Perform forward list scheduling for the following ILOC code:

```
loadI 1024 => r0
loadI 0 => r1
storeAI r1 => r0, 0
loadI 63 => r3
storeAI r3 => r0, 4
loadI 5 => r5
loadAI r0, 0 => r6
add r5, r6 => r7
storeAI r7 => r0, 8
loadAI r0, 8 => r3
loadI 9 => r10
sub r3, r10 => r11
storeAI r11 => r0, 12
loadAI r0, 4 => r13
loadI 3 => r14
mult r13, r14 => r15
storeAI r15 => r0, 16
loadAI r0, 16 => r3
loadI 7 => r18
mult r3, r18 => r4
storeAI r4 => r0, 20
loadAI r0, 12 => r21
loadAI r0, 20 => r22
add r21, r22 => r23
storeAI r23 => r0, 24
loadAI r0, 24 => r25
storeAI r25 => r0, 28
output 1052
```

There are many possible variants of the basic forward list scheduling algorithm.
1. Show the dependence graph for the basic block. Show all true and anti-dependences needed to ensure the correct order of execution. You may omit dependences that are “covered” by other dependences in the graph.

2. Label the nodes in the dependence graph based on the longest latency-weighted path (see our class notes). Use the latencies as we discussed in class. Show the result of forward list scheduling using the longest latency-weighted path heuristic.

3. Instead of the longest latency-weighted path, use a selection heuristic that prefers nodes with the highest latency. Use the latencies as we discussed in class. Show the result of this scheduling heuristic.

4. Come up with your own heuristic for selecting nodes (instructions) in the forward list scheduling algorithm. Show the resulting reordered instructions.