Homework 1 Reflection

For me, the closed box testing was more thorough. With not knowing what the code looks like for making tests it led to me making a lot of tests to see if I could break my code. So more like QA tests. For instance if the only thing I knew was I had to pass a list and an element to find I designed a lot of tests along the lines of “does it work with int lists, does it work with float lists, does it work with string lists”.

Also for closed box I pretty much did the equivalence partitioning along the only 2 outcomes, either you feed the function a valid input or you don’t. So the first section of tests were every scenario I could think of that would raise an error. The second set of tests were the valid inputs which I then split down further. Because when you have a valid input the binary search could either find the element you give it or it doesn’t find it. So Again in closed box I took a QA approach of giving it elements to find of every type, elements of size 1, elements where the answer would be right in the middle, to the left of the midpoint, and to the write of the input, and lists that needed to be sorted first before search.

When doing open box tests, my focus shifted on just reaching each line of code in some fashion. This led to way fewer tests to fit this requirement. For my code I needed 6 minimum tests to reach full code coverage: 4 to test each type of input that would raise an error, one test where the element was present and the element would first be to the right then to the left of the midpoint, and then a final test where the element was not in the list at all. While it may have been quicker and more efficient doing the white box testing, for this particular case, I would feel more confident in the closed box testing which really tried to find edge cases as opposed to just making sure each line will run properly at least once given a certain input.

The tests were done using pytests so you can access the blackbox testing by just specifying the file name ClosedboxTests.py in the command line. For code coverage breakdown use pytest cov. Bellow are screenshots how the coverage of these tests

