1) What kind of classes or functions allow us to specific a type to use? Note: the type is not determined until we call the function.

i.e., we have a function which accepts 2 parameters, both types are what we specified.

ANSWER:

2) Look at the following code. Then, tell me (1) what makes this possible and (2) the type of z and c.

Segment 1:
```
int x = 0;
int y = 100;
auto z = x + y;
```

Segment 2:
```
Complex a = 0;
Complex b = 24;
auto c = a + b;
```

WHAT MAKES THIS POSSIBLE?
WHAT IS THE TYPE OF Z?
WHAT IS THE TYPE OF C?

3) What are public, private, protected, internal, file-private considered?

ANSWER:

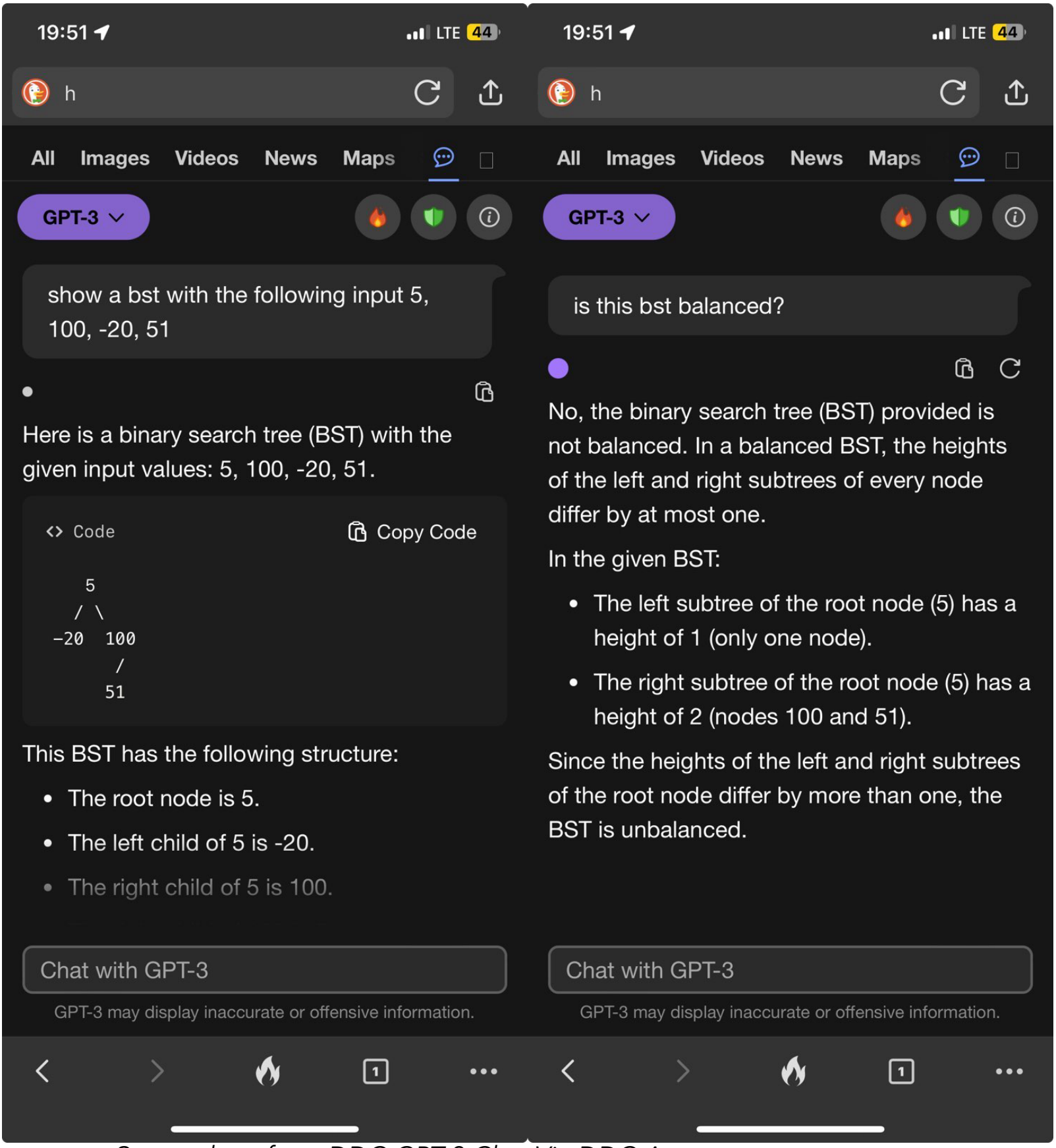
4) What is the difference between the statements below? Be specific, also include any functions that are called, explicitly or implicitly.

a) Complex& operator-(Complex lhs, Complex rhs)
b) Complex& operator-(Complex& lhs, Complex rhs)
c) Complex& operator-(Complex lhs, Complex &rhs)
d) Complex operator-(Complex &lhs, Complex rhs)

ANSWER:

5) Suppose we have a file containing 5, 100, -20, 51. We insert these into a BST. Do you agree with OpenAI's GPT-3 Model? (See images on next page)

ANSWER:



Screenshots from DDG GPT-3 Chat Via DDG App

6) Now, suppose we have a file containing 0 15 -25 5 20 -1 -26. Is it balanced? Draw a diagram if it helps.

ANSWER:

7) Evaluate the following computation.

10 4 * 5 / 2 4 5 + + +

ANSWER:

Know about:

1. **Information hiding**
2. **Typedef**
3. **Template**
4. **Function overloading**
5. **Copy constructors (when they are called)**
6. **Constructors**
7. **Destructors**
8. **How to pass streams**
9. **Abstraction**
10. **`this`**
11. **Stacks**
12. **Queues**
13. **BSTs**

Be able to:

1. **Write a copy constructor given a class**
2. **Write an output stream insertion operator**
3. **Write an input stream operator for an object**
    a. **i.e., cin >> Complex; // You need to write the overloaded op**
4. **Write both a member and non-member binary overloaded operator function**
    a. **i.e., CLASS & CLASS::operator= (CLASS & rhs) // this is lhs, implicitly**
    b. **i.e. CLASS & operator= (CLASS & lhs, CLASS & rhs)**