**­Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра автоматизованих систем обробки інформації**

**і управління**

**Звіт**

з лабораторної роботи № 5 з дисципліни

«Алгоритми та структури даних 2. Структури даних»

**«Деревовидні структури даних»**

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-01 Галько Міла Вячеславівна*

**Перевірив**

(прізвище, ім'я, по батькові)

*Головченко М.М.*

Київ 2021

**Лабораторна робота № 5**

**«Деревовидні структури даних»**

**Мета роботи** – вивчити основні підходи формалізації та імплементації алгоритмів побудови та обробки базових деревовидних структур даних.

**Завдання**

Розробити алгоритм (1) розв’язання задачі відповідно до варіанту №6. Виконати програмну реалізацію задачі (2). Не використовувати вбудовані деревовидні структури даних (контейнери). Зробити висновок по лабораторній роботі (3).

Варіант №6: Побудувати двійкове дерево пошуку з літер заданого рядка. Видалити з дерева літери, що зустрічаються більше одного разу. Вивести елементи дерева, що залишилися, при його постфіксному обході.

(1)

1. StringInput.
2. BinaryTree tree.
3. CharsToTree:
   1. for (int i = 0; i < str.size(); i++):

tree.Add(str[i]).

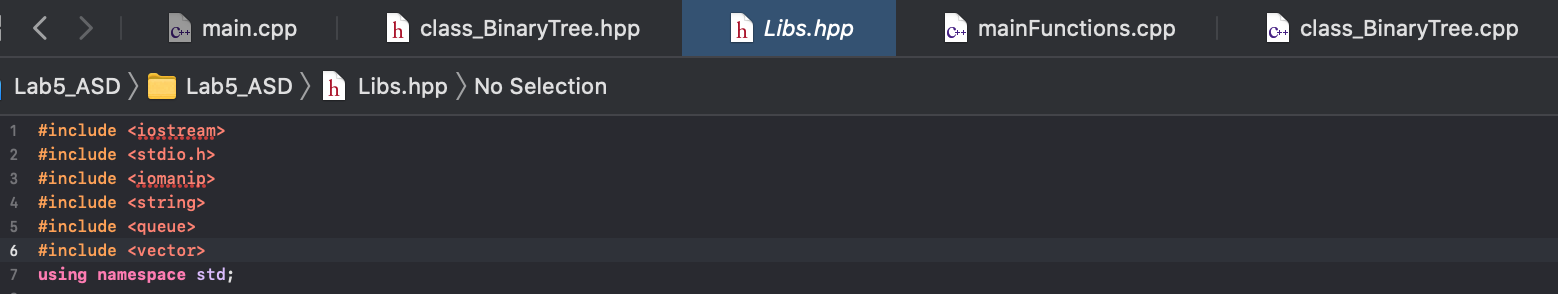
1. tree.OutputTree
2. tree.ReverseTreePassingOutput:
   1. if (root == nullptr): Tree is empty
   2. else:
      1. ReverseReading(root, que)
      2. int queSize = que.size()
      3. for (int i = 0; i < queSize - 1; i++):
         1. Output: que.front()
         2. que.pop()
3. tree.Check():
   1. if (root == nullptr): Tree is empty
   2. else:
      1. ReverseReading(root, que)
      2. int queSize = que.size()
      3. for (int i = 0; i < queSize; i++):
         1. array[i] = que.front()
         2. que.pop()
      4. vector<char> removingChars
      5. for (int i = 0; i < queSize; i++):
         1. for (int j = 1; j < queSize; j++) :
            1. if (array[i] == array[j] && !alreadyAdded(array[i], removingChars) && i != j):

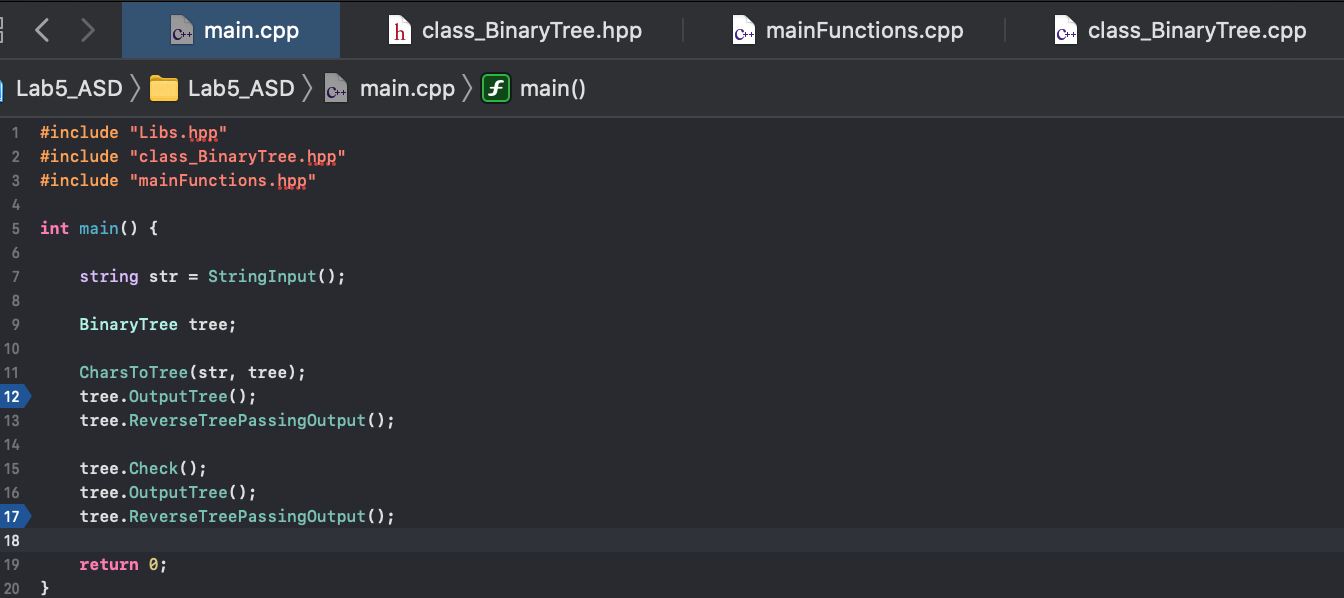
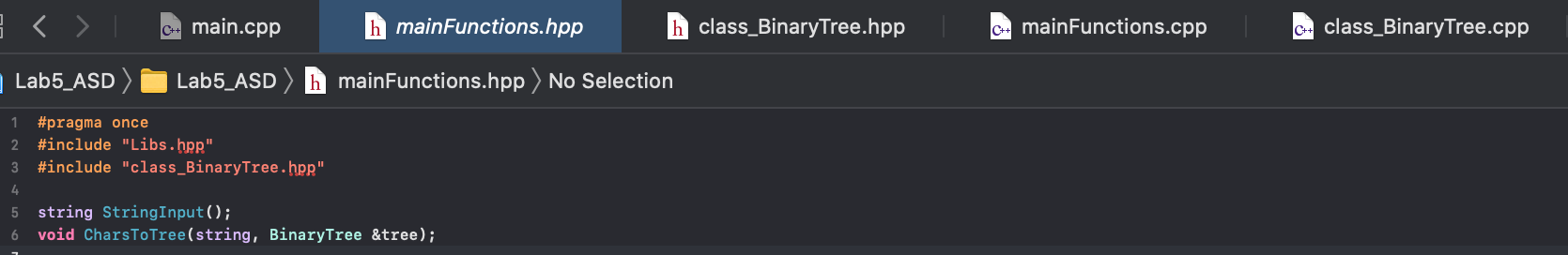
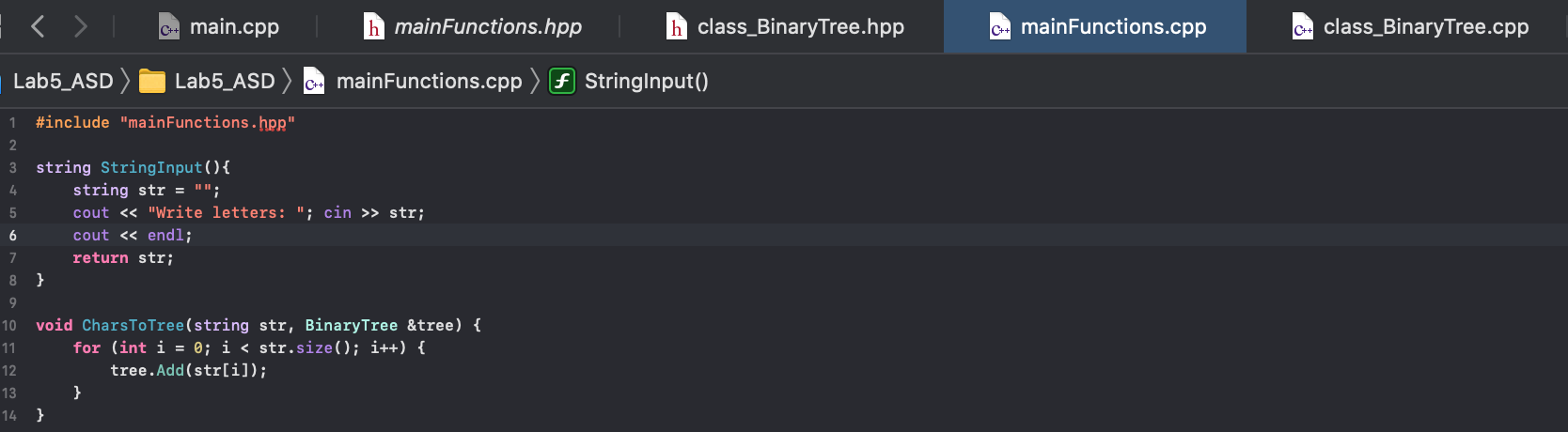
removingChars.push(array[i]).

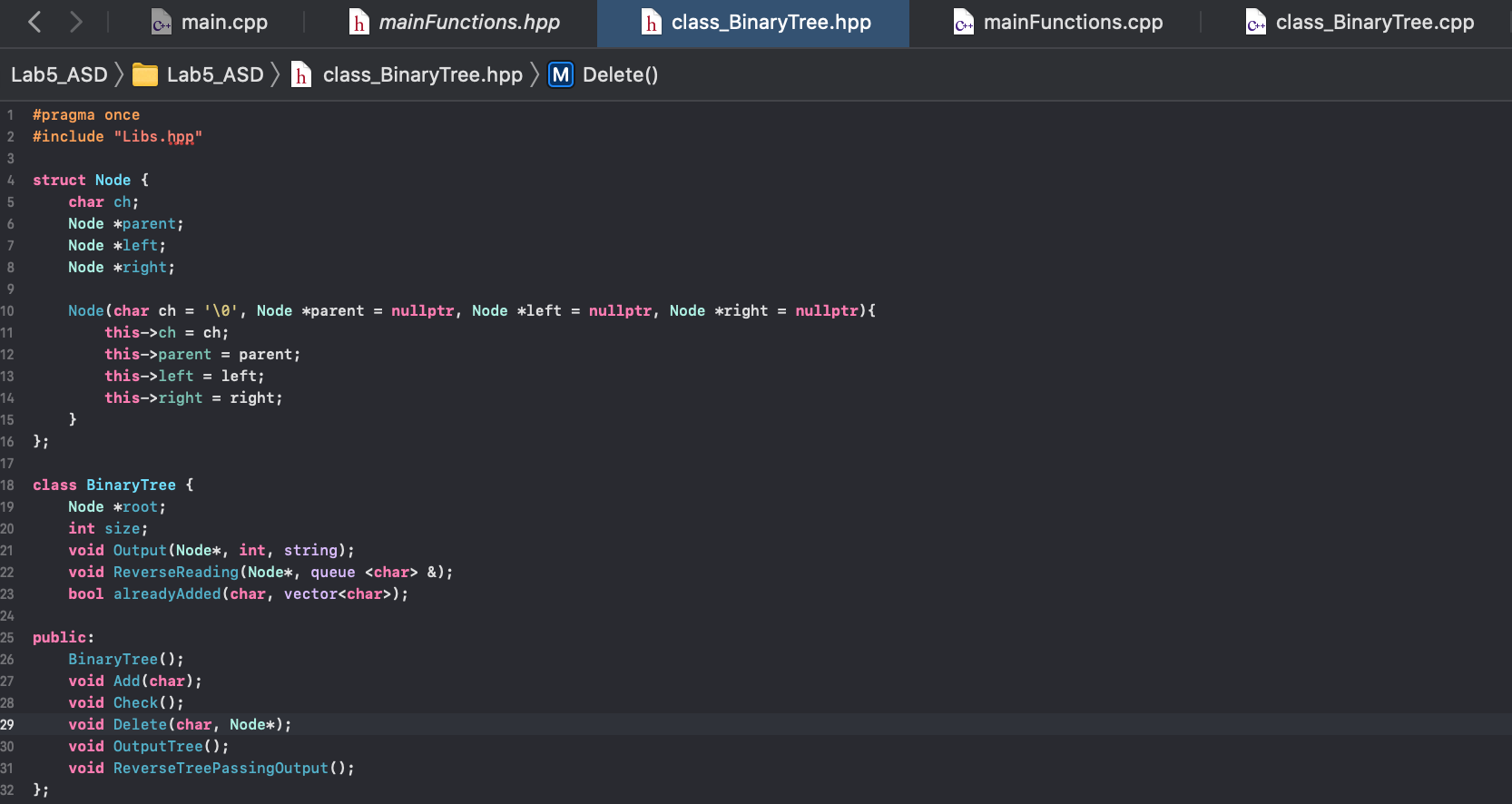
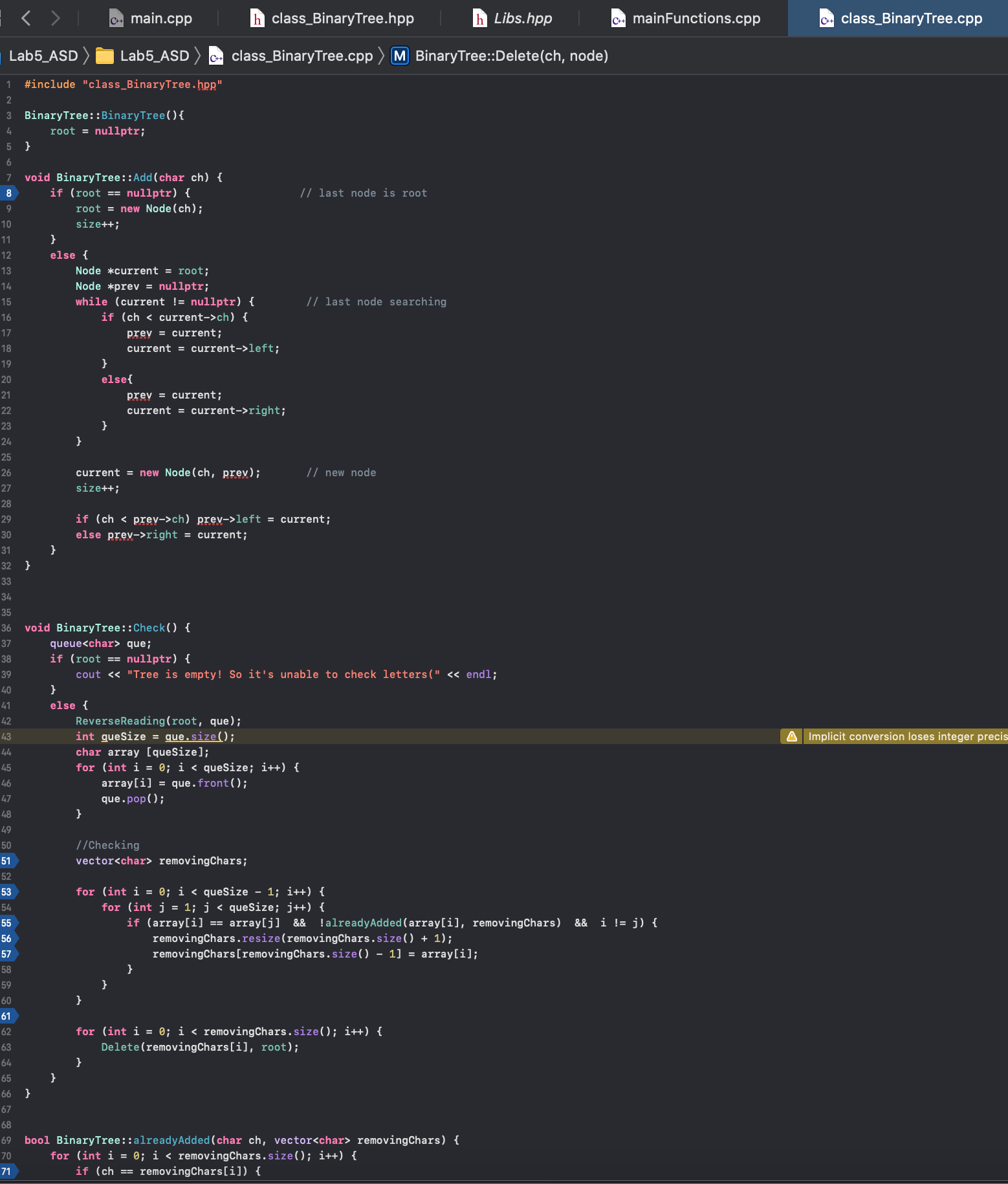
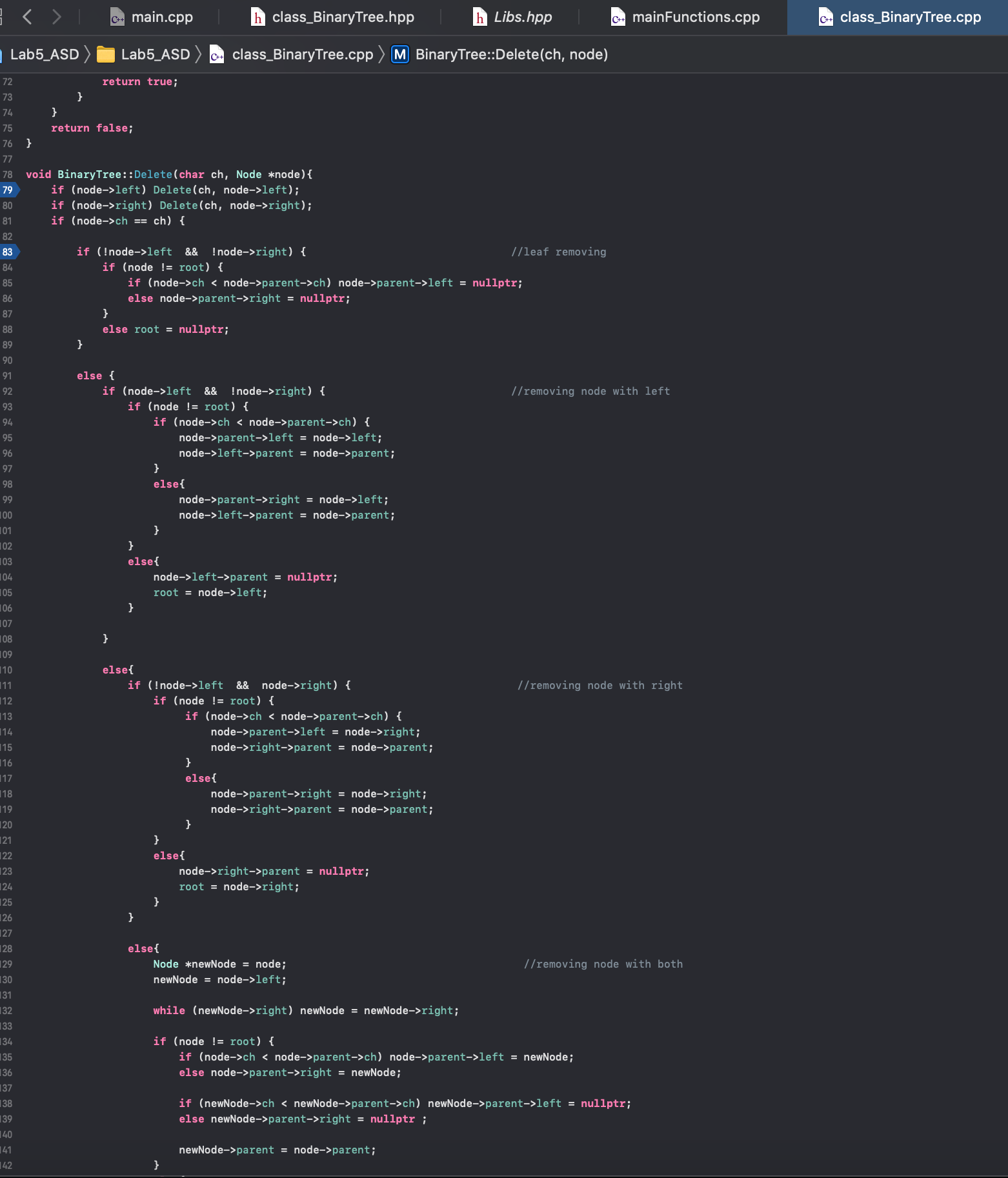
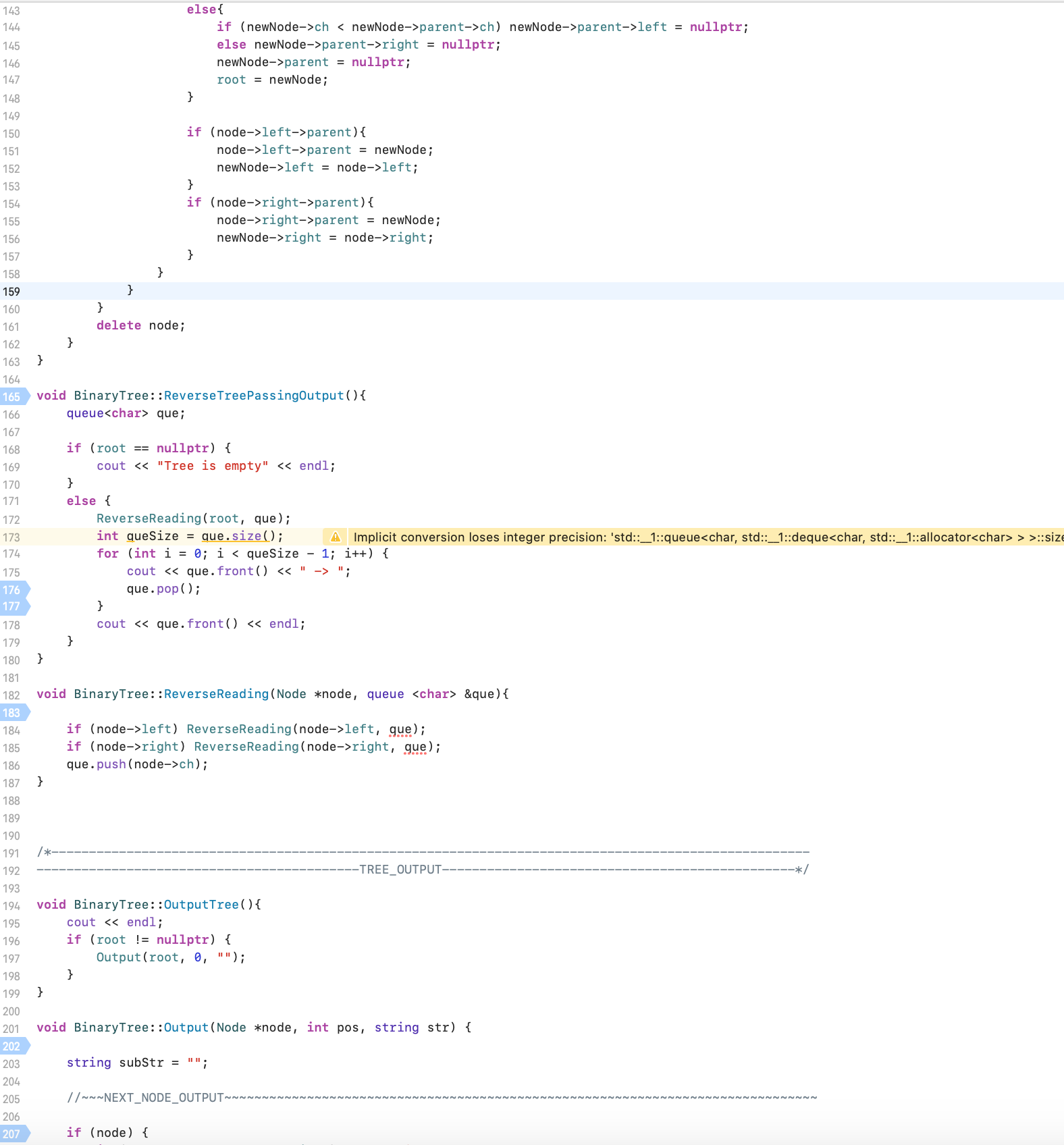
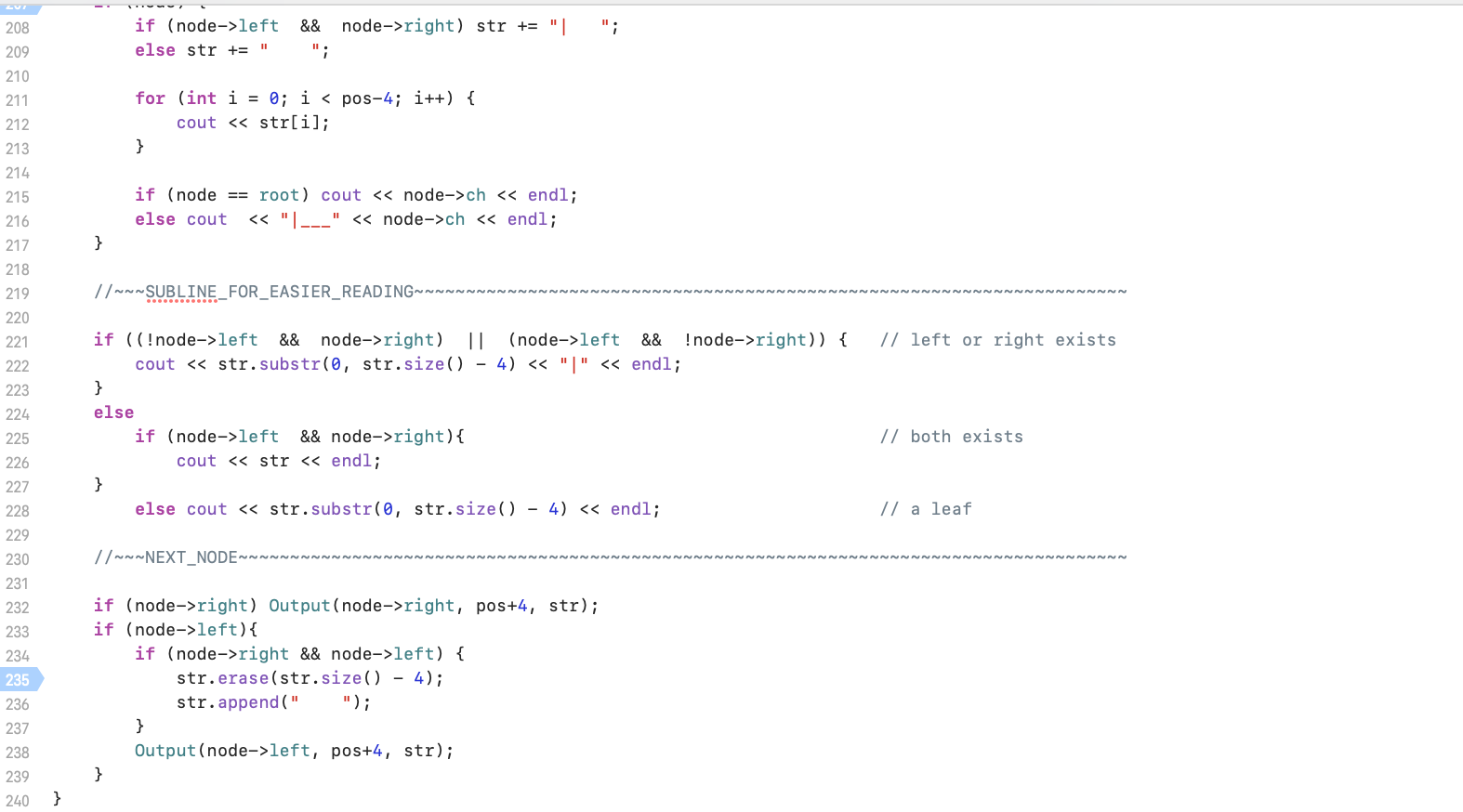
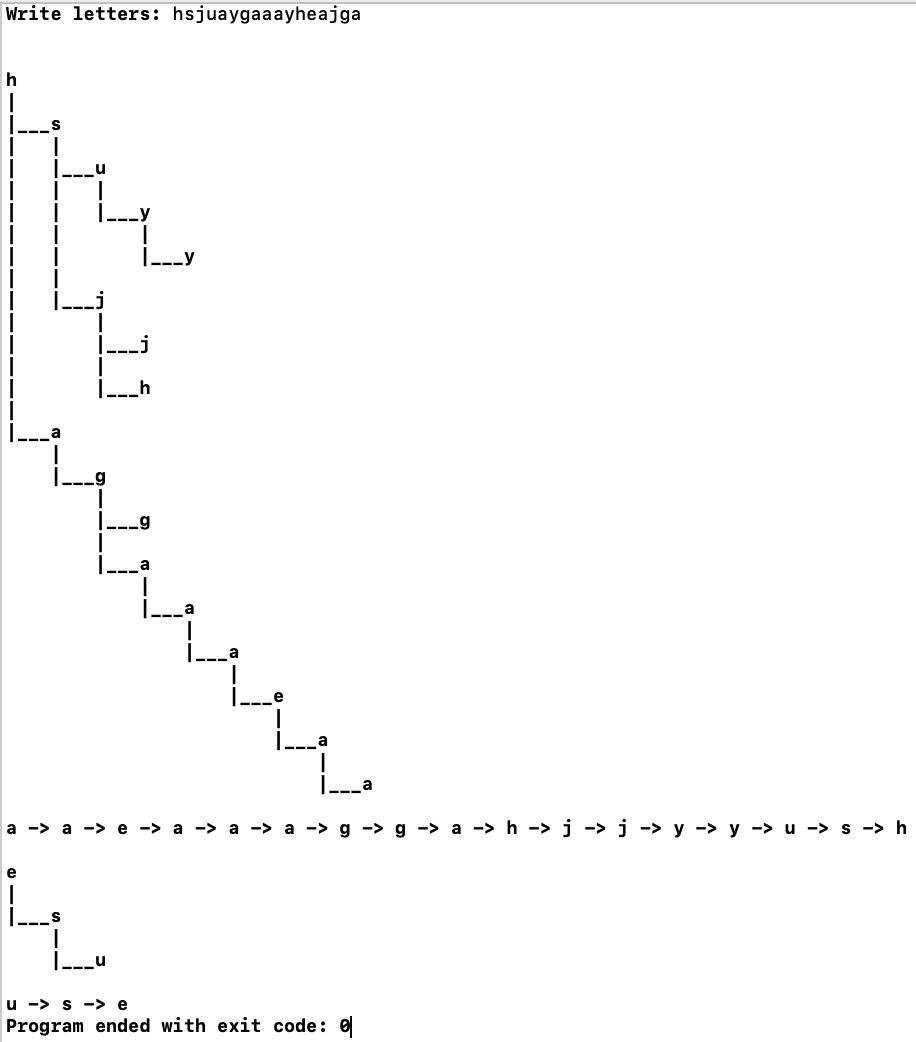
* + - 1. for (int i = 0; i < removingChars.size(); i++):
         1. Delete(removingChars[i], root);

1. tree.OutputTree()
2. tree.ReverseTreePassingOutput()

(2)



(3) Висновок:

В данній лабораторній роботі я теоритично засвоїла тему деревовидних структур, вивчила різні алгоритми їх побудови. Практично мала справу саме із бінарним деревом пошуку та з його постфіксним обходом.

В ході виконання задачі написала:

1. алгоритм виконяння завдання;
2. програмну реалізацію:
   1. бінарне дерево являє собо класс із структурою зв’язків
   2. обхід (постфіксний) здійснений через рекурсію.
   3. класс «бінарне дерево» має методи: додавання вузлів, видалення вузлів, вивід дерева та обхід дерева знизу-вверх.