# Lecture 2:
# Binary and Strings

CSE 29: Systems Programming and Software Tools

Aaron Schulman (Shalev)

# Lecture 2 Overview

- How to access strings and arrays in C
- How does binary number representation work
- String functions and how things can go wrong

# Week 1 Announcements

- **Announcements**:
  - Lab attendance is **required** tomorrow, and a lot happens there.
  - Make sure to go to lab
  - Submit the welcome survey before lab on Tuesday of week 1

# Course Website / Syllabus

**https://ucsd-cse29.github.io/fa25/**

# Demo: Accessing Binary Data in C

- **Big Lesson:**
  - Programs can interpret and present bits (in arrays) **in many ways:**
    - » They can be **Integers** printf("%d", arr[i])
    - » They can be **Characters** printf("%c",arr[i])
    - » They can be **Strings** of Characters printf("%s",arr)
    - » There are many other ways to interpret arrays of bits too!
  - **C gives us direct access to the bit representations of data in memory!**

# Why is 'H' or 72 = bin 1001000 ?


$$\mathbf{1\ 0\ 0\ 1\ 0\ 0\ 0} = 1x2^6 + 1x2^3$$
$$= 64 + 8 = 72$$

Binary (Base 2) is *similar* to Decimal (Base 10) representation:

$$\mathbf{7\ 2} = 7x10^1 + 2x10^0$$
$$= 70 + 2 = 72$$

# Converting integers to/from binary arrays


**1 bit**

**0** =0

**1** =1

**2 values: $2^1$**

**2 bits**

| | | |
|---|---|---|
| **0** | **0** | =0 |
| 0 | 1 | =1 |
| 1 | 0 | =2 |
| 1 | 1 | =3 |

**4 values: $2^2$**

**3 bits**

| | | | |
|---|---|---|---|
| **0** | **0** | **0** | =0 |
| 0 | 0 | 1 | =1 |
| 0 | 1 | 0 | =2 |
| 0 | 1 | 1 | =3 |
| 1 | 0 | 0 | =4 |
| 1 | 0 | 1 | =5 |
| 1 | 1 | 0 | =6 |
| 1 | 1 | 1 | =7 |

**8 values: $2^3$**

# What does each index of a bit array mean?

**1 byte**

| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|
| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |

**MSB** **LSB**

# Bit array -> Integer conversions

$$\mathbf{1\ 0\ 1\ 0} = 1x2^3 + 0x2^2 + 1x2^1 + 0x2^0$$
$$= 8 + 0 + 2 + 0 = 10$$

$$\mathbf{1\ 1\ 1\ 0} = 1x2^3 + 1x2^2 + 1x2^1 + 0x2^0$$
$$= 8 + 4 + 2 + 0 = 14$$

# Demo: How do strings work in C?

- Big Idea:
  - Strings are just arrays of characters
  - The string is terminated when there is a null character at the end

# What is a string, an array of characters!

**char sup[7] = “Hello”;**

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| **‘H’** | **‘e’** | **‘l’** | **’l’** | **’o’** | **‘ ’** | **‘\0’** |

Special char.
**NUL Character**
End of string = **0**

# Demo: What if we forget the NULL Char?


- Big Idea:
  - There are no training wheels anymore in C, this is not Java
  - If you tell the computer to do something, it will do exactly what you say.

# What happens if the null is not there?

```
char sup[7] = "Hello";
char hi[2] = {'H', 'i};
```

| 0 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| **'H'** | **'i'** | **'H'** | **'e'** | **'l'** | **'l'** | **'o'** | **' '** | **'\0'** |