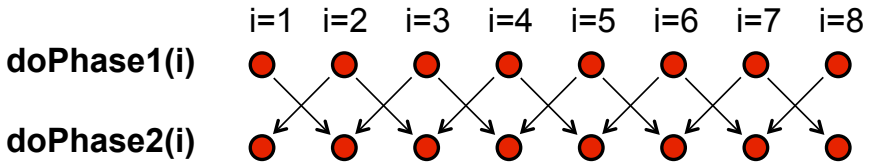
Worksheet #16a: Left-Right Neighbor Synchronization using Phasers (turn the page over for the Worksheet #16b)

Name: _____

Netid: _____



Complete the phased clause below to implement the point-to-point synchronization shown above for m tasks (generalization of slide 9)

```
1. finish (() -> {
2.   final HjPhaser[] ph =
         new HjPhaser[m+2]; // array of phaser objects
3.   forseq(0, m+1, (i) -> { ph[i] = newPhaser(SIG_WAIT) });
4.   forseq(1, m, (i) -> {
5.     asyncPhased(
          ph[i-1].inMode(......),
          ph[i].inMode(......),
          ph[i+1].inMode(......), ()->{
6.      doPhase1(i);
7.      next();
8.      doPhase2(i); }); // asyncPhased
9.   }); // forseq
10.}); // finish
```


Worksheet #16b: Reordered Asyncs with One Phaser (turn the page over for Worksheet #16a)

Name: _____ Netid: _____

Task A4 has been moved up to line 6. Does this change the computation graph in slide 9? If so, draw the new computation graph. If not, explain why the computation graph is the same.

```
1. finish (() -> {
2.     ph = newPhaser(SIG_WAIT); // mode is SIG_WAIT
3.     asyncPhased(ph.inMode(SIG), () -> {
4.       // A1 (SIG mode)
5.       doA1Phase1(); next(); doA1Phase2(); });
6.     asyncPhased(ph.inMode(HjPhaserMode.WAIT), () -> {
7.      // A4 (WAIT mode)
8.      doA4Phase1(); next(); doA4Phase2(); });
9.     asyncPhased(ph.inMode(SIG_WAIT), () -> {
10.       // A2 (SIG_WAIT mode)
11.       doA2Phase1(); next(); doA2Phase2(); });
12.    asyncPhased(ph.inMode(HjPhaserMode.SIG_WAIT), () -> {
13.     // A3 (SIG_WAIT mode)
14.     doA3Phase1(); next(); doA3Phase2(); });
15.    });
```