1. (a) Identify all data dependencies in the following sequence of instructions, when executed on the pipelined MicroMIPS implementation (with no forwarding).

```
        addi  $9, $zero, 0    [1]
        addi $12, $zero, 5000 [2]
loop:   addi $12, $12, 4      [3]
        lw    $8, 40($12)     [4]
        add   $9, $9, $8      [5]
        addi $11, $11, -1     [6]
        bne  $11, $zero, loop [7]
```

Solution: Instruction [3] depends on [2] (for 1-st iteration), [4] depends on [3], [5] depends on [4] and [1], [7] depends on [6].

(b) Determine the number and places of bubbles that must be inserted to correctly execute the instruction sequence in part (a). Explain your reasoning in each step.

Solution: NOTE: Here, 2@3.5 means 2 bubbles are inserted between instructions [3] and the following instruction.
Assuming a register can be read out in the same cycle at during which it is updated, we have 2@2.5, 2@3.5, 2@4.5, and 2@6.5.

(c) Can you suggest any reordering of instructions that would reduce the number of bubbles ?

Solution: One possible reordering,

```
        addi $12, $zero, 5000 [2]
        addi  $9, $zero, 0    [1]
loop:   addi $12, $12, 4      [3]
        addi $11, $11, -1     [6]
        lw    $8, 40($12)     [4]
        add   $9, $9, $8      [5]
        bne  $11, $zero, loop [7]
```

Now we have, 1@1.5, 1@6.5, 2@4.5.

2. [5 marks] (*****) Data hazards occur if the sequence of read and write (to/from memory or registers) in the instruction sequence are such that the end result is inconsistent. There are four combinations of the read/write sequences and we will name them, "X after Y", where X and Y could be *Read* or *Write* (RAW, RAR, WAW, WAR).

(a) Which of these four are possible hazards in general pipeline implementations? Explain why the other(s) are not hazards.

Solution: RAW, WAW, and WAR are all possible hazards. RAR will not be a hazard since data is not modified due to a read.

(b) Which of the potential data hazards that you enumerated above, if any, is not a hazard in the 5-stage pipeline implementation that we have considered in class. Why?

Solution: WAW and WAR are not hazards in the 5-stage pipeline implementation since the write operations for the second instruction happen only after read or write of the first instruction in the pipeline.

(c) Consider the following instruction sequence. What type of a data hazard does this sequence contain.

```
add  $r2,  $r3  $r4
sub  $r5,  $r6  $r2
```

Solution: RAW type of hazard.

(d) Consider the instruction sequence below

```
sw  $r0,  100($r2)
lw  $r1,  100($r2)
```

Compare the sequence of operations in the above sequence and the one in the previous question and explain why there will be or will not be a "read after write" hazard in the 5-stage implementation that we consider in class.

Solution: There will be no RAW, since the MEM stage of the 1-st instruction completes before the MEM stage of the 2-nd instruction.

(e) Consider the following code sequence to be implemented on the basic 5-stage pipeline that we have considered in class (with no throughput enhancement technique employed)

```
add  $r5,  $r6  $r7
lw   $r6,  100($r7)
sub  $r7,  $r6  $r8
```

Assuming that the pipeline is flushed before this segment begins execution, how many clock cycles does this segment take. Show the instruction in each stage of the pipeline in each clock cycle.

Solution:

	IF	**ID**	**EX**	**MEM**	**WB**
1	add				
2	lw	add			
3	sub	lw	add		
4	x	sub	lw	add	
5	x	sub	STALL	lw	add
6	x	sub	STALL	STALL	lw
7	x	x	sub	-	-
8	x	x	x	sub	-
9	x	x	x	x	sub

3. [3 marks] (*) A computer architect needs to design the pipeline of a new microprocessor. An example workload program core has 10^6 instructions, and each instruction takes 100 ps to finish.

(a) How long does it take to execute the workload program on a nonpipelined processor ?

(b) The current state-of-the-art microprocessor has about 20 pipeline stages. Assume it is perfectly pipelined. How much speedup will it achieve compared to the nonpipelined processor ?

(c) Real pipelining isn't perfect, since implementing pipelining introduces some overhead per pipeline stage. Will this overhead affect instruction latency, instruction throughput, or both ?

Solution:

(a) Execution time = $10^6 \times 100$ ps = 100 μs.

(b) Speedup = 20.

(c) Pipeline affects both instruction latency and instruction throughput. Ideally pipelining does not improve the individual instruction latency, but it will affect the average instruction latency. This is due to the additional hardware that should be added for instruction pipelining.

4. (a) Delayed branching can help in the handling of control hazards. For all delayed conditional branch instructions, irrespective of whether the condition evaluates to true or false,
   A. the instruction following the conditional branch instruction in memory is executed
   B. the first instruction in the fall through path is executed
   C. the first instruction in the taken path is executed
   D. the branch takes longer to execute than any other instruction

   Solution: (A) the instruction following the conditional branch instruction in memory is executed

   (b) The following code is to run on a pipelined processor with one branch delay slot:

```
I1: add $r2, $r7, $r8
I2: sub $r4, $r5, $r6
I3: add $r1, $r2, $r3
I4: sw $r1, 100($r4)

...
bltz $r1, Label
                <-- Delay slot
...
```

   Which of the instructions I1, I2, I3 or I4 can legitimately occupy the delay slot without any other program modification?

   Solution: I1 or I2 can occupy the delay slot.

5. Consider three machines with the same processor but different cache configurations. *C1* is direct mapped with one-word blocks, *C2* is direct mapped with four-word blocks and *C3* is two-way set associative with four-word blocks. *C1* has an instruction miss rate of 4% and a data miss rate of 8%, while the corresponding numbers for *C2* are 2% and 5% respectively and for *C3* are 2% and 4% respectively. Assume half of the instructions have a data reference, and the cache miss penalty is 6 + `Block size in words`. The CPI for this workload on a machine with *C1* was measured to be 2.0. Determine the time spent by each machine on cache misses. If the clock period is 2.0 ns for the first and second configurations and 2.4 ns for the third, determine the MIPS (million instructions per second) rating of the three machines.

Solution:

Cache	Miss penalty	Instr. miss cycle per instr.	Data miss cycle per instr.	Avg. miss cycle per instr.
C1	$6 + 1 = 7$	0.28	0.56	0.56
C2	$6 + 4 = 10$	0.20	0.50	0.45
C3	$6 + 4 = 10$	0.20	0.40	0.40

MIPS ratings would be, for C1 $(2.56 \times 2 \text{ ns})^{-1} = 195.3$, for C2 $(2.45 \times 2 \text{ ns})^{-1} = 204$, and for C3 $(2.4 \times 2.4 \text{ ns})^{-1} = 174$.

6. [4 marks] (*) The following C program is run (with no optimizations) on a machine with a cache that has four-word (16-byte) blocks and holds 256 bytes of data

```
int i, j, c, stride, array[256]
. . .
for (i=0; i<10000; i++)
  for (j=0; j<256; j=j+stride)
  c=array[j]+5;
```

If we consider only the cache activity generated by references to the array and we assume that integers are words, what is the expected miss rate when the cache is direct mapped and `stride` = 132 ? How about `stride` = 131 ? Would either of these change if the cache were to be two-way set-associative?

Solution: Each block has 4 array elements and the cache can hold 16 blocks. Hence, array[0], ..., array[3] are in block#0 array[4], ..., array[7] are in block#1; and so on until block#15. Then wrapping around block#0 will have array[64] to array[67]. Accordingly array[132] will be in block#1. Thus array[0] and array[132] will be in block#0 and block#1, respectively. Therefore, all accesses except for the first will be a hit and we will have nearly 100% hit rate.

On the other hand, array[131] will be placed in block#0,.i.e.,array[0] will be replaced. All accesses to array[0] and array[131] will be misses, so there will be a 100% miss rate.

If the cache is two-way set associative, access to array[0] and array[132] will be to the same as was in the direct mapped case. For array[0] and array[131], they can coexist in the same set#0 and hence nearly 100% hit rate.

7. A processor with two levels of caches had a CPI of 1 when there is no level-1 cache miss. At level-1, the hit rate is 95 % and a miss incurs a 10-cycle penalty. For the two-level caches as a whole, the hit rate is 98 % (meaning that 2% of the time the main memory must be accessed) and the miss penalty is 60 cycles.

   (a) What is the effective CPI after cache misses are factored in ?

   Solution: Effective CPI $= 1 + 0.05 \times 10 + 0.02 \times 60 = 2.7$.

   (b) If a single-level cache were to be used in lieu of this two-level cache system, what hit rate and miss penalty would be needed to provide the same performance ?

   Solution: If hit rate is h and miss penalty is c cycles, for comparable performance to that in part (a) we need $(1 - h)c = 1.7$.

8. The following sequence of number represents memory addresses in a 64-word main memory: 0, 1, 2, 3, 4, 15, 14, 13, 12, 11, 10, 9, 0, 1, 2, 3,4, 56, 28, 32, 15, 14, 13, 12, 0, 1, 2, 3. Classify each of the accesses as a cache hit, compulsory miss, capacity miss, or conflict miss, given the following cache parameters. In each case, also provide the final cache contents.

   (a) Direct-mapped, 2-word block, 4-block capacity

   (b) Two-way set associative, 4-word blocks, 2-set capacity, LRU replacement

9. [4 marks] (*) A computer system has 4 GB of byte-addressable main memory and a 256 KB unified cache memory with 32-byte blocks.

   (a) Draw a diagram showing each of the components of a main memory address (i.e., how many bits for tag, set index, and byte offset) for a four-way set-associative cache.

(b) The performance of the computer system with four-way set-associative cache architecture proves unsatisfactory. Two redesign options are being considered, implying roughly the same additional design and production costs. Option A is to increase the size of the cache to 512 KB. Option B is to increase the associativity of the 256 KB cache to 16-way. In your judgement, which option is more likely to result in greater overall performance and why ?

Solution:

(a) 16-bits for tag, 11-bits for set index, and 5-bits for offset.

(b) This is just for understanding.

(c) Doubling the size of cache will likely have a greater impact on performance.

10. An address for a byte-addressable memory presented to the cache unit is divided as follows: 13-bit tag, 14-bit block index, 5-bit byte offset.

(a) What is the cache size in bytes ?

(b) What is the cache mapping scheme ?

(c) For a given byte in cache, how many different bytes in the 2^{32}-byte main memory can occupy it ?

Solution:

(a) $14 + 5 = 19$ bits; this implies cache size $= 2^{19} = 512$ KB.

(b) Direct mapped since no mention about associativeness.

(c) As many as there are different tags, i.e., $2^{13} = 8192$.