: ");
                    String ans2 = scanner.next();
                    studentData.remove(ans2);
                    number -= 1;
                }
                System.out.println("Here are the names after these changes: ");
                System.out.println(studentData);
                break;
            }
            case 3 -> {
                System.out.println("Enter the index you would like to replace: ");
```

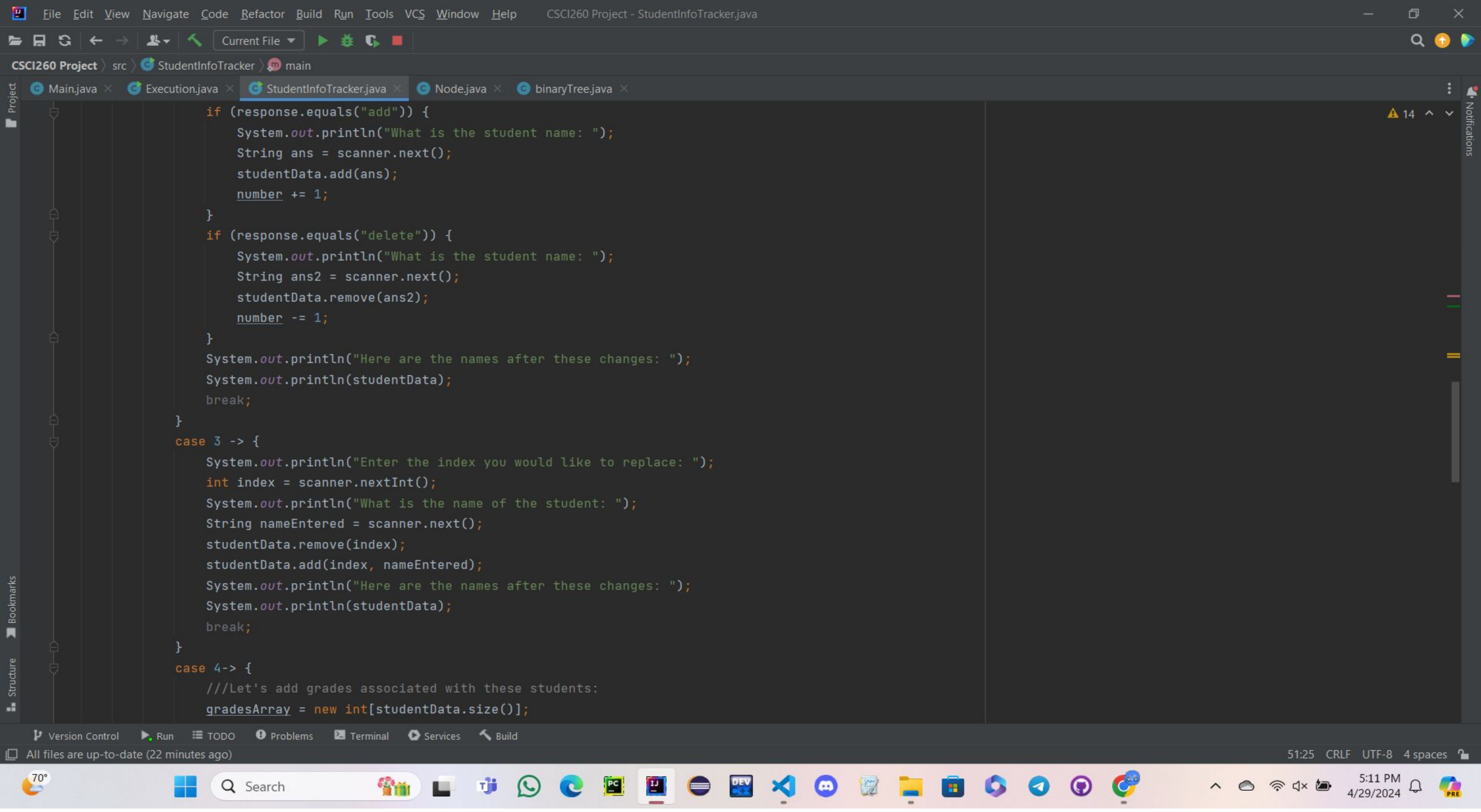
Version Control | Run | TODO | Problems | Terminal | Services | Build

All files are up-to-date (22 minutes ago) 51:25 CRLF UTF-8 4 spaces

70° Search 5:11 PM 4/29/2024

- Code Explanation

- ➔ Switch statement : performs different actions based on the action that the user decides to do
- ➔ Case 1: User is prompted to enter student name and these names are stores in the “names” array. It prints out all the entered name
- ➔ Case2: Allows to add or delete student name. It adds all the names from names array to student data which is the linked list. User is asked whether they want to add or delete a student. If option add is chosen, it asks for name and adds it to student data list. If delete option is chosen, it asks student for the name and removes it from student data list. It shows the updated list with all the student names after the process of adding and deleting has taken place


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help CSCI260 Project - StudentInfoTracker.java

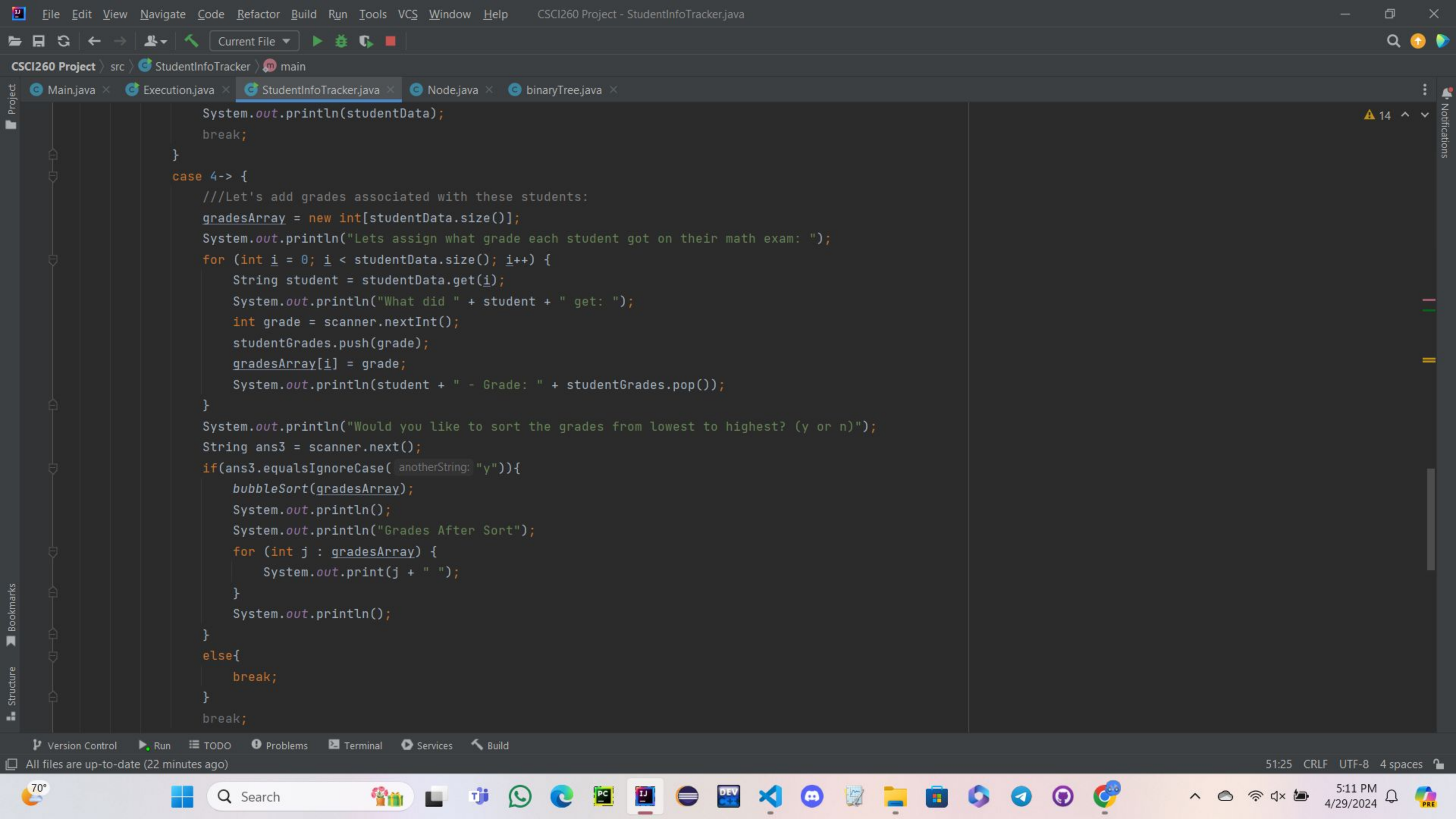
CSCI260 Project 〉 src 〉 StudentInfoTracker 〉 main

Main.java | Execution.java | StudentInfoTracker.java | Node.java | binaryTree.java

```java
                if (response.equals("add")) {
                    System.out.println("What is the student name: ");
                    String ans = scanner.next();
                    studentData.add(ans);
                    number += 1;
                }
                if (response.equals("delete")) {
                    System.out.println("What is the student name: ");
                    String ans2 = scanner.next();
                    studentData.remove(ans2);
                    number -= 1;
                }
                System.out.println("Here are the names after these changes: ");
                System.out.println(studentData);
                break;
            }
            case 3 -> {
                System.out.println("Enter the index you would like to replace: ");
                int index = scanner.nextInt();
                System.out.println("What is the name of the student: ");
                String nameEntered = scanner.next();
                studentData.remove(index);
                studentData.add(index, nameEntered);
                System.out.println("Here are the names after these changes: ");
                System.out.println(studentData);
                break;
            }
            case 4-> {
                ///Let's add grades associated with these students:
                gradesArray = new int[studentData.size()];
```

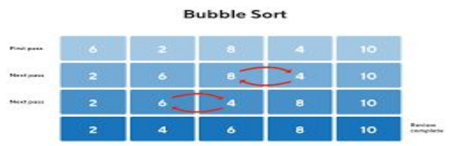
All files are up-to-date (22 minutes ago) 51:25 CRLF UTF-8 4 spaces

5:11 PM 4/29/2024


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - StudentInfoTracker.java
Current File
CSCI260 Project › src › StudentInfoTracker › main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
Version Control
Run
TODO
Problems
Terminal
Services
Build
All files are up-to-date (22 minutes ago)
51:25 CRLF UTF-8 4 spaces

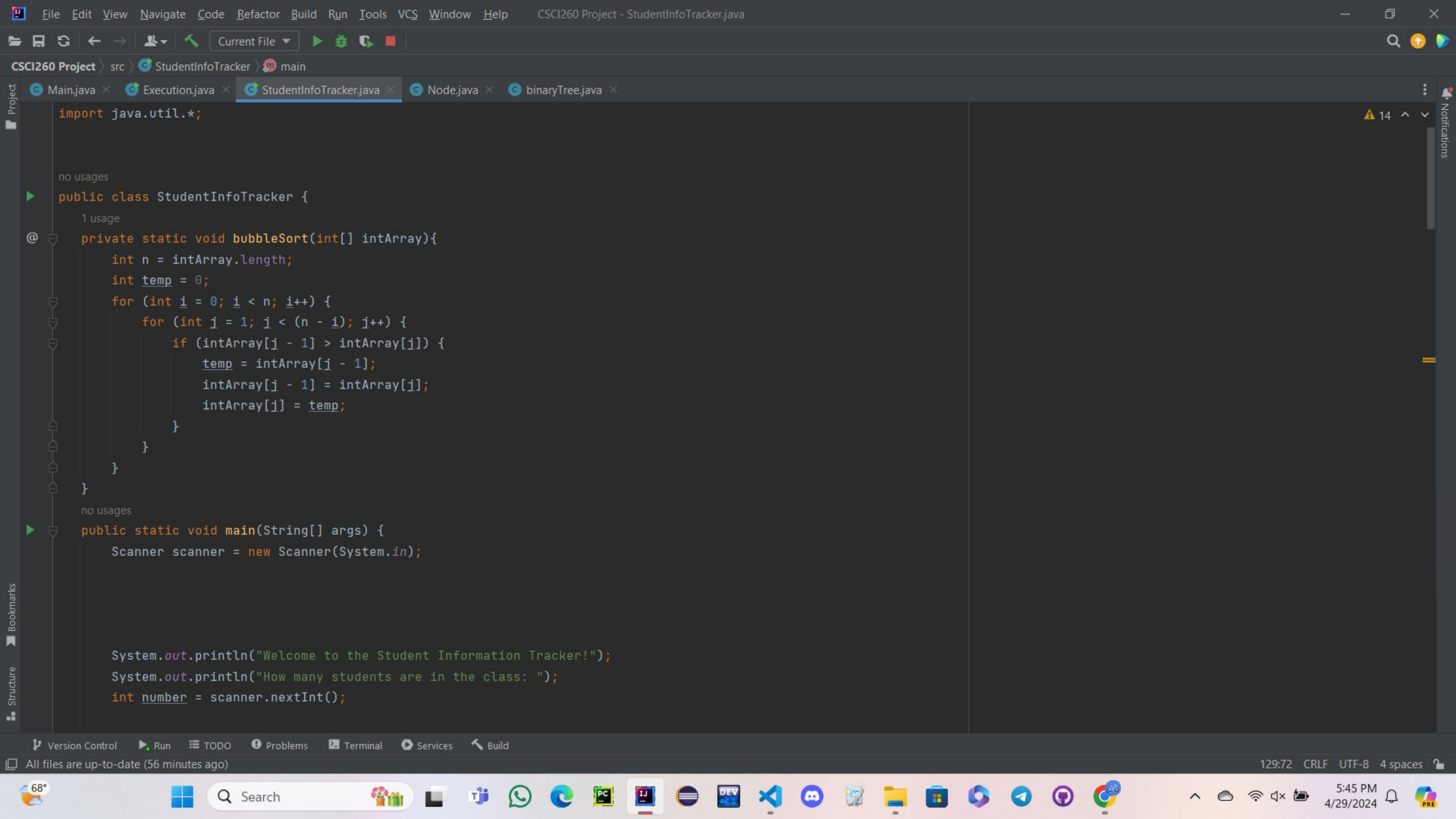
```
                System.out.println(studentData);
                break;
            }
            case 4-> {
                ///Let's add grades associated with these students:
                gradesArray = new int[studentData.size()];
                System.out.println("Lets assign what grade each student got on their math exam: ");
                for (int i = 0; i < studentData.size(); i++) {
                    String student = studentData.get(i);
                    System.out.println("What did " + student + " get: ");
                    int grade = scanner.nextInt();
                    studentGrades.push(grade);
                    gradesArray[i] = grade;
                    System.out.println(student + " - Grade: " + studentGrades.pop());
                }
                System.out.println("Would you like to sort the grades from lowest to highest? (y or n)");
                String ans3 = scanner.next();
                if(ans3.equalsIgnoreCase( anotherString: "y")){
                    bubbleSort(gradesArray);
                    System.out.println();
                    System.out.println("Grades After Sort");
                    for (int j : gradesArray) {
                        System.out.print(j + " ");
                    }
                    System.out.println();
                }
                else{
                    break;
                }
                break;
```

- Code Explanation

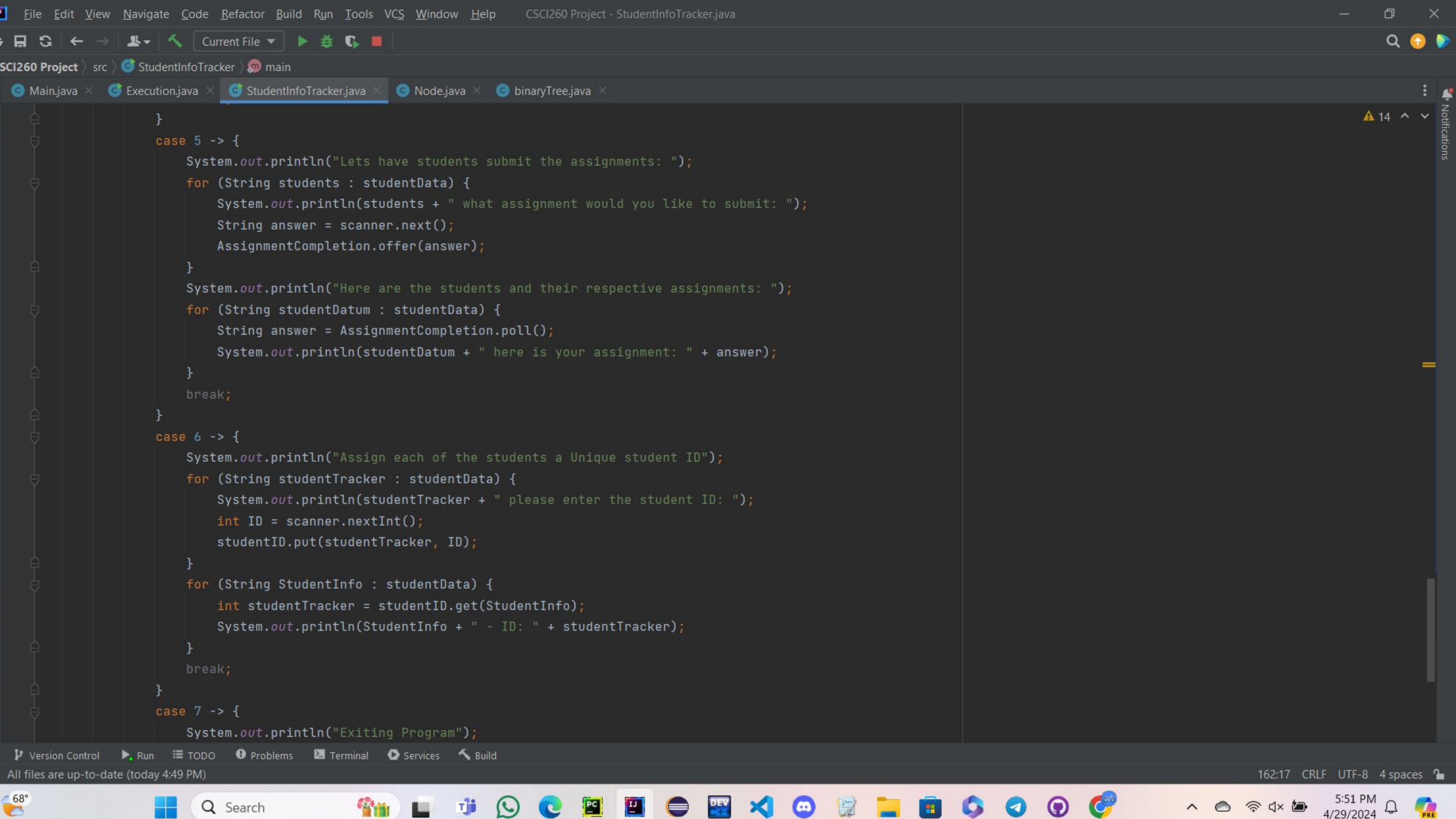


➔ Case 3 is used in order to replace student names user is asked to enter the index of the name that they would like to replace and then user is asked to enter a new name. The new name is inserted at the data in the student data linked list and finally the updated list is printed out

➔ Case 4 is used in order to store student grades. grades Array is initialized to store student grade. User is asked to enter grades for each student while iterating over each student in the student data linked list. The grades are then pushed into a stack called studentGrades. The student grades are then printed out. User is then asked if they want to sort grades from lowest to highest and if ‘y’ is entered grades are sorted from least to greatest using the bubble sort algorithm displayed on the next slide.

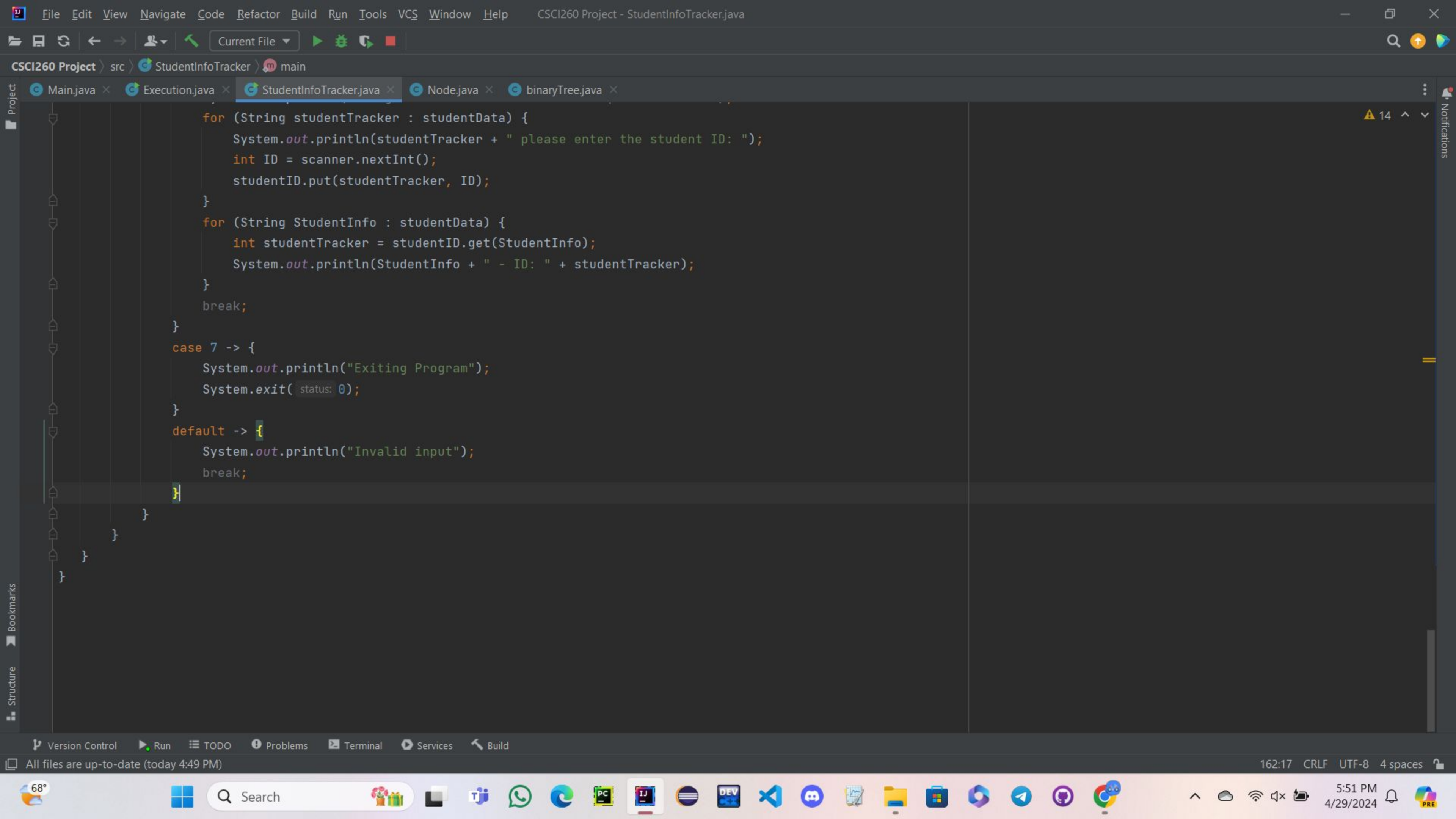

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - StudentInfoTracker.java
Current File
CSCI260 Project › src › StudentInfoTracker › main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
Version Control
Run
TODO
Problems
Terminal
Services
Build
All files are up-to-date (56 minutes ago)
129:72 CRLF UTF-8 4 spaces

```java
import java.util.*;


no usages
public class StudentInfoTracker {
    1 usage
    private static void bubbleSort(int[] intArray){
        int n = intArray.length;
        int temp = 0;
        for (int i = 0; i < n; i++) {
            for (int j = 1; j < (n - i); j++) {
                if (intArray[j - 1] > intArray[j]) {
                    temp = intArray[j - 1];
                    intArray[j - 1] = intArray[j];
                    intArray[j] = temp;
                }
            }
        }
    }
    no usages
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);




        System.out.println("Welcome to the Student Information Tracker!");
        System.out.println("How many students are in the class: ");
        int number = scanner.nextInt();
```



```
            }
            case 5 -> {
                System.out.println("Lets have students submit the assignments: ");
                for (String students : studentData) {
                    System.out.println(students + " what assignment would you like to submit: ");
                    String answer = scanner.next();
                    AssignmentCompletion.offer(answer);
                }
                System.out.println("Here are the students and their respective assignments: ");
                for (String studentDatum : studentData) {
                    String answer = AssignmentCompletion.poll();
                    System.out.println(studentDatum + " here is your assignment: " + answer);
                }
                break;
            }
            case 6 -> {
                System.out.println("Assign each of the students a Unique student ID");
                for (String studentTracker : studentData) {
                    System.out.println(studentTracker + " please enter the student ID: ");
                    int ID = scanner.nextInt();
                    studentID.put(studentTracker, ID);
                }
                for (String StudentInfo : studentData) {
                    int studentTracker = studentID.get(StudentInfo);
                    System.out.println(StudentInfo + " - ID: " + studentTracker);
                }
                break;
            }
            case 7 -> {
                System.out.println("Exiting Program");
```

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - StudentInfoTracker.java
Current File
SCI260 Project > src > StudentInfoTracker > main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
14
Version Control
Run
TODO
Problems
Terminal
Services
Build
All files are up-to-date (today 4:49 PM)
162:17 CRLF UTF-8 4 spaces
68°
Search
5:51 PM
4/29/2024


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - StudentInfoTracker.java
Current File
CSCI260 Project 〉 src 〉 StudentInfoTracker 〉 main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
Version Control
Run
TODO
Problems
Terminal
Services
Build

```java
                for (String studentTracker : studentData) {
                    System.out.println(studentTracker + " please enter the student ID: ");
                    int ID = scanner.nextInt();
                    studentID.put(studentTracker, ID);
                }
                for (String StudentInfo : studentData) {
                    int studentTracker = studentID.get(StudentInfo);
                    System.out.println(StudentInfo + " - ID: " + studentTracker);
                }
                break;
            }
            case 7 -> {
                System.out.println("Exiting Program");
                System.exit( status: 0);
            }
            default -> {
                System.out.println("Invalid input");
                break;
            }
        }
    }
  }
}
```

All files are up-to-date (today 4:49 PM)   162:17 CRLF UTF-8 4 spaces

68° Search 5:51 PM 4/29/2024

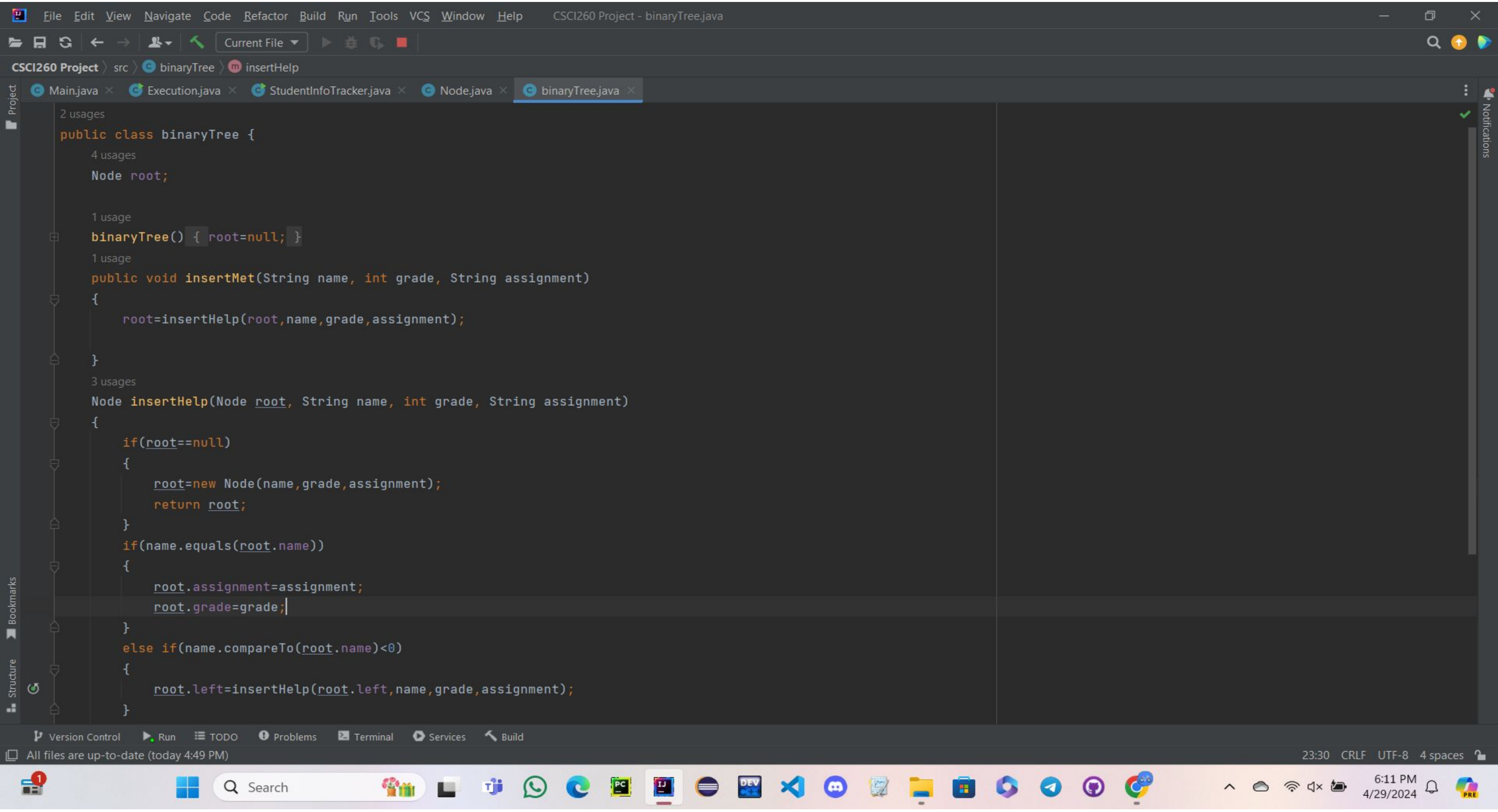
- Code explanation

➔ Case 5 allows user to submit student assignment using queues. student is asked what assignment they would like to submit. It does this by iterating over each student in the student data list. Assignments are stored in AssignmentCompletion queue After that it displays all the student names along with their assignment

➔ Case 6 asks for student ID numbers using a hashtable. Each student is asked for their ID. It iterates over each student in the student data list and asks them what the ID is. The ID is stored in studentID hashtable. It prints out at the end all student names along with their student ID

➔ Case 7 is the exit program, which simply prints a message indicating ther end of the program.

➔ Case 8 is default case which is is a user enters invalid input



File Edit View Navigate Code Refactor Build Run Tools VCS Window Help CSCI260 Project - binaryTree.java

CSCI260 Project 〉 src 〉 binaryTree 〉 insertHelp

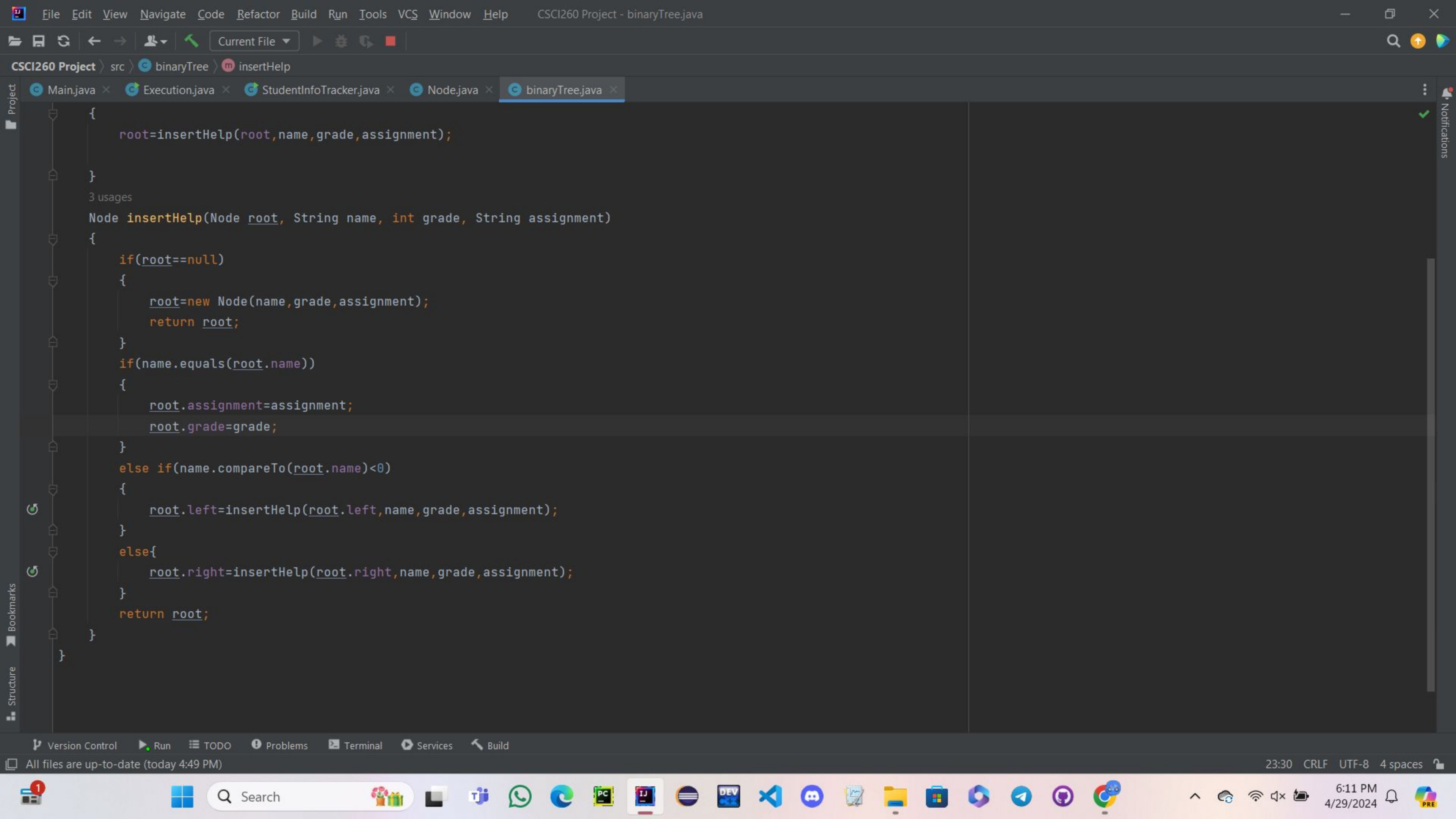
Main.java | Execution.java | StudentInfoTracker.java | Node.java | binaryTree.java

```
2 usages
public class binaryTree {
    4 usages
    Node root;

    1 usage
    binaryTree() { root=null; }
    1 usage
    public void insertMet(String name, int grade, String assignment)
    {
        root=insertHelp(root,name,grade,assignment);

    }
    3 usages
    Node insertHelp(Node root, String name, int grade, String assignment)
    {
        if(root==null)
        {
            root=new Node(name,grade,assignment);
            return root;
        }
        if(name.equals(root.name))
        {
            root.assignment=assignment;
            root.grade=grade;
        }
        else if(name.compareTo(root.name)<0)
        {
            root.left=insertHelp(root.left,name,grade,assignment);
        }
```

Version Control | Run | TODO | Problems | Terminal | Services | Build

All files are up-to-date (today 4:49 PM) 23:30 CRLF UTF-8 4 spaces

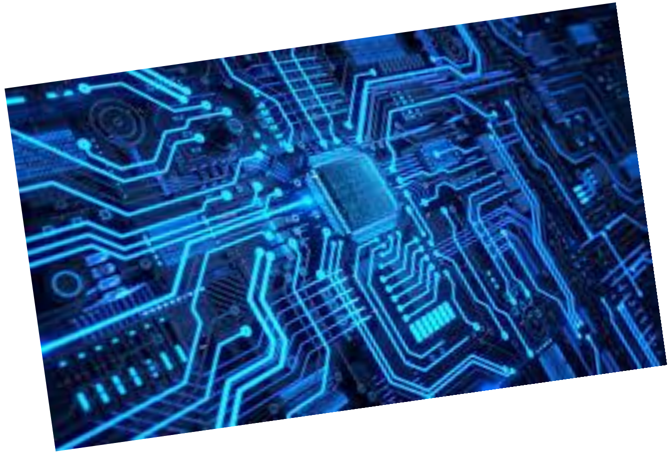
Search 6:11 PM 4/29/2024

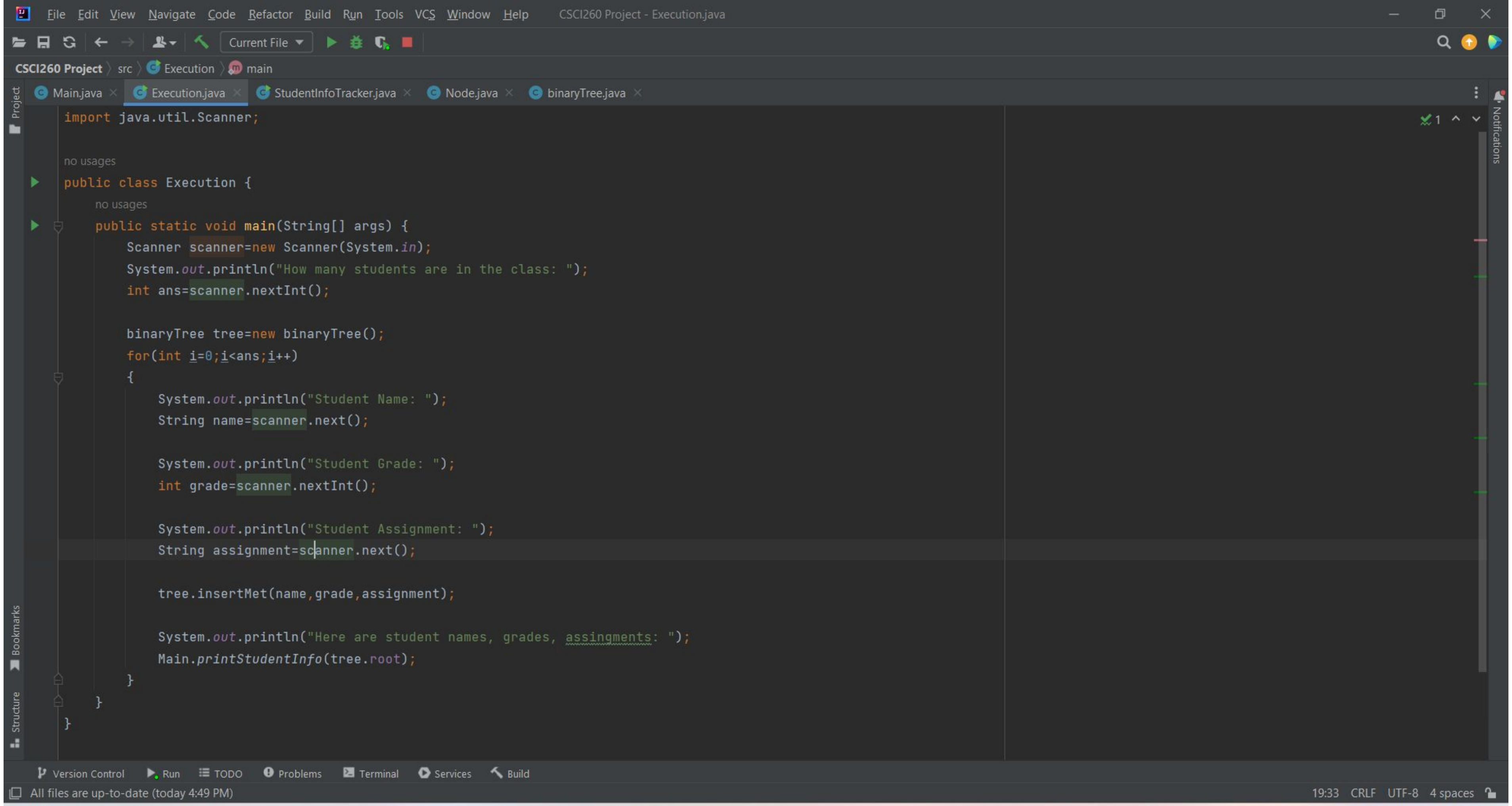

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - binaryTree.java
Current File
CSCI260 Project 〉 src 〉 binaryTree 〉 insertHelp
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
Version Control
Run
TODO
Problems
Terminal
Services
Build
All files are up-to-date (today 4:49 PM)
23:30 CRLF UTF-8 4 spaces
6:11 PM
4/29/2024

```
    {
        root=insertHelp(root,name,grade,assignment);

    }
    3 usages
    Node insertHelp(Node root, String name, int grade, String assignment)
    {
        if(root==null)
        {
            root=new Node(name,grade,assignment);
            return root;
        }
        if(name.equals(root.name))
        {
            root.assignment=assignment;
            root.grade=grade;
        }
        else if(name.compareTo(root.name)<0)
        {
            root.left=insertHelp(root.left,name,grade,assignment);
        }
        else{
            root.right=insertHelp(root.right,name,grade,assignment);
        }
        return root;
    }
}
```

- Code explanation

➔ we have a binary tree class and its function is to represent a binary tree; it contains essential methods that are useful for putting student information.




CSCI260 Project - Execution.java

CSCI260 Project 〉 src 〉 Execution 〉 main

Main.java | Execution.java | StudentInfoTracker.java | Node.java | binaryTree.java

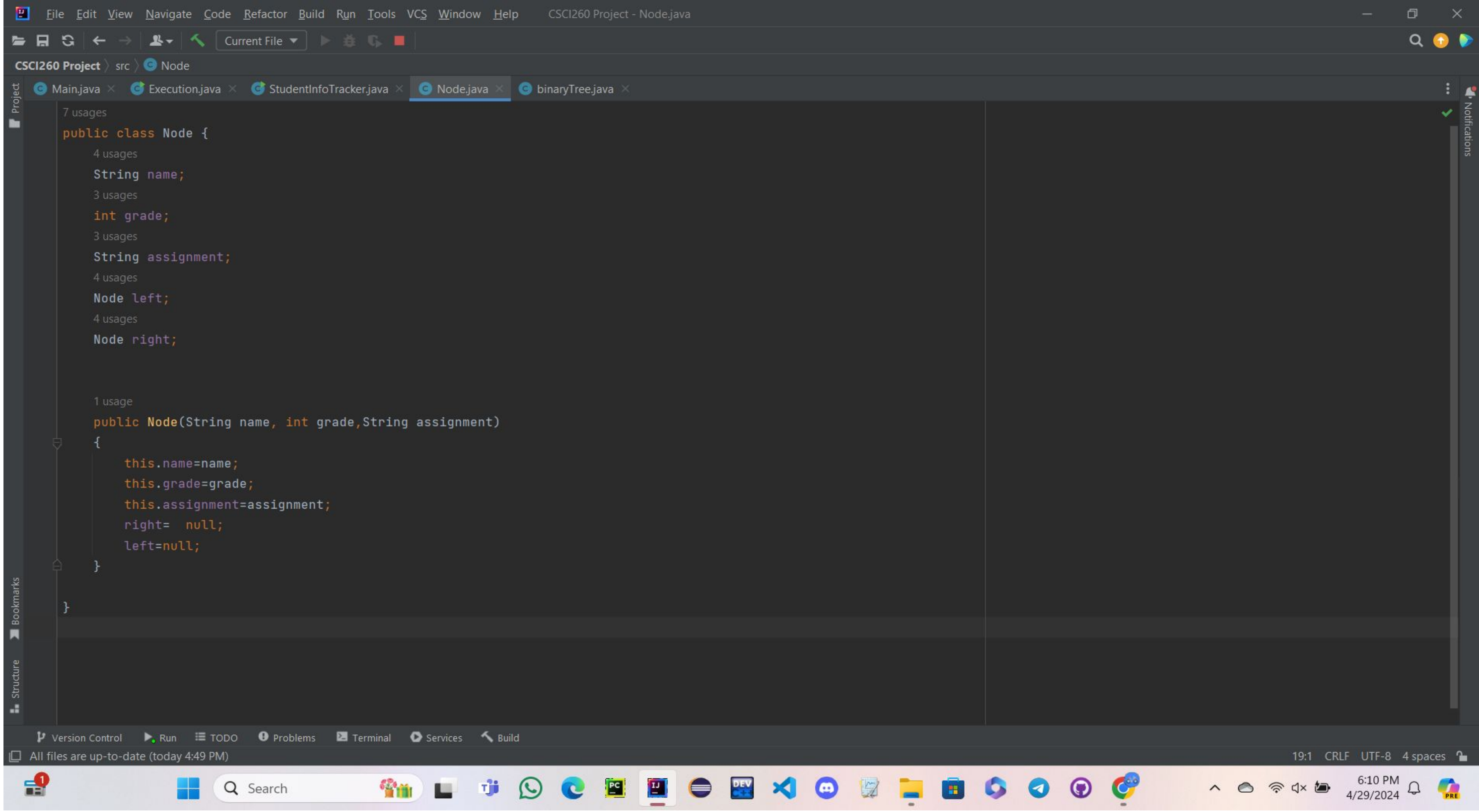
```java
import java.util.Scanner;

no usages
public class Execution {
    no usages
    public static void main(String[] args) {
        Scanner scanner=new Scanner(System.in);
        System.out.println("How many students are in the class: ");
        int ans=scanner.nextInt();

        binaryTree tree=new binaryTree();
        for(int i=0;i<ans;i++)
        {
            System.out.println("Student Name: ");
            String name=scanner.next();

            System.out.println("Student Grade: ");
            int grade=scanner.nextInt();

            System.out.println("Student Assignment: ");
            String assignment=scanner.next();

            tree.insertMet(name,grade,assignment);

            System.out.println("Here are student names, grades, assingments: ");
            Main.printStudentInfo(tree.root);
        }
    }
}
```

All files are up-to-date (today 4:49 PM)

19:33 CRLF UTF-8 4 spaces

- Code Explanation

➔ We also have the execution class where the program itself is executed. The user is asked to enter the number of students and it iterates over each student showcasing the essential information such as student names, grades, and assignments and after the information is entered printStudentInfo() is used in order to print out all the information about the student.



File Edit View Navigate Code Refactor Build Run Tools VCS Window Help CSCI260 Project - Node.java

CSCI260 Project 〉 src 〉 Node

Main.java × Execution.java × StudentInfoTracker.java × Node.java × binaryTree.java ×

```
7 usages
public class Node {
    4 usages
    String name;
    3 usages
    int grade;
    3 usages
    String assignment;
    4 usages
    Node left;
    4 usages
    Node right;


    1 usage
    public Node(String name, int grade,String assignment)
    {
        this.name=name;
        this.grade=grade;
        this.assignment=assignment;
        right=  null;
        left=null;
    }

}
```

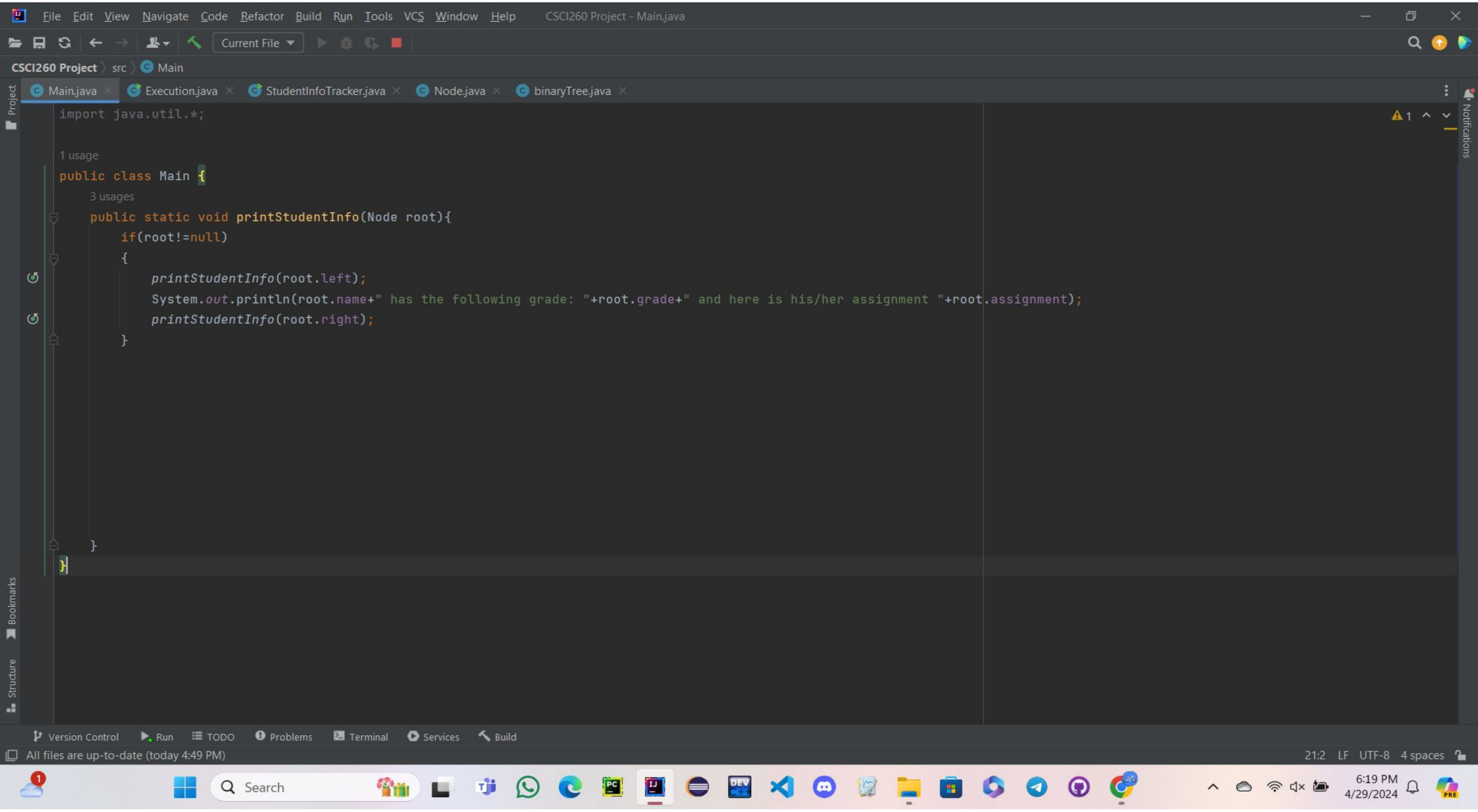
Version Control Run TODO Problems Terminal Services Build

All files are up-to-date (today 4:49 PM) 19:1 CRLF UTF-8 4 spaces

6:10 PM 4/29/2024

- Code Explanation

➔ we have a Node class and its job is to hold certain information about the student such as their name, grade, and assignments.



```java
import java.util.*;

public class Main {
    public static void printStudentInfo(Node root){
        if(root!=null)
        {
            printStudentInfo(root.left);
            System.out.println(root.name+" has the following grade: "+root.grade+" and here is his/her assignment "+root.assignment);
            printStudentInfo(root.right);
        }



    }
}
```

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
CSCI260 Project - Main.java
Current File
CSCI260 Project src Main
Main.java
Execution.java
StudentInfoTracker.java
Node.java
binaryTree.java
1 usage
3 usages
Version Control Run TODO Problems Terminal Services Build
All files are up-to-date (today 4:49 PM)
21:2 LF UTF-8 4 spaces
Search
6:19 PM
4/29/2024

- Code Explanation

→ If we look at the Main class it contains the static method printStudentInfo() that is used to print student information stored in the tree. Student information is displayed on the console and the information is organised on the basis of student name. root.left() and root.right() are used in order to maintain the structure for the binary tree