This exam has 14 pages; the final sheets are provided as reference and scratch paper, but must be handed in with the exam.

Your scantron will be used for two groups of 10 questions, for a total of 20 responses on the scantron.

You have 120 minutes.

No calculators or other electronics are allowed. You may bring one 8.5” x 11” sheet of handwritten notes that must be handed in with the exam.

To make sure you receive credit, please write clearly and show your work.

We will not answer questions regarding course material.
Question 1: Pipelining (40 points)

Above is the pipeline discussed in class.

**Part (a)** The processor is currently executing the code shown to the right. In the current cycle, the *addi* instruction is in the **writeback** stage. Assume that the branch is **not taken**. Compute the signal values for the following instruction and **enter your answers on the scantron**. For all questions, use the following key: a) 0, b) 1, c) 2, d) 3, e) don’t care/can’t know. (10 points)

<table>
<thead>
<tr>
<th>PC</th>
<th>instruction</th>
<th>0x00010000</th>
<th>addi</th>
<th>$1, $0, 768</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>bne</td>
<td>$1, $2, C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw</td>
<td>$3, 16($1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add</td>
<td>$2, $2, $3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ori</td>
<td>$2, $2, 8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw</td>
<td>$2, 16($1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Part (b)** Given the functional unit latencies on the right, what is the clock cycle of the corresponding non-pipelined (e.g., single-cycle) machine? Explain.

<table>
<thead>
<tr>
<th>Func. Unit</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU/Adder</td>
<td>4ns</td>
</tr>
<tr>
<td>Register File</td>
<td>3ns</td>
</tr>
<tr>
<td>Cache</td>
<td>5ns</td>
</tr>
</tbody>
</table>

and, what is the largest speedup that can be achieved by pipelining shown above (e.g., consider when there is a large number of instructions executed and there are no stalls and flushes)? Explain. (5 points)

**Max speedup = ___________**
Question 1: Pipelining, cont.

For all of this page consider the MIPS assembly code below. Corresponding C code is shown on the right.

```
loop:    sll  $t1, $t0, 2
         add  $t1, $a0, $t1
         lw   $t2, 0($t1)
         lw   $t3, 8($t2)
         bne  $t3, $t7, skip
         add  $v0, $v0, 1
skip:    add  $t0, $t0, 1
         slt  $t1, $t0, $a1
         bne  $t1, $0, loop
```

Part (d) Assume this loop iterate many times, and the if statement is executed 50% of the time. Compute the average number of cycles that this loop takes per iteration. Explain your answer for partial credit. Assume conditional branches are resolved in the EX stage. (10 points)

```
for (int i = 0 ; i < N ; i ++) {
    if (A[i]-->x == 1) {
        count ++;
    }
}
```

Part (c) Annotate the MIPS assembly to indicate all of the true data dependences, including those that span loop iterations. (5 points)

Part (e) Re-schedule/re-write the loop to make it faster. Faster code will achieve more points, but your answer must fit in the space below. (5 points)
Consider the data path shown above.

**Part (f)**
In what pipeline stage are jump register (jr) instructions resolved? *Explain how you know.* Feel free to highlight parts of the data path to support you answer. (5 points)
Question 2: Cache Analysis (25 points)

**Part (a)** For an 32KB 2-way set-associative, write-back cache with 64B blocks on a machine with 32-bit address spaces (both virtual and physical) and no hardware prefetching, compute the average number of misses per inner-loop iteration for the following loop nest (**and explain how you computed it**!). Assume everything is in registers, except the data structures A, B, and C. **Ignore address translation for this question.** (20 points)

\[
\text{const int N = 37, M = 2196};
\]
\[
\text{double A[N][M], C[N]; } \quad // \text{floats are 4 bytes, doubles are 8 bytes big}
\]
\[
\text{struct annotate { int index; float offset; double scaling; };
}\]
\[
\text{struct annotate B[M];
}\]
\[
\text{for (int i = 0 ; i < N ; i ++) {
\}
\text{for (int j = 0 ; j < M ; j += 2) {
\}
\}
}\}
\]

**Part (b)** How would the miss rate change if we inverted the loops (shown below)? **Explain.** (5 points)

\[
\text{for (int j = 0 ; j < M ; j += 2) {
\}
\text{for (int i = 0 ; i < N ; i ++) {
\}
\}
}\}
**Question 3: Cache-aware Programming (25 points)**

**Part (a)** Rewrite the following code to optimize its cache performance on a system with **no support for prefetching** and a single-level cache. The cache is a 2-way set associative 16KB cache with 32B blocks. Floats are 4 byte data types. (15 points)

```c
#define N 100000
float A[N], B[N], total_sum = 0;

for (int i = 0 ; i < N ; i += 20) {
    for (int j = 0 ; j < N ; j += 20) {
        total_sum += A[i] * B[j];
    }
}
```

Question 3: Cache-aware Programming (cont.)

Part (b) Rewrite the following code to optimize its cache performance on a system with no support for prefetching and a single-level cache. Assume the L1 cache is a 2-way set associative 32KB cache with 32B blocks. Double precision floating point numbers are 8 byte data types. (10 points)

const int N = 10000;
double A[N], B[N], C[N], D[N], E[N];

for (int i = 0 ; i < N ; i ++) {
    C[i] = k * (D[i] + B[i]);
}
for (int i = 0 ; i < N ; i ++) {
    E[i] = A[i] / E[i];
    D[i+1] = (D[i] + C[i]);
}
Multiple Choice (10 points)

Choose the best answer and answer on the provided scantron.

For a 16KB 4-way set-associative, write-back cache with 64B blocks on a machine with 32-bit address spaces (both virtual and physical):  

11) How many bits is the block offset?  
   a) 3  
   b) 4  
   c) 5  
   d) 6  
   e) 7

12) How many bits is the index?  
   a) 3  
   b) 4  
   c) 5  
   d) 6  
   e) 7

13) Which of the following is false, or select (e) if all of the statements are true:
   
   a. The page table base pointer can only be written by the operating system.
   b. A page fault is when the desired page is not currently mapped into memory.
   c. Accessing multi-level page tables involves using a different portion of the virtual page number as the index for each level of the page table.
   d. Virtual memory systems rely on both spatial and temporal locality for efficiency.
   e. All of the statements are true.

14) If everything else is equal, a higher clock period means better performance.  
   A) TRUE  
   B) FALSE

15) When reading a single random sector from a disk, what is likely to be the smallest component of the read time:  
   A) circulation latency  
   B) transfer time  
   C) transition time  
   D) seek time  
   E) rotational latency

16) Pipelining improves:  
   A) Throughput, but not latency  
   B) Latency, but not throughput  
   C) Both latency and throughput  
   D) Neither latency nor throughput

17) Assuming the array B is large, what kind(s) of prefetching are likely to effective for prefetching accesses to B?

for (int i = 0 ; i < LENGTH ; i ++) {
   A[i] = B[C[i]]
}

   A) hardware prefetching  
   B) software prefetching  
   C) either hardware or software prefetching  
   D) neither hardware nor software prefetching

18) In real machines, the least recently used policy works because it takes advantage of:
   A) spatial and temporal locality in programs  
   B) spatial locality in programs (not temporal)  
   C) temporal locality in programs (not spatial)  
   D) neither spatial nor temporal locality
Multiple Choice, cont.

For the following pieces of code, indicate which forms of locality are definitely exhibited for the accesses to the bolded data structure:

   a) temporal locality   b) spatial locality   c) both   d) neither

19) for (int i = 0 ; i < NUM ; i ++) {
    for (int j = 0 ; j < NUM ; j ++) {
        A[i][j] = initial_value[i];
    }
}

20) for (int i = 0 ; i < NUM ; i ++) {
    x += flights[i]->cost;
}