Endianness:
One major difference between ISAs is how multi-byte characters are stored. Knowing that `sizeof(int) == 4`, what do we expect from the following program?

05-endian.c

```
4  int i = 3 + (2 << 8) + (1 << 16);  // 66051
5  char *s = (char *)&i;
6  printf("%02x %02x %02x %02x\n", s[0], s[1], s[2], s[3]);
```

Big Endian:

s[0]	s[1]	s[2]	s[3]

Little Endian:

s[0]	s[1]	s[2]	s[3]

Conversation Between Host/Network Order:

05-htonl.c

```
4  int main() {
5    // By default, a variable is in "host order":
6    uint32_t value = 66051;
7    printf("value: 0x%08x; %%d == %d\n", value, value);
8
9    // htonl coverts a "host order" value to "network order" (even
10 if it might do nothing on some systems):
11   uint32_t network_value = htonl(value);
12   printf("htonl: 0x%08x; %%d == %d\n", network_value,
     network_value);
13
14   // ntohl converts a "network order" value to "host order" (even
15 if it might do nothing on some systems):
16   uint32_t host_value = ntohl(network_value);
17   printf("ntohl: 0x%08x; %%d == %d\n", host_value, host_value);
18 }
```

Beyond Characters: Files and File Types
Using binary digits, often represented as characters using an encoding like UTF-8, we can build more complex file types.

File Extensions: An Easy Identifier
The most common way to identify the contents of a file is by the **file extension**. The file extension is defined as:

Examples:

cs340.png	mp1.c	mp1.h	taylor.swift.mp4

Which files are "plain text files"?

How do we detect non plain text files?

Memory Hierarchy:
The third foundation of a computer system is the "memory" -- the storage of data to be processed by our CPU. There are many different types of common **memory** and **storage** in a system:

1.

2.

3.

4.

5.

6.

Sample Programs:

05-col.c

```
16  for (unsigned int c = 0; c < SIZE; c++) {
17    for (unsigned int r = 0; r < SIZE; r++) {
18      array[(r * SIZE) + c] = (r * SIZE) + c;
19    }
20  }
```

-vs-

05-row.c

```
16  for (unsigned int r = 0; r < SIZE; r++) {
17    for (unsigned int c = 0; c < SIZE; c++) {
18      array[(r * SIZE) + c] = (r * SIZE) + c;
19    }
20  }
```

...what is different about **05-col.c** and **05-row.c**?

Running Times: **05-col.c**:

05-row.c:

Key Idea: Locality of Reference

In working with memory in any computer system, we want to access it as quickly as possible. However, space is extremely limited in the fastest memory, so we need strategies on what data to keep close.

General Purpose Memory:

- CPU Registers:
- CPU Cache (i9-13900K, Released Q4'22):
- RAM:

System Memory:

1. **[Limited]**:
2. **[Shared]**:
3. **[Simple]**:

To help us to begin to organize this RAM, we divide the RAM up into chunks called _____.

On Linux, find the size of a page:

```
#  getconf PAGESIZE
```

...on most modern systems, a page is _____ KB.

Virtual Memory:
Modern systems provide an abstraction between _____ and _____:

1. A _____ translates a _____ into a **physical address**. *It's just a pointer!*
2. Every memory address is made up of the _____ and the _____.
3. Virtual Memory is **NOT shared** between processes/apps.
4. **EVERY memory address** _____ is a virtual memory address!!

Virtual Memory Example:

P1 Page Table:

[0]:
[1]:
[2]:
[3]:
[4]:
[5]:
[6]:
[7]:
[8]:
[9]:
[10]:
[11]:
[12]:
[13]:
[14]:
[15]:

RAM:

[0]:
[1]:
[2]:
[3]:
[4]:
[5]:
[6]:
[7]:
[8]:
[9]:
[10]:
[11]:
[12]:
[13]:
[14]:
[15]:

P2 Page Table:

[0]:
[1]:
[2]:
[3]:
[4]:
[5]:
[6]:
[7]:
[8]:
[9]:
[10]:
[11]:
[12]:
[13]:
[14]:
[15]:

P3 Page Table:

[0]:
[1]:
[2]:
[3]:
[4]:
[5]:
[6]:
[7]:
[8]:
[9]:
[10]:
[11]:
[12]:
[13]:
[14]:
[15]:

OS Logs:

```
P1: 3 pages (a)
P3: 5 pages (b)
P1: 2 pages (c)
P3 exits
P2: 4 pages (d)
P2: 5 pages (e)
P1:
  Extend a to
  5 pages
  (ex: realloc)
```