CS208

F, 3 April 2026

int — (in C, using our gcc compiler)

32-bit / 4-byte

type

```
int x = 0x0123ABCD;
```

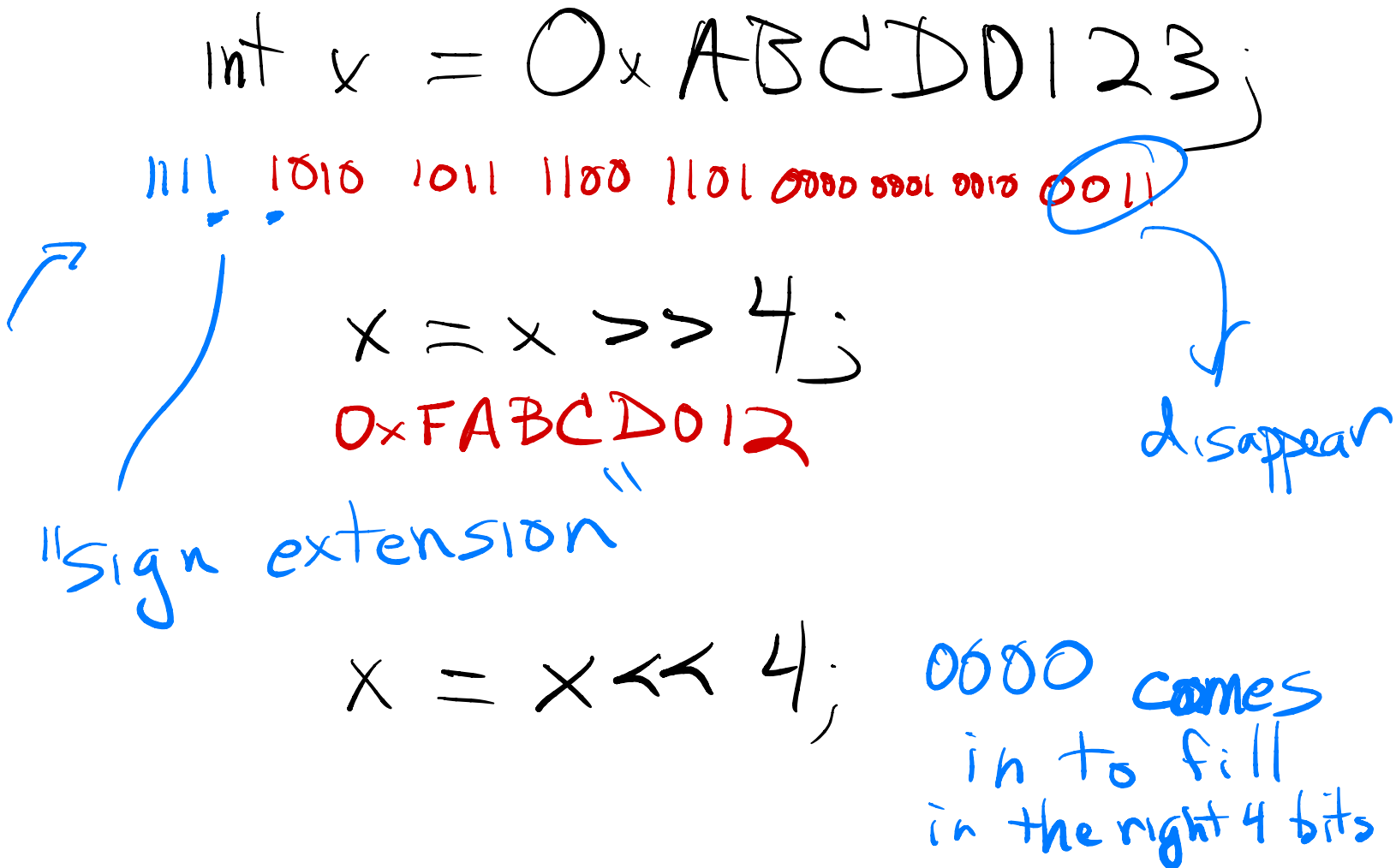
0000 → 0000 0001 0010 0011 1010 1011 1100 1101

```
x = x >> 4;
```

0000 0000 0001 0010 0011 1010 1011 1100

fall off the end

0x00123ABC


int x = 0xABCDD123;
1111 1010 1011 1100 1101 0000 0001 0010 0011
x = x >> 4;
0xFABCDD12
disappear
"sign extension"
x = x << 4;
0000 comes in to fill in the right 4 bits

175_{ten}  $1 \rightsquigarrow 1750$

shift left one decimal digit

$= 10 \times$ original

$0b10110 << 1 \rightsquigarrow 0b101100$

$16+4+2 = 22$

$32+8+4 = 44 = 2 \times$ original

```
int x = 0x0123ABCD;

print_integer_in_binary(x)
```

Strategy:

```
print(x % 2)
x = x >> 1;
print(x % 2)
x = x >> 1
⋮
```

prints backwards

Strategy 2

```
int x = 0x0123ABCD;
unsigned int mask = 0b1000000000000000000000000000000;
```

0x80000000; 31 0's

print(1 if x & mask ≠ 0; 0 otherwise)

```
mask = mask >> 1;
```

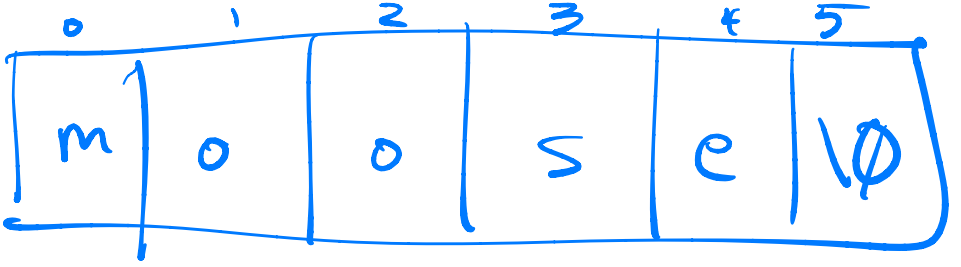
want mask 0x40000000

get 0xC0000000

In C

"moose"

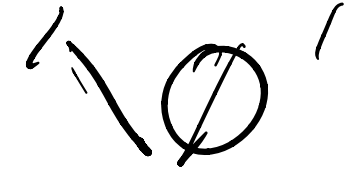
is a "null-terminated char string"



In C

'm'

a single char/byte containing the ASCII value of m

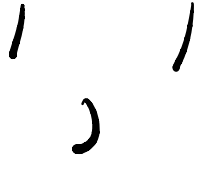


0000 0000

null byte

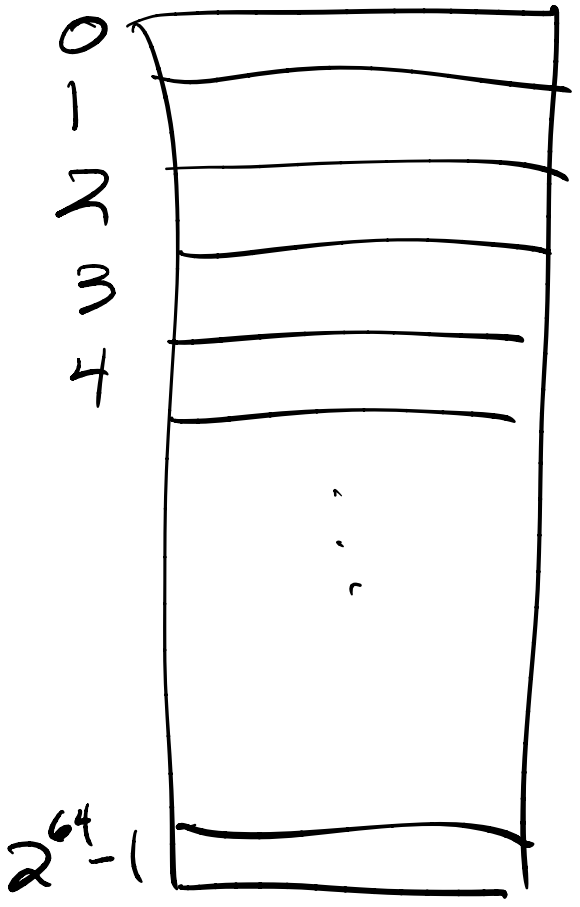
0000 1010

newline or linefeed byte



comma byte

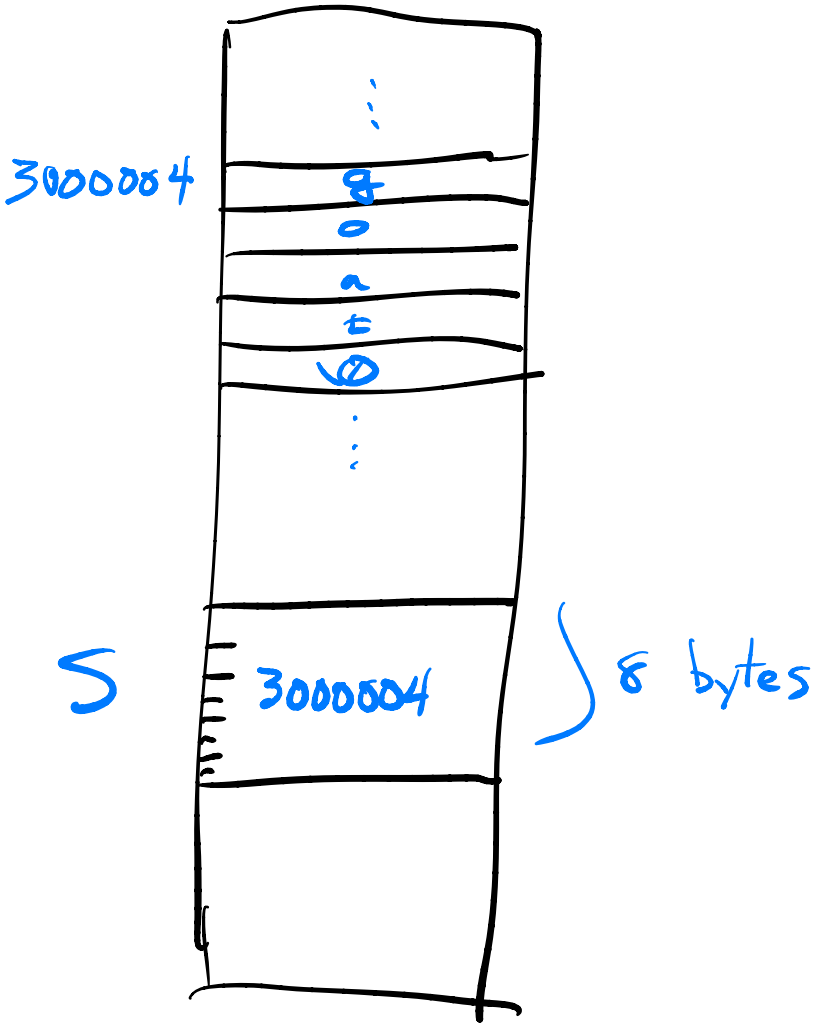
Memory



Sequence of bytes

Each byte has a numerical "address"

```
char *s = "goat";
```



char *s = "goat";

s is a "pointer to" (or "address of") the null-terminated string