C TYPES

On n bits I can represent 2^n different *things*

C TYPES

On n bits I can represent 2^n different *things*

On 1 byte = 8 bits, I can represent 2^8 = 256 *things*

C TYPES

On n bits I can represent 2^n different *things*

On 1 byte = 8 bits, I can represent 2^8 = 256 *things*

But what do I want to represent?

C TYPES

On n bits I can represent 2^n different *things*

On 1 byte = 8 bits, I can represent 2^8 = 256 *things*

But what do I want to represent?

The most “natural”: **all positive numbers in [0,255]**

C TYPES

With unsigned char we get the standard binary representation of 8-bit numbers:

- 00000000: **0**
- 00000001: **1**
- 00000010: **2**
- 00000011: **3**
- **…**
- 11111111: **255** = 2^8 - 1

C TYPES

With unsigned char we get the standard binary representation of 8-bit numbers:

- 00000000: **0**
- 00000001: **1**
- 00000010: **2**
- 00000011: **3**
- **...**
- 11111111: **255** = 2^8 - 1

BE CAREFUL OF **OVERFLOW**!!!!

C TYPES

More generally: unsigned n-bit integers are in [0,2^n-1]

- **unsigned char:** 1 byte, numbers in [0,2^8-1]
- **unsigned short:** 2 bytes, numbers in [0,2^16-1]
- **unsigned int:** 4 bytes, numbers in [0,2^32-1]
- **unsigned long:** 8 bytes, numbers in [0,2^64-1]

C TYPES

- **unsigned** n-bit integers are in [0,2^n-1]
- **signed** n-bit integers are in [-2^(n-1), 2^(n-1)-1]