Create a single-cycle CPU that can perform following two instructions. The CPU should be as simple as possible.

STUR Rd, [Rn, DAddr9];
LDUR Rd, [Rn, DAddr9];

a. Draw the datapath of your CPU.

b. Can these two instructions complete in one cycle? If they can, update your datapath and show the control signal setting. If they cannot, show your reasons.
When running a program on a single cycle CPU with a 200Mhz clock and a CPI of 1, the program will take 25 seconds to execute.

If you wanted the same program to take 5 seconds to execute on a pipeline CPU with an issue rate of 1 and a 5 cycle drain, what would the clock rate have to be?
The B-Type instructions has a 26-bit branch address, but the CB-type instruction only has a 19-bit address. What could we do if we want to conditional branch to an address that has a length between 19 bits and 26 bits?
Assume that the following code is run on our 5-stage pipelined processor from class (with forwarding, accelerated branching, and branch/load delay slots). What is the resulting value of reg X5 and how many cycles does the code take to complete?

```
ADDI   X5, X31, #5
ADDI   X0, X5, #26
SUBS   X4, X5, X0
B.LT   JUMP
SUBI   X5, X0, #6
STUR   X5, [X4, #26]
JUMP:
ADD    X5, X0, X4
```
Consider a computer executing the following mix of instructions

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Frequency</th>
<th>Clock Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU</td>
<td>40</td>
<td>1</td>
</tr>
<tr>
<td>LOAD</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>STORE</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>BRANCH</td>
<td>10</td>
<td>2</td>
</tr>
</tbody>
</table>

What is the average CPI?
PLEASE specify the value for each of the control logic for the following 2 instructions. A diagram is provided below to help.
Draw a Single Cycle CPU that following instruction:

MUL Rd, Rn, Rm
What would the following C++ code be in memory? Assume we start using memory at 1000.

```c++
int var = 100;
int *ptr1;
int **ptr2;
*ptr1 = &var;
**ptr2 = &ptr1;
```
Consider the pipelined CPU that has branch and load delay slots, forwarding logic and accelerated branches. For the following code, explain what is happening in each stage of the pipelined processor during cycle 5.

```
STUR  X1, [X31, #32]
LDUR  X2, [X31, #32]
ADD   X1, X2, X2
LDUR  X1, [X31, #100]
CBZ   X1, Loop
```

Loop:
CONVERT THE FOLLOWING INTO MACHINE CODE

; TEST
    CMP  X0, X1
    B.LT  END
    B LOOP

LOOP:
    SUBI  X3, X0, #7
    STUR  X15, [X3,%3]!
    ADD  X3, X3, X2
    ADDI  X2, X2, #10
    LSL  X2, X2, #3
    B  B TEST

B END

* include the labels for the various branches (*the count for each line)*
Bad Things happened, and now our 5-stage pipelined CPU with forwarding is running the following instructions:

```
ADDI  x0, x31, #31
LDUR  x1, [x0, #1]
LDUR  x2, [x0, #9]
ADDI  x3, x2, x1
ADD   x31, x31, x31
```

What are the resulting values in registers $x_0$, $x_1$, $x_2$, and $x_3$?

You may leave answers as a sum of memory values and constants.

Example: $x_0 = \text{mem}[16] + \text{mem}[8] + 29$
HW 4 a

Translate the Java code to ARM assembly code. Use a minimum number of instructions. Assume that v is an array of integers and the length of v (v.length) is stored in register x0. You can use as many register as you want but keep it simple.

```
CompStore (int [ ] v){
    for(int k = 1; k < v.length; k++) {
        v[k] = k + v[k-1] - v[k];
    }
}
```
Consider a processor with the following delays of the individual units:

<table>
<thead>
<tr>
<th></th>
<th>IF</th>
<th>ID</th>
<th>EX</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>380 ps</td>
<td>380 ps</td>
<td>220 ps</td>
<td>300 ps</td>
<td>170 ps</td>
</tr>
</tbody>
</table>

(a) Imagine this is a pipelined processor. If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the new stage, would this affect the clock time of the processor? What would be the clock time of the old and new processor?

(b) Imagine each stage of the processor took 380 ps. How would the cycle time of a pipelined processor change? How would the cycle time of a non-pipelined processor change?