CS220 Sample Exam 1 Questions

The only outside materials you may use on this exam is the MIPS green card. You may not use a calculator, your book, notes, or the SPIM simulator.

1. Write the decimal number 126 as an 8-bit two's complement binary number.

10000010

2. Why does the MIPS multiply instruction need two registers, **HI** and **LO**, for the result?

Because multiplying two 32-bit numbers yiels a 64 bit result.

3. How many different opcodes could the MIPS instruction set support?

64 – why?

4. Write the number **-73** as an 8-bit two's complement binary number.

10110111

5. Overflow can occur when adding or subtracting two n-bit two's-complement binary numbers. How do you know when overflow occurs?

Add two numbers of the same sign and get a number of the opposite sign

6. What decimal number does the 16-bit two's complement number 1111111111111100 represent?

-4

7. Name four different regions of memory our MIPS assembly programs use.

Look at the green card

8. For an ***n***-bit two's complement number what is the range of decimal values that can be represented? Express your answer in terms of ***n***.

-2^{n-1} *to* $2^{n-1}-1$

9. What MIPS instruction does the 32-bit binary machine code 00000000_{hex} represent?

n***op***

10. What is the machine code representation for the MIPS instruction **slti $v0, $s2, 10** ?

Don't worry about it – we didn't cover this yet

11. Consider the following function **f** in MIPS assembly language. Notice that **f** calls a function **g**, where **g** takes no parameters and **g** returns a value in **$v0**. We don't know who called **f.** Note that **f** also returns its result in **$v0**.

```
f:mul $s0, $a0, $a0  # use s0 to hold square the first parameter

  add $t0, $s0, $s0  # use t0 for something

  jal g              # call g, result of g comes back in $v0
```

```
add $v0, $t0, $v0 # add result of g and $t0 (where $t0 was calculated above)

add $v0, v0, $s0     # add $v0 and $s0 (where $s0 was calculated above)

jr $ra           # return from f
```

Function **f** has several problems where it does not obey the MIPS procedure calling conventions. Rewrite **f** so that it obeys the MIPS procedure calling conventions. Add comments explaing.

<u>Don't worry about this one we didn't cover it yet.</u>

12. Write a MIPS assembly language program that reads an integer from the console (use a nice prompt) and counts the number of ones in that integers binary representation and print the result back to the console.

- Hint: If the number is odd that means bit-zero, the least significant bit, is a 1. Add it to the count. Now shift the entire number to the right one place and repeat until zero.
- Full credit for having **main** call a function named **count_ones** that takes the inetegr as a parameter and returns the number of ones to **main**. This function should also obey the MIPS procedure calling conventions.

I wont write this out in assembly but here's the algorithm.

```
Read n from the console
count = 0;

while n != 0 {
  if n is odd
    count = count + 1;

  n = n / 2;   // dividing by two moves all the bits to the right once. Why?
}
```

13. Suppose processor $\mathbf{P_1}$ has a cycle time of 0.9 nanoseconds and processor $\mathbf{P_2}$ has a cycle time of 0.6 nanoseconds. Furthermore $\mathbf{P_1}$ has a CPI of 2.5 for a program and $\mathbf{P_2}$ has a CPI of 3 for the same program. Which machine is faster for the program and by how much?

Find the time for P1 and the time for P2 and take a ration canceling out the IC. P1 is slower. P2 is faster by 1.25 times or 25%.

14. What is the machine code representation for the MIPS instruction **addi $t0, $t2, -1**

Don't worry about this. We didn't cover this yet.

15. Assume that the execution time of a program **p** on a processor is 192 seconds. Also assume that **p** spends 25% of the execution time doing floating-point multiplications. What would the execution time be if we doubled the performance of floating-point multiplication? What is the overall performance improvement of **p** running on the modified processor?

After we double the floatinf point speed the program takes 168 seconds. Improvement is 192/168 is 14%.

16. If the clock cycle time for a processor is 3.3×10^{-10} seconds what is the clock rate?

Just take the reciprocal, its about 3 GHz

17. Assume a program takes 10.7 billion instructions to execute on a processor and the processor has an average CPI of 1.5 for that program. What would the clock rate of the processor need to be to allow the program to execute in two seconds?

Just solve for the rate in the CPU Time equation. The answer is close to 8GHz (not very likely to achieve that).