CSC 373 Winter 2020 Prof. Lytinen
Midterm Exam

Name:

Directions:

You have 90 minutes to complete this exam. It is open book, open notes, and you may use your laptop or other computer (calculators are also allowed). You may not communicate with others during the exam (except with me or your proctor), either in person or electronically. The exam is worth 25% of your overall grade for the quarter, and will be graded on a scale of 0-25. Note the point value of each problem. Please upload your solutions to the Midterm D2L submissions folder.

I have provided a template file for the coding problems called **midterm.c**. If you would like to use it, there is also a **midtermtest.c** file, which will run the functions in **midterm.c** . Because the online students are not taking the exam at the same time as the inclass students, I have placed these files at **http://condor.dpu.depaul.edu/~slytinen/373w20/midterm.zip**

Problems

1. (3 points) How much memory space does a process think that it has to work with? How does the operating system provide this illusion to processes?.

2. (3 points) Consider this program. What is its output? Explain. Note that the output is in decimal.

```
int main() {
  char x = 0x7f;
  x += 1;
  printf("%d\n", x);
}
```

3. (5 points) We would like to write a C function which fills an array with the powers of 2 that are less than or equal to **x** (some positive integer), and which returns the number of powers of 2 that are generated. The integer **x** is passed as a parameter (and perhaps some other parameters as well). Write the **powers_of_2** function.

4. (5 points) Without using any functions in the <string.h> library, write a function **to_upper_case** , which changes any alphabetic lower case characters in a string into upper case characters. The function returns void. Use pointer syntax.

5. (4 points) Fill in the table below. Any binary number that starts with 1 is a negative number in 2s complement. Likewise for any hex number that starts with 8-f. Assume that the numbers are represented in 1 byte.

Decimal	**8-bit Binary (2s complement)**	**2-digit Hex (2s-complement)**
33		
	11110110	

6. (1 point) Fill in the table.

Base 10	**Binary scientific**	**IEEE 32-bit**
3 1/4		

7. (4 points) Write a C function that emulates the behavior of this x86-64 assembly language function. Your C code does **not** necessarily have to compile into the exact same assembler code; rather it is sufficient if the C function returns the same value as the assembly language provided it is passed the same parameters.

```
f:
  xorl %eax, %eax
  cmpl %edi, %esi
 sete %al, 1
 addl %edx, %eax
ret
```