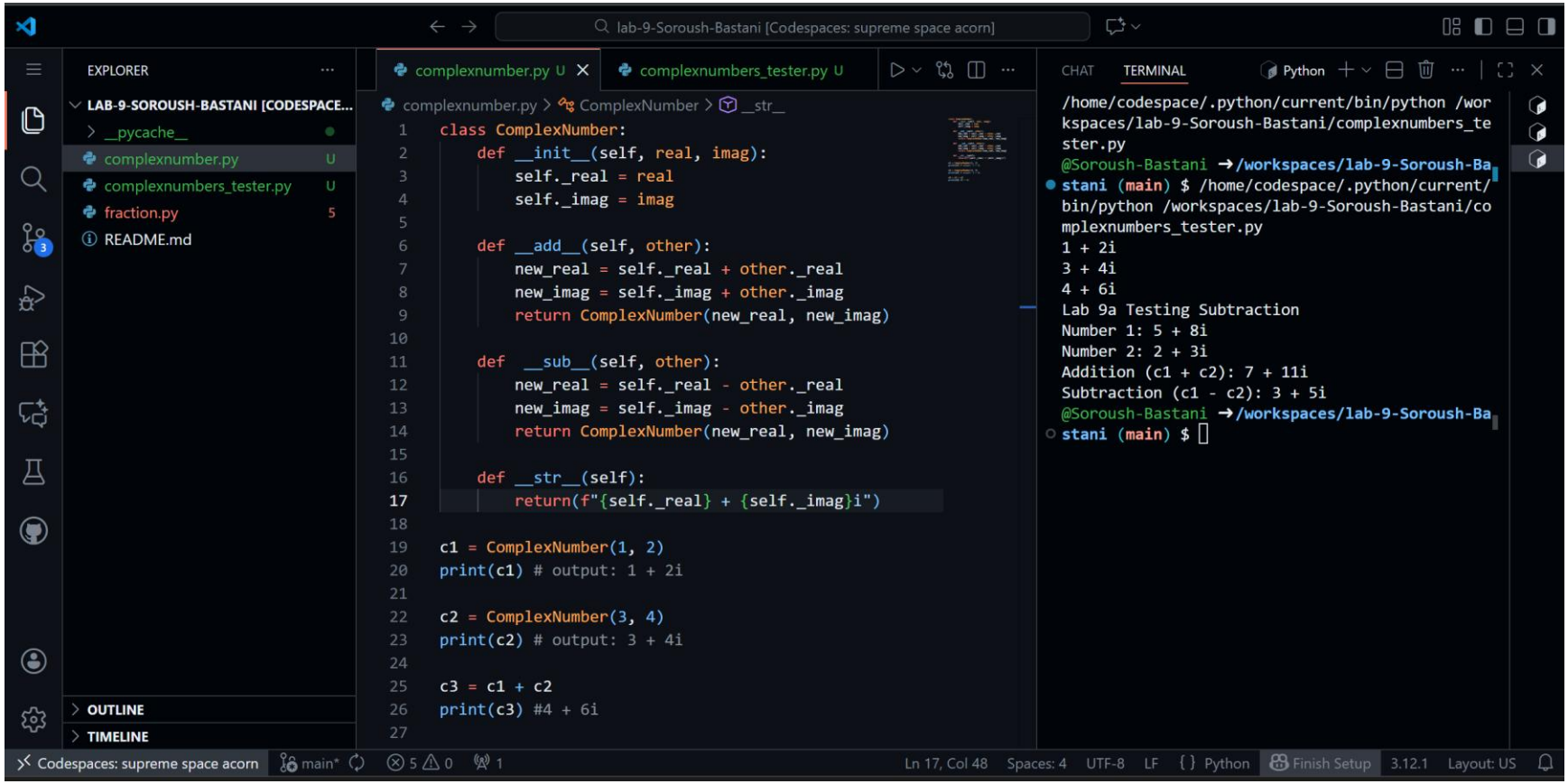
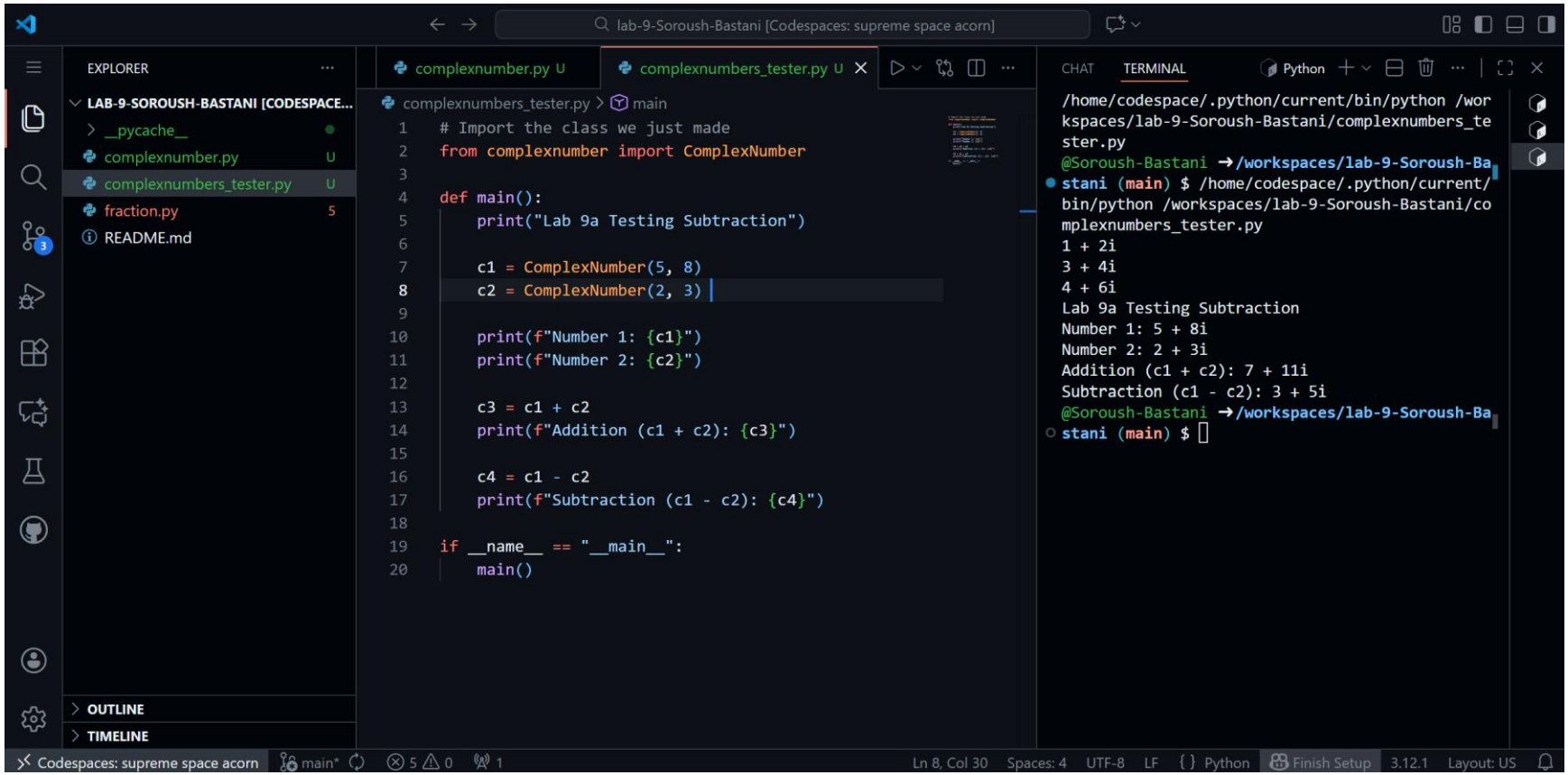
SOROUSH BASTANI
SBASTANI1

# Lab 9a

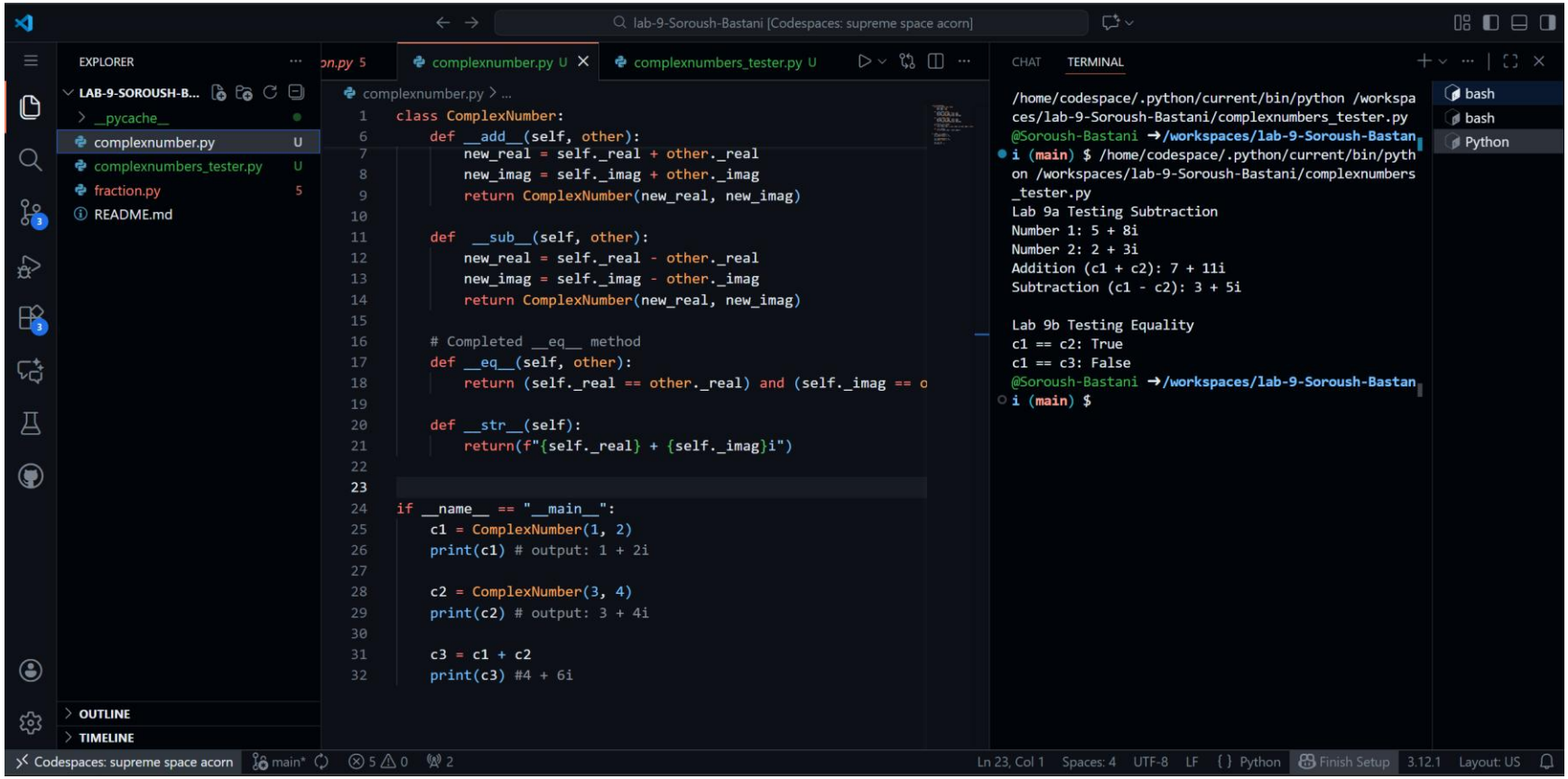


SOROUSH BASTANI
SBASTANI1

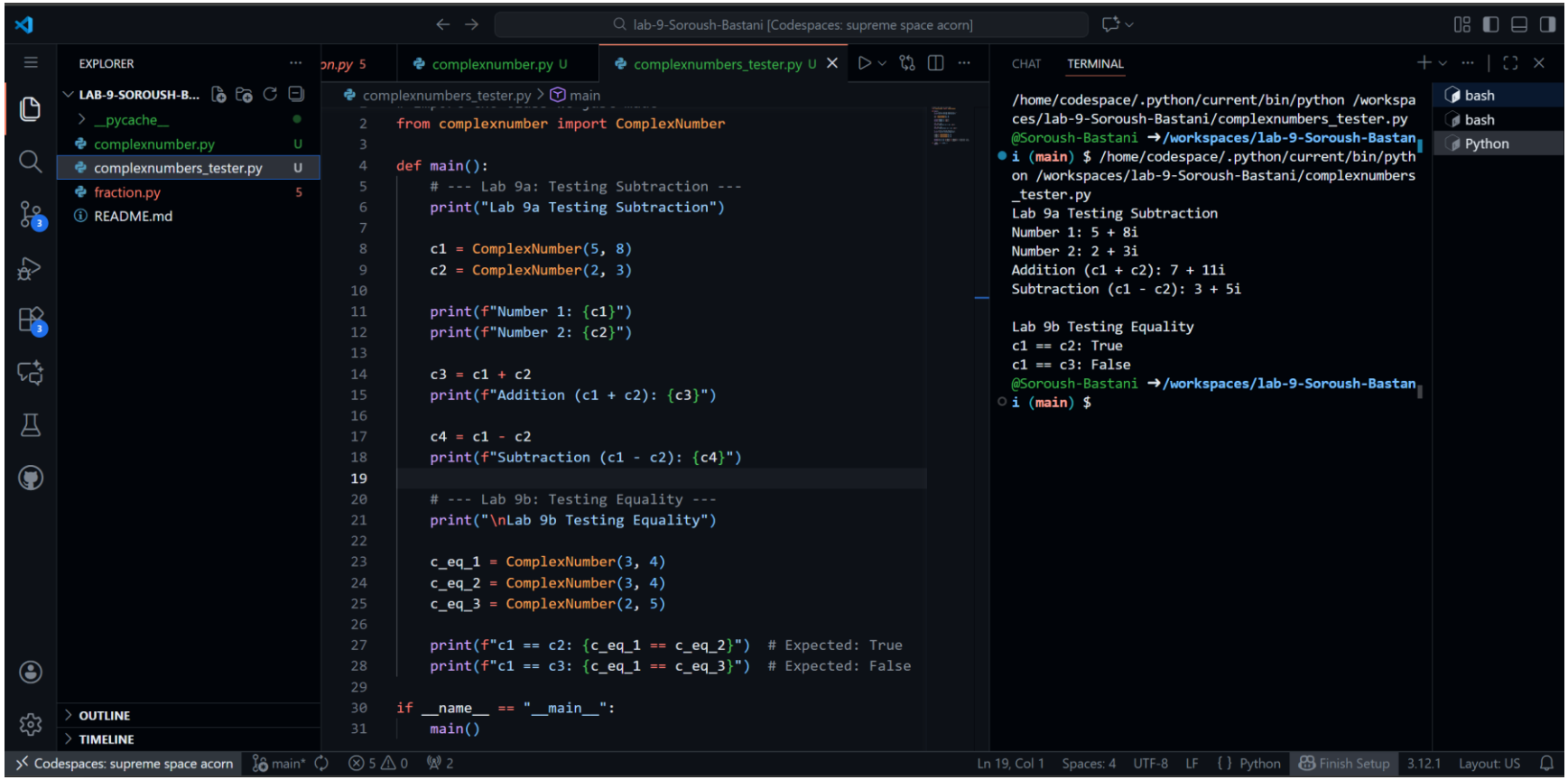


SOROUSH BASTANI
SBASTANI1

# Lab 9b

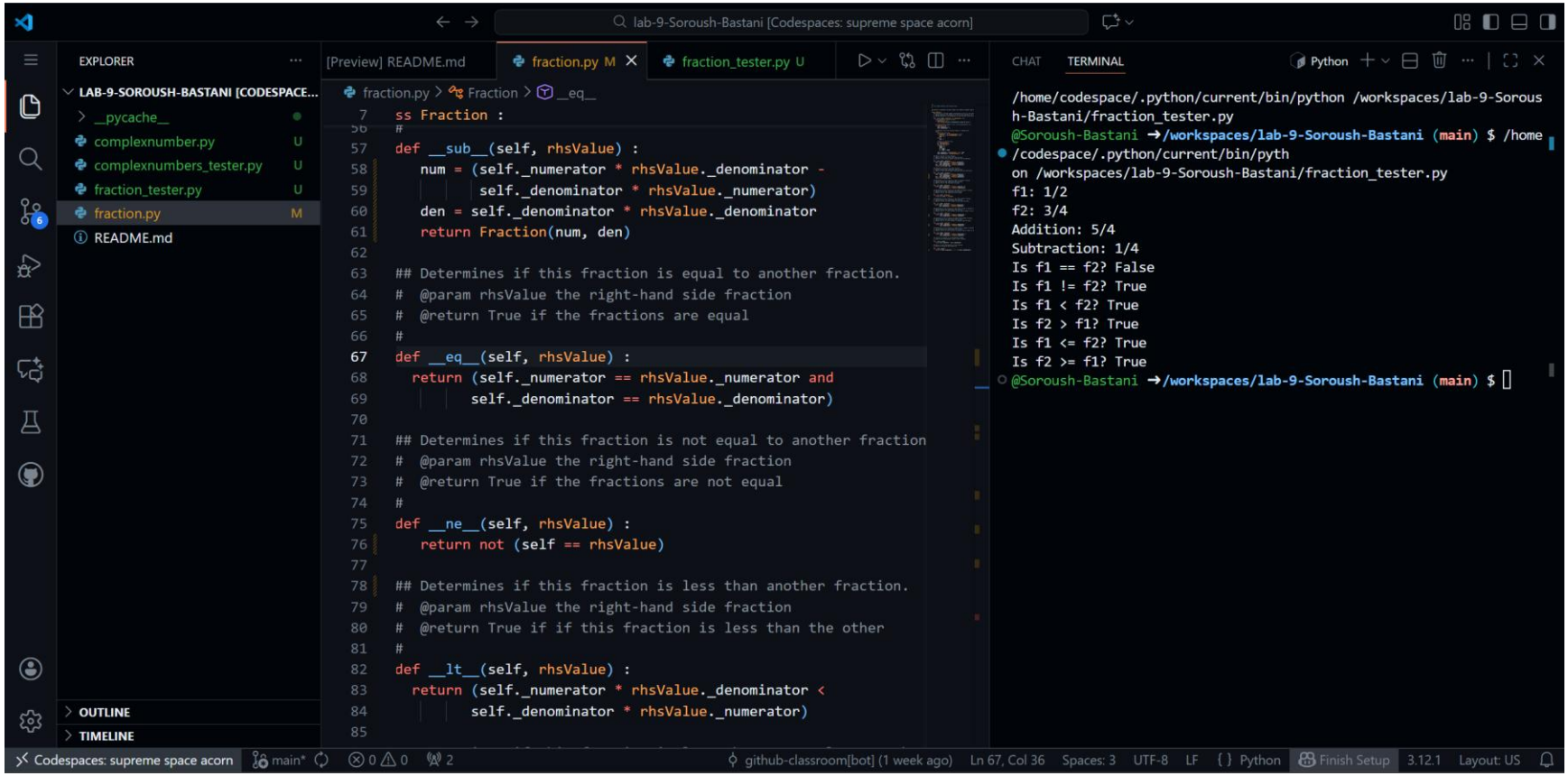


SOROUSH BASTANI
SBASTANI1

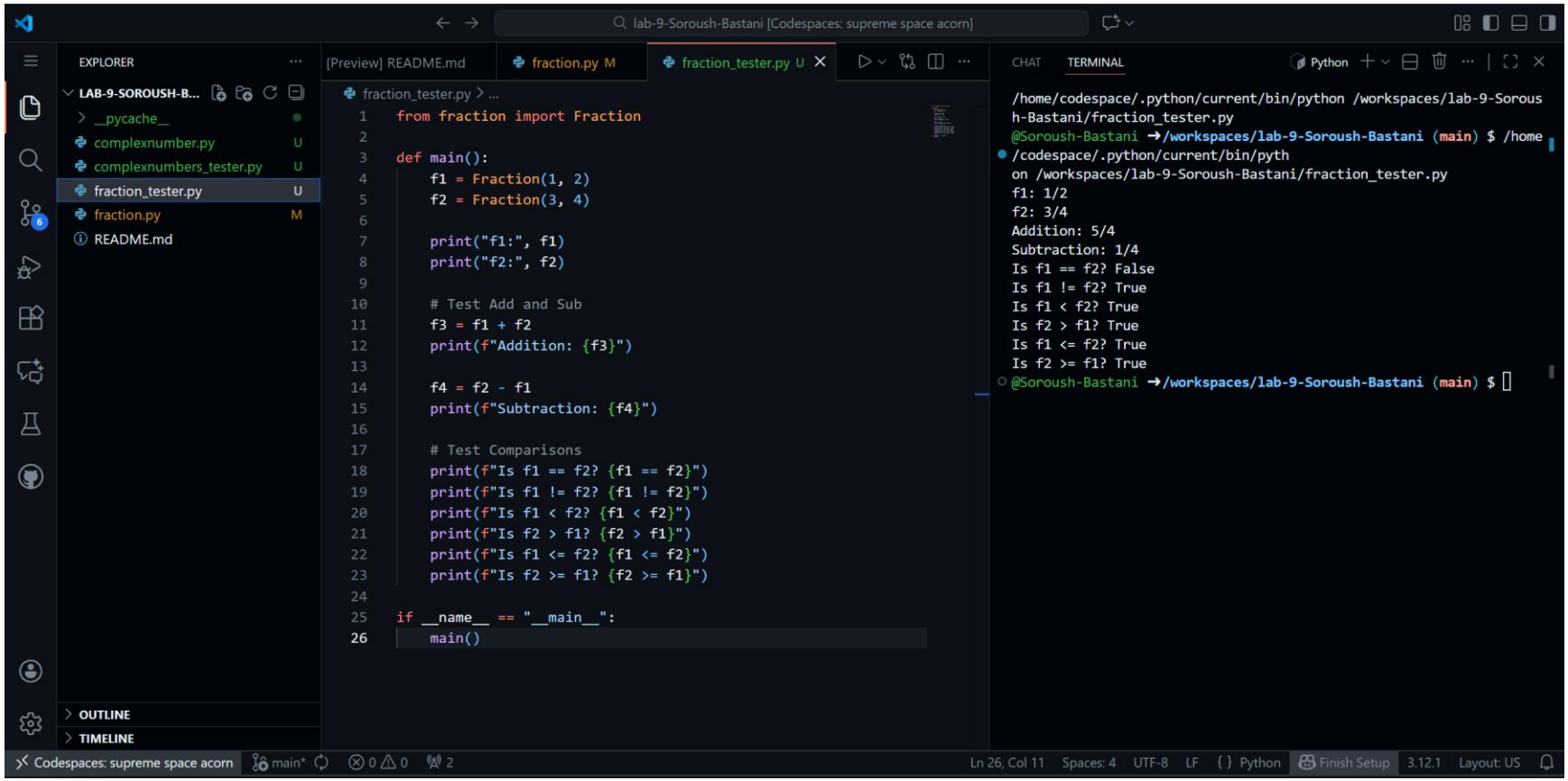


SOROUSH BASTANI
SBASTANI1

# Lab 9c



SOROUSH BASTANI
SBASTANI1



SOROUSH BASTANI
SBASTANI1

# Lab 9d

Working with the ComplexNumber and Fraction classes really clarified the difference between a class and an object for me. I realized that the class is essentially just a blueprint or a template, while the objects (like c1, c2, or f1) are the actual specific versions built from that blueprint. It was interesting to see how I could write the code once in the class, but create multiple objects that each held their own unique data without mixing them up.

The concept of operator overloading was definitely the coolest part of this lab. Before this, I thought + and - were fixed rules in Python. It feels powerful to know that I can define __add__ to make the plus sign do whatever I want, like adding complex fractions. It made the testing code much easier to read because writing f1 + f2 looks like a normal math equation, rather than having to call something clunky like f1.add(f2).

Finally, I understand why encapsulation and self._numerator are so important. If we just used global variables, every object would share the same data, which would be a mess.
Using self ensures that the data stays safe and locked inside that specific object, so changing one fraction doesn't accidentally break another one.