You may work in pairs for this assignment. If you choose to work with a partner, make sure only one of you makes a submission a solution and that the file lists names and PIDs for both of you as described in the assignment below.

Prepare your answers to the following questions in a plain text file. Submit your file to the Curator system by the posted deadline for this assignment. No late submissions will be accepted. For all questions, show supporting work if you want partial credit.

You will submit your answers to the Curator System (www.cs.vt.edu/curator) under the heading MIPS04.

For questions 1 and 2, refer to the pipeline design with forwarding, shown below, which supports execution of the following MIPS instructions: add, sub, and, or, slt, and sw, and lw so long as no stalls are needed to resolve load-use hazards.

Remember: this pipeline design does not include hazard detection, so it can forward operands but it cannot introduce stalls to deal with situations that forwarding alone will not handle.
1. Assume we have the 5-stage pipelined datapath described above, with forwarding hardware but without hazard-detection hardware. Consider executing the following sequence of instructions on this datapath:

```
add $t5, $t3, $t2   # 1.1
lw  $t3, 4($t5)     # 1.2
add $t4, $t3, $t4   # 1.3
sw  $t3, 0($t4)     # 1.4
sub $t1, $t4, $t3   # 1.5
lw  $t4, 0($t1)     # 1.6
```

a) [12 points] Identify all of the data hazards that can be resolved with forwarding alone; that is, data hazards that do not require inserting a stall (nop).

For each such data hazard, state the number of the leading instruction, the number of the following instruction, the register involved, and indicate whether the forwarding will be from the EX/MEM interstage buffer or from the MEM/WB interstage buffer.

b) [8 points] Even with forwarding, the given sequence of instructions cannot execute correctly, in the order given without inserting at least one nop instruction. Show how to achieve correct execution by inserting the minimal number of nop instructions, but not reordering the instructions.

2. Assume we have the 5-stage pipelined datapath described above, with forwarding hardware but without hazard-detection hardware. Consider executing the following sequence of instructions on this datapath:

```
lw  $t3, 4($t4)     # 2.1
add $t2, $t1, $t3   # 2.2
sw  $t5, 0($t3)     # 2.3
lw  $t4, 0($t1)     # 2.4
add $t3, $t4, $t5   # 2.5
```

a) [12 points] Identify all of the data hazards that cannot be resolved with forwarding alone; that is, data hazards that do require inserting stalls.

For each such data hazard, state the number of the leading instruction, the number of the following instruction, the register involved, and explain why forwarding alone cannot resolve the hazard.

b) [8 points] Can the given sequence of instructions be reordered so that they will execute correctly, using forwarding but without inserting any nop instructions? Justify your answer.
For questions 3 and 4, refer to the pipeline design with forwarding and hazard detection, shown below, which supports execution of the following MIPS instructions: add, sub, and, or, slt, lw, and sw.

3. [20 points] Assume we have the 5-stage pipelined datapath described above, with both forwarding hardware and hazard-detection hardware. Consider executing the following sequence of instructions (same as question 2) on this datapath:

```plaintext
lw  $t3, 4($t4)  # 3.1
add $t2, $t1, $t3  # 3.2
sw  $t5, 0($t3)  # 3.3
lw  $t4, 0($t1)  # 3.4
add $t3, $t4, $t5  # 3.5
```

Where would the Hazard Detection unit insert stalls?

4. [20 points] The Hazard Detection unit does not take RegWrite as input. Why not?
5. Suppose we eliminated the **Forwarding** unit from the datapath above (along with the associated MUXes, control and data lines that would no longer serve a role). We would still want the datapath to correctly execute the same set of instructions, but now it would only use stalls to resolve data hazards, and therefore it will sometimes need to insert a sequence of two or more stalls.

   a) [10 points] Consider only data hazards between instructions that are one cycle apart; that is, those that were previously handled by forwarding from the **EX/MEM** interstage buffer.

      What inputs, in addition to the ones shown above, would the **Hazard Detection** unit require in this modified design. Briefly explain why each of those inputs is needed.

   b) [10 points] Now consider only data hazards between instructions that are two cycles apart; that is, those that were previously handled by forwarding from the **MEM/WB** interstage buffer.

      What inputs, in addition to the ones shown above, would the **Hazard Detection** unit require in this modified design. Briefly explain why each of those inputs is needed.