1. What is the value of y after both of the following operations?

x = x ^ (~y);

y = y ^ x;

y = y ^ (x ^ (~y));

y = y ^ ~y ^ x = x

1. Given the following declarations, do the statements below always evaluate to true?

int x = foo();

int y = bar();

unsigned ux = cookie();

a.

x > ux ====> (~x+1) < 0

No. If x is negative, the implicit unsigned cast can cause it to be larger than ux. Then, -x is larger than 0.

b.

ux - 2 >= -2 ====> ux <= 1

Yes. To be greater than or equal to -2 in an unsigned comparison requires the bits to be either all ones or all ones followed by one zero. This can only happen when ux is 0 or 1.

c.

(x^y)^x == (x+y)^((x+y)^y)

X^x^y = y = 0^y

d.

(x < 0) && (y < 0) == (x + y) < 0

False, if overflow occurs upon adding x and y, a positive value can be achieved.

1. char\*\* apple[5][9];

char\* banana[1][9];

char strawberry[4][2];

How many bytes of space would these declarations require?

8\*5\*9, 8\*1\*9, 4\*2\*1

1. Consider the following struct:

typedef struct {

char first;

int second;

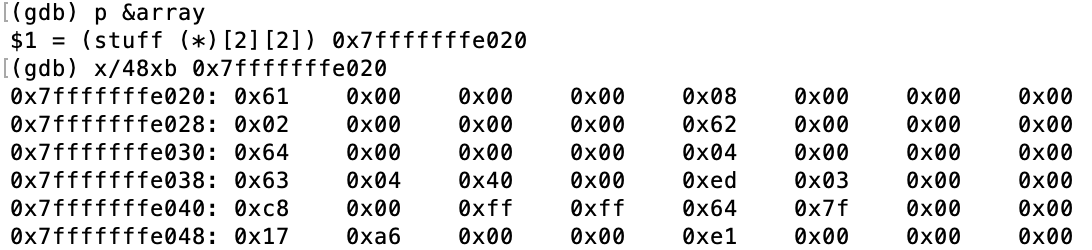
short third;

} stuff;

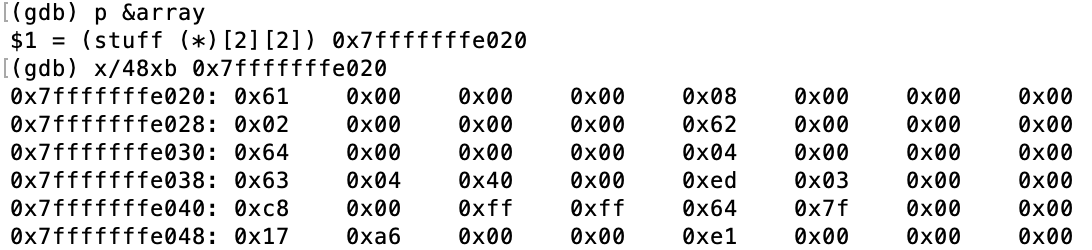
Say we are debugging an application in execution using gdb on a 64-bit, little-endian architecture. The application has a variable called array - defined as:

stuff array[2][2];

Using gdb we find the following information at a particular stage in the application:



And:



What is the value of

array[1][0].second

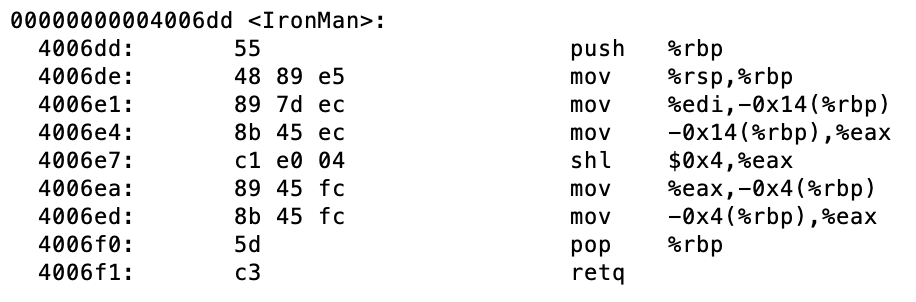
At this particular stage of the application?

i.e. what would be returned from the statement:

printf("%d\n", array[1][0].second);

array is a 2d array of stuff – and each stuff is of size 12. Thus, array[1][0] will be past the first element array, which consists of two stuff structs, meaning 24 bytes. Then, to get array[1][0].second, add 4 bytes. Thus, the 4 bytes after that byte value will be the int second. 0x03ed.

1. The following is part of the result of the command ‘objdump -d’ on an executable



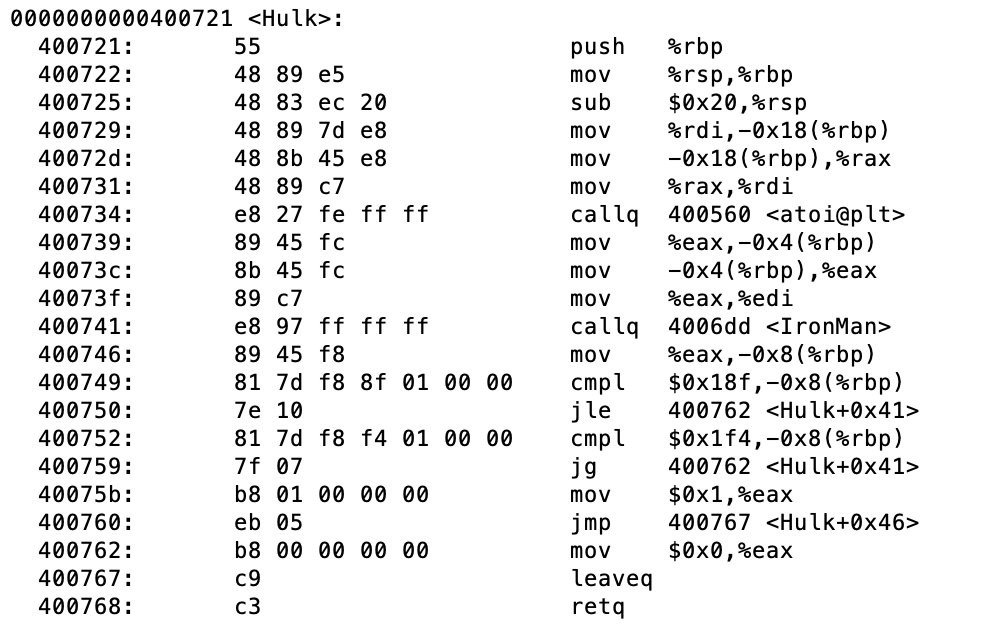
Say the declaration for the function IronMan was:

int IronMan(int scraps);

Given that the integer 23 was passed into the function, what is the return value?

23 \* 16

1. The following is a continuation from the previous problem:



Given that the function returns 1, what do we know about the value of %edi right before instruction 0x400741 is executed?

%edi contains a 32 bit integer.