Recitation#5: Reinforcements on code execution and memory layout

CS232 Spring 2021

When: February 26 at 2:00 pm

Puzzlers for Pointers, Addresses, and Values

A memory has the following contents (in little-endian format)

Variable	Address	Bytes	Final Value of Byte
A	0x08000000	**00 00 00 08**	
B	0x08000004	**04 00 00 08**	
C	0x08000008	**fe ff ff ff**	
D	0x0800000C	**ff ff ff ff**	
E	0x08000010	**00 00 00 00**	
F	0x08000014	**01 00 00 00**	
G	0x08000018	**02 03 04 05**	
H	0x0800001C	**33 35 31 00**	

Given the following declarations (assuming a 32-bit architecture):

```
int *A, *B; float C; int D; float E; int F; float G;

typedef struct xform {
    int i[2][2];
    float * factor;
    int color;
} xform;

xform *H;
```

Fill in columns for the address (in hex) that is changed in each statement and the value (in hex) to which it is changed. **NOTE: The statements are executed in sequence and changes made to memory apply in the following lines**.

C statements	Address(hex)	Value(hex)
`A = B + 2;`		
`C = (float) (*A + F);`		
`H = (xform *) &B;`		
`H->factor = &E + 2;`		
`D = (int) *((char *)(H->factor));`		
`H->i[(D >> 1)][1] = D + 3;`		

A->B+2

=0x08000004 + (2)

↳ 2*(sizeof int)

2*4 = 8

=0x08000004 + 8

=0x0800000C

C = (float)(*A+F)

↓

ffff ffff + 01000000

X0000000 = overflow

H = (xform*) &B

H → factor = &

IC 04

H → factor = &E + 2 (float) 2x4 = 8B

*(H).factor = 0x08000010 + (2)

=0x08000010 + 8

=0x08000018

xform *H {

int i[2][2] = 4x4 = 16B

float *factor = 4B

int color = 4B

} i = &i[0][0]

= 24 B

D = (int) * ((char*) (H->factor))

(int) * ((char*) (*H).factor)

1B 02 (char) => 00000002

H->i[(0->1)][1] = D + 3

2->1 = 00000010->1 = 00000001 = 1

i[1][1] = D + 3 = 02 + 3 = 05

E = 05