**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?.

A process on a 64-bit machine thinks that it has 2^64 bytes of memory space. The operating system provides this allusion by splitting up the memory space into the kernel, user address space and the illusory space which. The user address space contains the user stack and the run time heap.

1. (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);

}

-128 is the output. This is because chars are only 1 byte long thus 0x7f is the max value. Adding one would turn the result negative since the first bit of a number indicates its sign. 0x7f is 0111 1111 in binary, adding 1 to this would turn it into 1000 0000 which is -128.

1. (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.

On abelmarin\_midterm.c

1. (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.

On abelmarin\_midterm.c

1. (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 | 0010 0001 | 0x21 |
| -10 | 11110110 | 0xf6 |

1. (1 point) Fill in the table.

|  |  |  |
| --- | --- | --- |
| **Base 10** | **Binary scientific** | **IEEE 32-bit** |
| 3 1/4 | 1.101 \* 2^1 | 0 10000000 10100000000000000000000 |

1. (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

On abelmarin\_midterm.c