Assumptions:

1. A general assumption I made is that the after sequential processing, the inputs left as previous input with the delay modules should be processed. This is what ‘ProcessRemainingDelayModules()’ function does. This is in accordance with the example given in the sheet.

DELAY

REVERSE

INPUT: hello world

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | REVERSE | DELAY | OUTPUT |
| hello | olleh | hello | hello |
| world | dlrow | olleh | olleh |
|  |  | dlrow | dlrow |

1. When there are multiple input modules, each input is given to each of these modules. They are processed in the order of creation.

Design(Check out headers for function purposes):

1. The use of shared\_pointers was due to the shared ownership because there can be multiple modules pointing to a single module in the network.
2. I am using Factory design pattern for the creation of modules and as expected they return unique pointers so that the callee gets full ownership.
3. FindOuputs and FindDelayedModules are done using recursion. If there are constraints on the input words, maybe dynamic programming can be used.
4. In reality, I would have considered a few more cases for unit test.