# CSC 211: Object Oriented Programming

## Number Systems, Further look into DataTypes

Marco Alvarez

Department of Computer Science and Statistics
University of Rhode Island

Fall 2019


# Number systems

- A way to represent numbers
  - numbers are expressed in a certain **base**
- Why study number systems in CS?
  - to understand data representation
- Examples of number systems
  - binary
  - decimal
  - octal
  - hexadecimal



# Positional number systems

assuming base **b**:

$$\ldots d_2 b^2 + d_1 b^1 + d_0 b^0 + d_{-1} b^{-1} + d_{-2} b^{-2} \ldots$$

$$43.23 = 4 \cdot 10^1 + 3 \cdot 10^0 + 2 \cdot 10^{-1} + 3 \cdot 10^{-2}$$



# Decimal number system

- Base 10
- Symbols

0 1 2 3 4 5 6 7 8 9

$$456 = 4 \cdot 10^2 + 5 \cdot 10^1 + 6 \cdot 10^0$$

## Binary number system

- Base 2
- Symbols

0 1

Most Significant Bit

Least Significant Bit

$1010 = (1 \cdot 2^3) + (0 \cdot 2^2) + (1 \cdot 2^1) + (0 \cdot 2^0)$



## Binary to Decimal?

1 0 0 1 0 1 0 0 0

1 1 0 . 1 0 1



## Try these ..

1 0 0 1 1 1 0 1

1 1 0 1 0 0 1 1

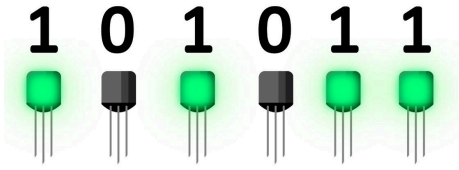
1 1 1 1 1 1 1 1

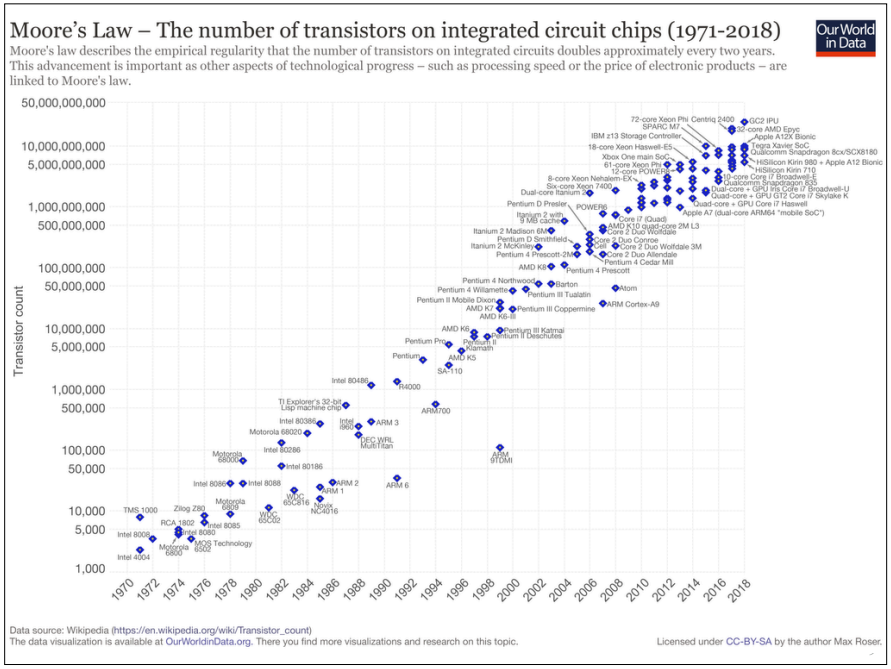
What is a **bit**? What is a **byte**?



## Bits and computers

- A bit can only have two values (states)
  - easy to embed into physical devices
- **Transistor**
  - processors have billions of transistors
  - transistors can be switched **on** and **off**






# Decimal to other bases

- Repeatedly divide by **base**
  - collect remainders
  - output in reverse order

$57_{10}$

- $57 \;/\; 2 = 28 \text{ R } 1$
- $28 \;/\; 2 = 14 \text{ R } 0$
- $14 \;/\; 2 = 7 \text{ R } 0$
- $7 \;/\; 2 = 3 \text{ R } 1$
- $3 \;/\; 2 = 1 \text{ R } 1$
- $1 \;/\; 2 = 0 \text{ R } 1$

$111001_2$



# Hexadecimal number system

- Base 16
- Symbols

0 1 2 3 4 5 6 7 8 9 A B C D E F

$$4A1C = (4 \cdot 16^3) + (10 \cdot 16^2) + (1 \cdot 16^1) + (12 \cdot 16^0)$$



# Hexadecimal to decimal

1 0 5 0 B

A 0 1 0 F

# Binary to hexadecimal

| Hex | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bin | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Dec | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Oct | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |

1 0 0 1 1 1 0 1

1 1 0 1 0 0 1 1

1 1 1 1 1 1 1 1

Humans think in **base 10.** Computers think in **base 2.**
Humans use **base 16** to easily manipulate data in **base 2.**



# Color codes

Shades of yellow color chart

| Color | HTML / CSS Color Name | Hex Code #RRGGBB | Decimal Code (R,G,B) |
|---|---|---|---|
| | lightyellow | #FFFFE0 | rgb(255,255,224) |
| | lemonchiffon | #FFFACD | rgb(255,250,205) |
| | lightgoldenrodyellow | #FAFAD2 | rgb(250,250,210) |
| | papayawhip | #FFEFD5 | rgb(255,239,213) |
| | moccasin | #FFE4B5 | rgb(255,228,181) |
| | peachpuff | #FFDAB9 | rgb(255,218,185) |
| | palegoldenrod | #EEE8AA | rgb(238,232,170) |
| | khaki | #F0E68C | rgb(240,230,140) |
| | darkkhaki | #BDB76B | rgb(189,183,107) |
| | yellow | #FFFF00 | rgb(255,255,0) |
| | olive | #808000 | rgb(128,128,0) |
| | greenyellow | #ADFF2F | rgb(173,255,47) |
| | yellowgreen | #9ACD32 | rgb(154,205,50) |

https://www.rapidtables.com/web/color/Yellow_Color.html

What is the color code of 'greenyellow' in **binary**?



# 31 oct = 25 dec?



# Going back to C++ …

# Integer literals in C++

```
int d = 42;
int o = 052;
int x = 0x2a;
int X = 0X2A;
int b = 0b101010; // C++14
```

- **decimal-literal** is a non-zero decimal digit (1, 2, 3, 4, 5, 6, 7, 8, 9), followed by zero or more decimal digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
- **octal-literal** is the digit zero (0) followed by zero or more octal digits (0, 1, 2, 3, 4, 5, 6, 7)
- **hex-literal** is the character sequence `0x` or the character sequence `0X` followed by one or more hexadecimal digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, A, b, B, c, C, d, D, e, E, f, F)
- **binary-literal** is the character sequence `0b` or the character sequence `0B` followed by one or more binary digits (0, 1)

https://en.cppreference.com/w/cpp/language/integer_literal



DISPLAY 2.2 Some Number Types

| Type Name | Memory Used | Size Range | Precision |
|---|---|---|---|
| *short* (also called *short int*) | 2 bytes | −32,768 to 32,767 | (not applicable) |
| *int* | 4 bytes | −2,147,483,648 to 2,147,483,647 | (not applicable) |
| *long* (also called *long int*) | 4 bytes | −2,147,483,648 to 2,147,483,647 | (not applicable) |
| *float* | 4 bytes | approximately $10^{-38}$ to $10^{38}$ | 7 digits |
| *double* | 8 bytes | approximately $10^{-308}$ to $10^{308}$ | 15 digits |
| *long double* | 10 bytes | approximately $10^{-4932}$ to $10^{4932}$ | 19 digits |

*These are only sample values to give you a general idea of how the types differ. The values for any of these entries may be different on your system.* `Precision` *refers to the number of meaningful digits, including digits in front of the decimal point. The ranges for the types* `float`, `double`, *and* `long double` *are the ranges for positive numbers. Negative numbers have a similar range, but with a negative sign in front of each number.*

from: Problem Solving with C++, 10th Edition, Walter Savitch

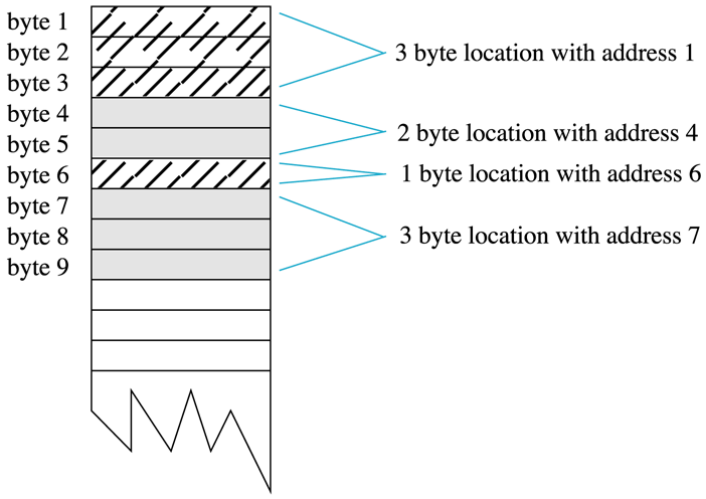


| Type | Size in bits | Format | Value range | |
|---|---|---|---|---|
| | | | Approximate | Exact |
| character | 8 | signed | | **-128** to **127** |
| | | unsigned | | **0** to **255** |
| | 16 | unsigned | | **0** to **65535** |
| | 32 | unsigned | | **0** to **1114111** (**0x10ffff**) |
| integer | 16 | signed | **± 3.27 · $10^4$** | **-32768** to **32767** |
| | | unsigned | **0** to **6.55 · $10^4$** | **0** to **65535** |
| | 32 | signed | **± 2.14 · $10^9$** | **-2,147,483,648** to **2,147,483,647** |
| | | unsigned | **0** to **4.29 · $10^9$** | **0** to **4,294,967,295** |
| | 64 | signed | **± 9.22 · $10^{18}$** | **-9,223,372,036,854,775,808** to **9,223,372,036,854,775,807** |
| | | unsigned | **0** to **1.84 · $10^{19}$** | **0** to **18,446,744,073,709,551,615** |
| floating point | 32 | IEEE-754 | • min subnormal: **± 1.401,298,4 · $10^{-45}$**<br>• min normal: **± 1.175,494,3 · $10^{-38}$**<br>• max: **± 3.402,823,4 · $10^{38}$** | • min subnormal: **±0x1p-149**<br>• min normal: **±0x1p-126**<br>• max: **±0x1.fffffep+127** |
| | 64 | IEEE-754 | • min subnormal: **± 4.940,656,458,412 · $10^{-324}$**<br>• min normal: **± 2.225,073,858,507,201,4 · $10^{-308}$**<br>• max: **± 1.797,693,134,862,315,7 · $10^{308}$** | • min subnormal: **±0x1p-1074**<br>• min normal: **±0x1p-1022**<br>• max: **±0x1.fffffffffffffp+1023** |

https://en.cppreference.com/w/cpp/language/types



## Memory Locations and Bytes




from: Problem Solving with C++, 10th Edition, Walter Savitch