For the example below, will reordering the five async statements change the meaning of the program (assuming that the semantics of the reader/writer methods depends only on their parameters)? If so, show two orderings that exhibit different behaviors. If not, explain why not.

1. var left = new DataDrivenFuture();
2. var right = new DataDrivenFuture();
3. finish {
   4. asyncAwait(left) leftReader(left); // Task3
   5. asyncAwait(right) rightReader(right); // Task5
   6. asyncAwait(left, right)
   7. bothReader(left, right); // Task4
   8. async left.put(leftWriter()); // Task1
   9. async right.put(rightWriter()); // Task2
   10. }
For the example below, will reordering the five async statements change the meaning of the program (assuming that the semantics of the reader/writer methods depends only on their parameters)? If so, show two orderings that exhibit different behaviors. If not, explain why not.

No, reordering the asyncs doesn’t change the meaning of the program. Regardless of the order, Task 3 will always wait on Task 1. Task 5 will always wait on Task 2. Task 4 will always wait on both Task 1 and 2.

1. `var left = newDataDrivenFuture();`
2. `var right = newDataDrivenFuture();`
3. `finish {
   4.    `asyncAwait` left leftReader(left); // Task3
   5.    `asyncAwait` right rightReader(right); // Task5
   6.    `asyncAwait` (left,right)
   7.    bothReader(left,right); // Task4
   8.    `async` left.put(leftWriter()); // Task1
   9.    `async` right.put(rightWriter()); // Task2
   10. }`