Exercises for Week 7
Textbook Sections 4.5 and 4.6

Question 1

Based on Textbook Exercise 4.16 <§4.5>.

In this Exercise, we examine how pipelining affect the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

Assume also that instructions executed by the processor are broken down as follows:

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

a. What is the clock cycle time in a pipelined and non-pipelined processor?

b. What is the total latency of an `ld` instruction in a pipelined and non-pipelined processor?

c. If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

d. Assuming there are no stalls or hazards, what is the utilization (% of cycles it is used) of the data memory?

e. Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

f. Instead of a single-cycle organization, we can use a multi-cycle organization where each instruction takes multiple cycles but one instruction finishes before another is fetched. In this organization, an instruction only goes through stages it actually needs (e.g. `sd` only takes 4 cycles because it does not need the WB stage). Compare clock cycle times and execution times of single-cycle and multi-cycle organizations with a pipelined organization.

Question 2

Textbook Exercises 4.19 <§4.5>.

Assume that x11 is initialized to 11 and x12 is initialized to 22. Suppose you executed the code below on a version of the pipeline from Section 4.5 that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting NOP instructions where necessary). What would the final value of register x15 be? Assume the register file is written at the beginning of the cycle and read at the end of a cycle. Therefore, an ID stage will return the results of a WB state occurring during the same cycle. See Section 4.7 and Figure 4.51 for details.

```
addi  x11, x12, 5
add   x13, x11, x12
addi  x14, x11, 15
add   x15, x11, x11
```

Question 3

Textbook Exercises 4.20 <§4.5>.

Add NOP instructions to the code below so that it will run correctly on a pipeline that does not handle data hazards.

```
addi  x11, x12, 5
add   x13, x11, x12
addi  x14, x11, 15
add   x15, x13, x12
```

Question 4

Textbook Exercises 4.22 <§4.5>.

Consider the fragment of RISC-V assembly below:

```
sd    x29, 12(x16)
ld    x29, 8(x16)
sub   x17, 12(x16)
beqz  x17, label
add   x15, x11, x14
sub   x15, x30, x14
```

Suppose we modify the pipeline so that it has only one memory (that handles both instructions and data). In this case, there will be a structural hazard every time a program needs to fetch an instruction during the same cycle in which another instruction accesses data.

a. Draw a pipeline diagram to show were the code above will stall.

b. In general, is it possible to reduce the number of stalls/NOPs resulting from this structural hazard by reordering code?

c. Must this structural hazard be handled in hardware? We have seen that data hazards can be eliminated by adding NOPs to the code. Can you do the same with this structural hazard? If so, explain how. If not, explain why not.

d. Approximately how many stalls would you expect this structural hazard to generate in a typical program? (Use the instruction mix from Exercise 4.8.)