John Carroll
Jcc0044
902521946

# Homework 6
# COMP 3350

**1.** Run-time stack:

| # | Value | Points to: |
|---|---|---|
| 1 | 0012ff58h | The saved value of EBP |
| 2 | 0040105ch | The return address for a recursive call. |
| 3 | 00000000h | The argument for a recursive call. |
| 4 | 0012ff64h | The saved value of EBP |
| 5 | 0040105ch | The return address for a recursive call. |
| 6 | 00000001h | The argument for a recursive call. |
| 7 | 0012ff70h | The saved value of EBP |
| 8 | 0040105ch | The return address for a recursive call. |
| 9 | 00000002h | The argument for a recursive call. |
| 10 | 0012ff94h | The saved value of EBP |
| 11 | 0040103bh | The return address for the initial call. |
| 12 | 00000003h | The argument for the initial call. |
| 13 | 99999999h | garbage |
| 14 | 99999999h | garbage |
| 15 | 99999999h | garbage |
| 16 | 99999999h | garbage |

**2.** Number of bytes will be pushed onto the stack when the breakpoint is hit is found in the equation:

Number of bytes = 16 + (12(i + 1))

Ex: 16 + (12(0 + 1)) = 28

16 + (12(3 + 1)) = 64

**3.** Declarations after the 0BEEFh,

```
WORD 0F00Dh
DWORD    11223344h
BYTE     0A1h
BYTE     0B2h
```

**4.** 2 + 2 + 4 + 1 + 1 = 10 bytes, not sure about DWORD 3344h in Q.3, which would be 4 bytes for that part as is. STOP = 0040500Ah

**5.** Subquestions:

John Carroll
Jcc0044
902521946

a)

0 10000001 01010000000000000000000

1 sign bit, 8 biased exponent bits, 23 significant bits

**Formula:**

(-1)sign bit * 1.significant bits * 2(exponent bits) – bias

(-1)0 x 1.01010000000000000000000 x 210000001 – 127

**Convert to decimal:**

(-1)0 x 1.3125 x 2129-127

1 x 1.3125 x 22

1.3125 x 4 = **5.25**

b) This is not a compiler error because _value$ = -4 is storing the value in the same manner, and reserving space for a local. Justified by footnote.

c) XOR EAX with itself gets 0 in EAX

d)
```
fld     DWORD PTR _value$[ebp]
fadd    QWORD PTR __real@3ff0000000000000
fstp    DWORD PTR _value$[ebp]
```

**fld loads onto a floating-point stack, FPU at ST(0), what is in a location in EBP**
**fadd adds on the FPU what was in its previous spot**
**fstp stores into EBP what is popped off of FPU's stack.**