ANSWER SHEET (DOUBLE SIDED)

ALL OF YOUR ANSWERS MUST BE ON THIS SHEET

NAME: _____

ID: _____

1) ***This Problem Bytes*** – Evaluate for each value of x. Answer in hexadecimal.

```
a) x = 0x04030201        answer: 0x
b) x = 0xfcfdfeff        answer: 0x
c) x = 0x04fd02ff        answer: 0x
d) x = 0x7f7f7f7f        answer: 0x
e) x = 0x807f807f        answer: 0x
```

2) Lost at C? – determine whether each is always true or not – if false, provide a counterexample.

			Circle your answer (true or false)	
a) `x > y`	→	`(y - x) < 0`	true	false counter example:
b) `x + y = 0`	→	`x = ~y + 1`	true	false counter example:
c) `x > ((-1)U >> 2)`	→	`(x * y) > 0U`	true	false counter example:
d) `x * y > 0U`	→	`(x ^ y) > 0`	true	false counter example:

3) ***Down in the Dumps*** – assume x = 26, w = 3, and v = 2. Answer the following at the time of the printf():

a) What is the starting address of table0? 0x

b) What is the value of variable index? 0x

c) What is the value of &table0[index]? 0x

d) What is the short stored at table0[index][w][v]? 0x

4) ***Mineshafts and Manticores*** – your hero is a PALADIN.

a) What is the address to handle PALADIN that is stored in the jump table: 0x

b) What weapon do you need to defeat the manticore – circle one:

AXE SPEAR MACE SWORD HALBERD GLAIVE

5) ***Reaching My Breaking Point*** – Suppose we set a breakpoint at the invocation of search_node and then run the program. The first time this breakpoint is hit, we dump some registers with `i r` in gdb:

```
rax            0x19 25
rcx            0x18 24
rsi            0x3e 62
rdi            0x6034f0  6305008
rip            0x400837  0x400837 <search_node>
```

Now we set a different breakpoint at 0x40085e. The first time this breakpoint is triggered, we dump:

```
(gdb) x/192xb $rsp
0x7fffffffe0d0:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe0d8:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe0e0:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe0e8:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe0f0:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe0f8:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe100:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe108:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe110:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe118:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe120:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe128:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe130:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe138:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe140:     0xf0 0x34 0x60 0x00 0x00 0x00 0x00 0x00
0x7fffffffe148:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe150:     0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe158:     0x74 0x08 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe160:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe168:     0x24 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffe170:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe178:     0x55 0xf5 0xa2 0xf7 0xff 0x7f 0x00 0x00
0x7fffffffe180:     0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x7fffffffe188:     0x58 0xe2 0xff 0xff 0xff 0x7f 0x00 0x00
```

Function search_node() is first invoked by this instruction (shown in objdump format):

```
400a1f:       e8 d1 fd ff ff          callq  400837 <search_node>
```

The printf at address 0x40085e will output a string (%s), among other things. Answer the following:

a) What is the key that we are looking for in the linked list (e.g. the value of the short): 0x

b) How many times will the search_node function be executed (in decimal)? (e.g. # of times RIP is set to `0x400837`)

c) What is the address of the specific string `stringy` to be printed using printf: 0x

d) What is the output that will be printed by the printf at 0x`40085e`: