# File and Network I/O

## CS 2022: Introduction to C

Instructor: Hussam Abu-Libdeh

Cornell University
(based on slides by Saikat Guha)

Fall 2009, Lecture 9

# Input and Output

- Keyboard I/O
- Disk I/O
- Network I/O

# Streams

- In many programming languages, input/output are done in streams
- Data exists on the stream, you consume part of it and move on
- Examples:
  - `stdout`: standard output stream
  - `stderr`: standard error output stream
  - `stdin`: standard input stream
  - files
  - network sockets (network connections)

# Output to Terminal

- Write a line to `stdout`
  - `puts("hello world");`
- Write a formatted line to `stdout`
  - `printf("Borat says: Hi %s!\n", i);`
- Can write to streams other than `stdout`
  - `fputs("an error message", stderr);`
    `fprintf(stderr, "Error on value %d\n", i);`

# Input from User (Keyboard)

## Reading till end of line

```
char buf[128];
fgets(buf, 128, stdin);
```

## Reading formatted input

```
int i, j;
char buf[128];
scanf("%d %d %s", &i, &j, buf);
```

# File I/O

## The C standard library way

- Use `fopen`/`fclose`
- Deals with *streams*

## The POSIX way

- Use `open`/`close`
- Deals with *file descriptors*

We will only discuss POSIX I/O here. To read more about C streams, check out the `man` pages and/or http://www.cs.cf.ac.uk/Dave/C/CE.html

# File I/O

## Opening and closing files

```
int fd;                          // File Descriptor
fd = open("/path/to/file", O_RDWR | O_CREAT);
close(fd);
```

## Reading and Writing

```
char buf[4096]; int len;
len = read(fd, buf, 4096)
len = write(fd, buf, 4096);
```

WARNING: Size passed is only a **suggestion**. May read/write fewer than requested number of bytes. Return value is number of bytes actually read/written. MUST retry if not fully read/written.

# File I/O

- `lseek(fd, numbytes, SEEK_CUR);`
  Seek numbytes from from current location.
- `sync();`
  Ensure bytes hit the disk. Not needed for the most part.
- `FILE *ffd = fdopen(fd, "r");`
  Construct a stream from file-descriptor.
- `fprintf(ffd, "format", args);`
  Write formatted text output to file.
- `fscanf(ffd, "format", args);`
  Read formatted input from the file.
- `fclose(ffd);`
  Close a stream.

# Network I/O

## Opening and closing network sockets

```
int sock;                              // File Descriptor
sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
close(sock);
```

## Internet Addresses

```
struct sockaddr_in addr;
addr.sin_family = AF_INET;
addr.sin_addr.s_addr = htonl(0x7F000001);
addr.sin_port = htons(8080);
```

Fill the address info manually or get the info automatically with `getaddrinfo()`.
See `man getaddrinfo`

# Network I/O

## Server

```
srv = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
err = bind(srv, (struct sockaddr *)&addr, sizeof(addr));
if (err) ...
err = listen(srv, 5);
if (err) ...
cli = accept(srv, NULL, 0);
```

## Client

```
cli = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
err = connect(cli, (struct sockaddr *)&addr, sizeof(addr));
if (err) ...
```

Read/Write data just as you would with file-descriptors.

# Very Helpful Resources

## Network Programming

Beej's guide to network programming:
http://beej.us/guide/bgnet/

## General C Programming

Dave's programming C tutorials:
http://www.cs.cf.ac.uk/Dave/C/CE.html