Use Linked list for each IF, ID, EX, MEM, WB

* During the IF stage, each instruction inserts itself into each of the linked list of each function.
* Exceptions would apply in the EX list where EX of MUL.S, ADD.S, and register functions can execute in parallel. It is possible that we would have separate EX linked list for MUL.S, ADD.S and normal register functions.
* When it jumps, the new instruction also inserts itself into each linked list.
* Use array position index as tracker in each linked list.

Use locks in register as stall – solves the data dependency

* During the ID stage, the instruction should lock its RD. Release it on the next cycle after its WB.
* Also during the ID stage, the instruction checks if its RS and/RT are locked. If it is locked, it will stall.

Stalling dependency – stall instructions below if the current instruction is stalling.

* Since it is a linked list, it should naturally stall the instructions for a specific function (ex. ID of Ins 1 will always need to finish first before Ins 2 can execute)
* Have ifFinished, idFinished and others inside each instruction so it can track which functions have already finished. Only if ID of instruction 1 has finished before EX of instruction 1 can proceed.

Exclude R0 and F0 as causes of data dependency.