

CSC209 Summer 2015 — Software Tools and Systems Programming

www.cdf.toronto.edu/~csc209h/summer/

Peter McCormick
pdm@cs.toronto.edu

Agenda

- What is systems programming?
- Course administration
- Exploring a system with the shell

What is *systems*
programming?

A View of the System Stack

Your Python Code

Python Libraries

CPython (compiler & VM)

C Standard Library

Operating System Kernel

Device Drivers

Hardware (CPU & Peripherals)

A View of the System Stack

Your Python Code

Python Libraries

CPython (compiler & VM)

C Standard Library

Operating System Kernel

Device Drivers

Hardware (CPU & Peripherals)

A View of the System Stack

Userspace (CSC209)

Kernel (CSC369)



Running Python Code

```
wolf:~$ cat hello.py  
print("Hello World\n")
```

```
wolf:~$ python3 hello.py  
Hello World!
```

← **Execute CPython**

Running Java Code

```
wolf:~$ cat HelloWorld.java
```

```
class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World!"); } }
```

```
wolf:~$ javac HelloWorld.java
```

 **Step 1 - Execute compiler**

```
wolf:~$ ls -l HelloWorld.class
```

```
-rw----- 1 pdm instrs 426 May 14 12:21 HelloWorld.class
```

```
wolf:~$ java HelloWorld
```

```
Hello World!
```

 **Step 2 - Execute JVM**

Running C Code

```
wolf:~$ cat hello.c
#include <stdio.h>
int main() {
    printf("Hello World!\n");
    return 0;
}
```

```
wolf:~$ gcc hello.c
```

← **Step 1 - Execute compiler**

```
wolf:~$ ls -l a.out
```

```
-rwx----- 1 pdm instrs 8377 May 14 12:39 a.out
```

```
wolf:~$ ./a.out
```

```
Hello World!
```

← **Step 2 - Execute *the program***



Image courtesy of Bell Labs/Lucent Technologies

Comparison — Then and Now

	PDP-7	iPhone 6	Difference
Year	1965	2014	<i>~½ century</i>
CPU	0.25 MHz	2x 1.4 GHz (~2800 MHz)	~10,000x
GPU	—	77 GFLOPs	
RAM	9-144 KB	1 GB (one million KB)	~7,000-11,000x
Storage	~325 KB (tape)	16-64 GB (flash)	50,000-200,000x

Ken Thompson
(invented Unix)

Dennis Ritchie
(invented C)



Image courtesy of Bell Labs/Lucent Technologies

Systems Programming

- When performance is a *feature*
- When resources constrained
- Low level — “close to the metal”
- Examples: operating systems, file & web servers, network, graphics, embedded systems, small devices

Software Tools

- Unix shell: Bash
- Documentation tools: man (manual pages aka *manpages*)
- Build tools: Make
- Debuggers: gdb
- Editors: vi/vim, emacs, many others...
- Power user system utilities: strace

Course Admin

Email — pdm@cs.toronto.edu

- Include your name
- Include “CSC209” in the subject
- Use formal language:
 - Proper English, not slang
 - Provide context, and clearly state your question
 - *Include* your name and CDF username (student #'s are less useful)

Website

<http://www.cdf.toronto.edu/~csc209h/summer/>

- Complete course information sheet & syllabus
- Lecture materials
- Assignments & labs
- Links to Piazza and MarkUs

- **Lectures**

- Thursdays 6-8pm

- **Labs**

- Thursday 8-9pm in Bahen labs

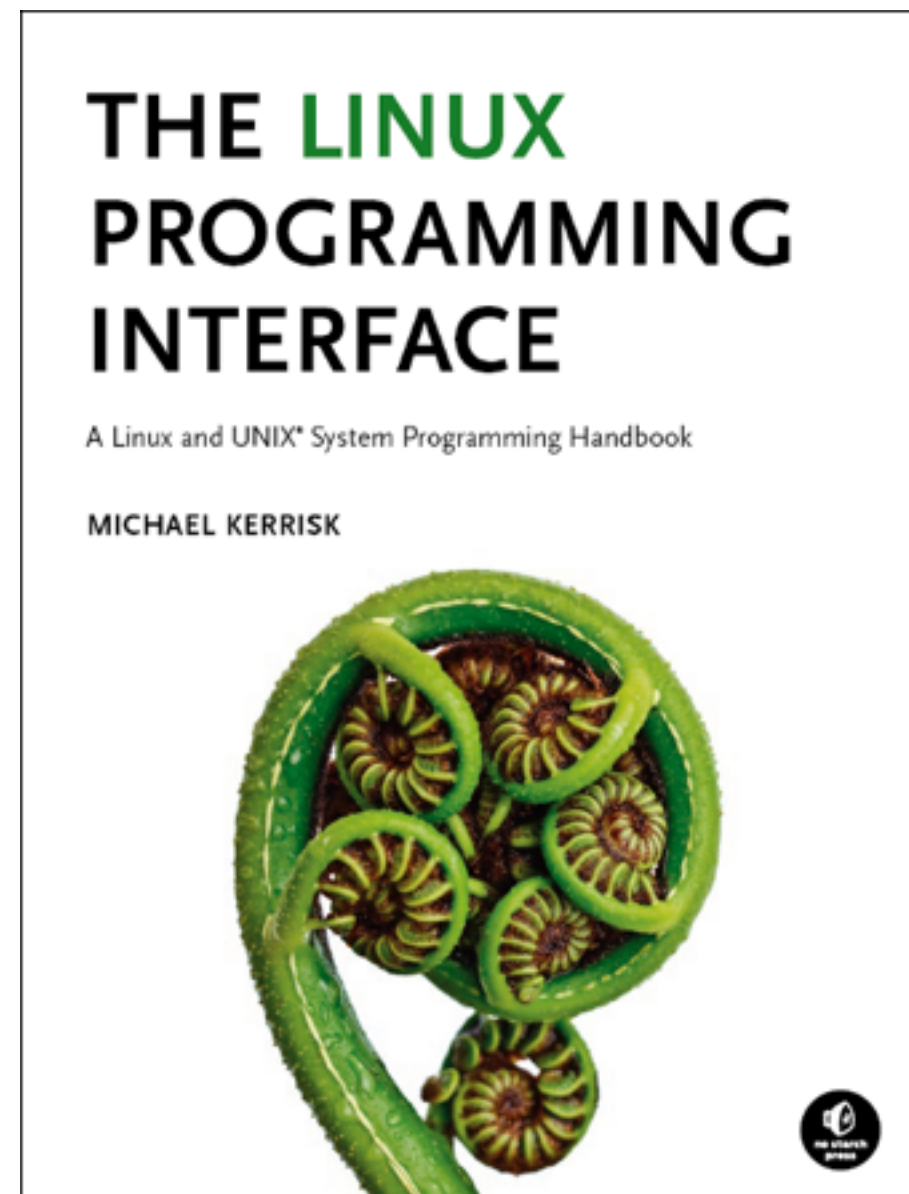
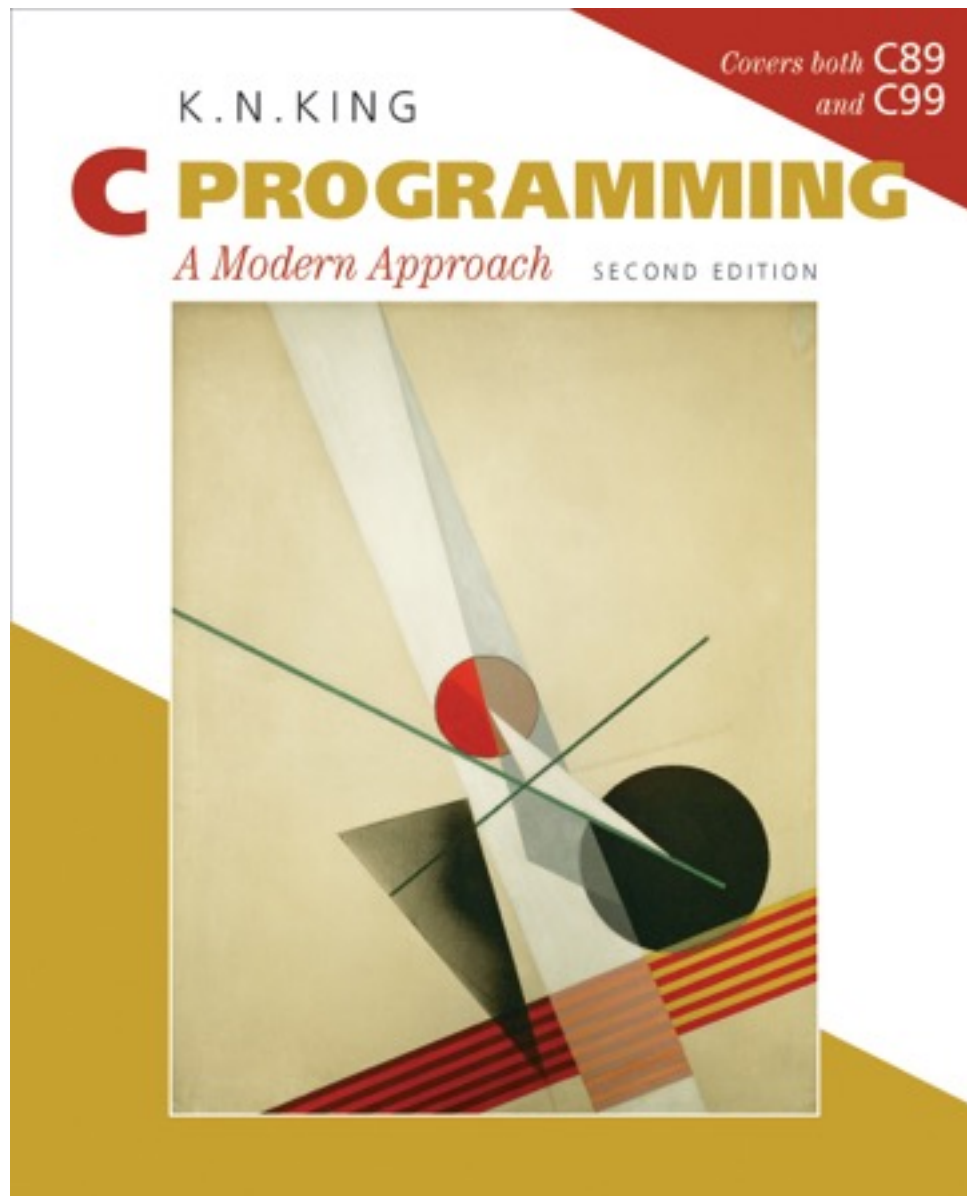
- **Office Hours**

- Tuesdays 2-4pm in BA3201

Midterm

- **Week of June 22-26, 2015**
- Exact date and time will be announced as soon as I know

Texts



Assignments

- A1: Basic C
- A2: Programming with pointers (using C)
- A3: Fork and pipes (using C)
- A4: Processes and communications (using C)
- All code ***must*** work on the CDF server to receive full marks
 - Don't leave it to the last minute to find out you have a problem
- Code that does not compile will receive **0**.

**If it doesn't compile on
CDF, your assignment
mark will be *zero* (0)**

Labs

- 10 labs this term, each lab exercise is worth 0.4% of your overall mark
- Exercises will be posted towards the beginning of the week
- Can be done in pairs or alone
- We will be using PCRS and/or MarkUs

Labs (2)

- Weekly lab sessions in Bahen with TA's to help you (Thursday 8-9pm)
 - Room assignments will be posted soon
 - No lab this week!
- You can work on them remotely too
- Labs are due on Friday at 10pm

Optional week 1 exercise on the shell has been posted; try it out!

Submitting Assignments

- We will be using MarkUs
- You will be provided with a Subversion (SVN) repository to commit your assignment code to
- Committing is submitting
 - No commit means no submission
 - Commit early, commit often, and *test on CDF*

Piazza

- Official source for course announcements
- Official discussion board
- If you have questions (that are not personal/specific to you), post them to Piazza

Anonymous Feedback

- You can provide me feedback completely anonymously:
 - <http://www.cdf.toronto.edu/~csc209h/summer/feedback.html>
- If you would like a response, please do provide some means of responding
- I would welcome your comments on the lectures, assignments, labs, tests or anything else relevant to the course

say209

- You can provide me with in-class questions/comments
- Simply log into CDF and run:

```
~csc209h/summer/pub/bin/say209 "Hello CSC209!"
```

Consider modifying your PATH environment variable on CDF by adding this to your `~/ .bashrc` (assuming you are using Bash):

```
export PATH=/u/csc209h/summer/pub/bin/:$PATH
```

Then, log out and back in again. Or, run `source ~/ .bashrc`

Now you can omit the full path and simply `say209 "Hi"`

Git Clone the Course Website

(completely optional)

<https://github.com/pdmccormick/csc209-summer-2015>

Plagiarism

“The work you submit must be your own, done without participation by others. It is an academic offence to hand in anything written by someone else without acknowledgement.”

Plagiarism

You are not helping your friend when you give him or her a copy of your assignment.

You are hurting your friend when you ask him or her to give you a copy of their assignment.

What counts as cheating?

- Cheating is:
 - Copying parts or all of another student's assignment
 - Including code from books, web sites, other courses without attribution
 - Getting someone else to do substantial parts of your assignment
 - Giving someone else your solution
- Cheating is *not*:
 - Helping to find a bug in a friend's code (but tread careful)
 - Helping each other understand man pages or example code.

Respect

1. Respect for yourself
2. Respect for your peers
3. Respect for your community

Use of Laptops and Devices in Lecture

- *Following along* with course material is encouraged
- If engaged in other activities, please show respect to your fellow students and sit towards rear of the lecture room

Self Study Topics

- Using Subversion or Git (git-svn)
- Using Linux systems
- Learn a text editor you can use over SSH: vi, emacs, joe, nano
- Learn a debugger: gdb, ddd, or in an IDE
- Setup your own Linux virtual machine
- Extra readings

What platform do you use?

Windows Users

- Use PuTTY to remotely connect over SSH to cdf.toronto.edu
- Access CDF help site
- Dive into Linux by installing a distribution in a virtual machine use Oracle VirtualBox
 - CDF runs Ubuntu 12.04 LTS
- Everyone can use the CDF Help Centre!

Questions about the
course?

Phrases you'll hear me say:

“The *machine*”

“Open the black box”

“The Unix philosophy”

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs that handle text streams, because text is a universal interface.

Unix *vs* Linux

Exploring a system
through the shell

Next Week

- Introduction to C!
- First lab will be held
 - Rooms will be announced on the website
 - Exercise will be posted
- Assignment 1 will be posted (and MarkUs accounts will be created)

Extra Slides

Courtesy of Karen Reid and used with permission and gratitude!

Files and File Systems

- What is a file?
 - A sequence of bytes
- A file system is the organization of files
 - hierarchy of directories (folders)
 - notion of current working directory (location in the file system)
 - access control

Files and Directories

Kerrisk
Ch 14

- “Everything is a file.”
- Unix provides a file interface for all Input/Output.
 - regular files
 - directories
 - devices
 - video (block)
 - keyboard (character)
 - sound (audio)
 - network (block)
- File interface = open, read, write, close



Try `ls -l /dev` and look
at the permissions string.

`crw-----`

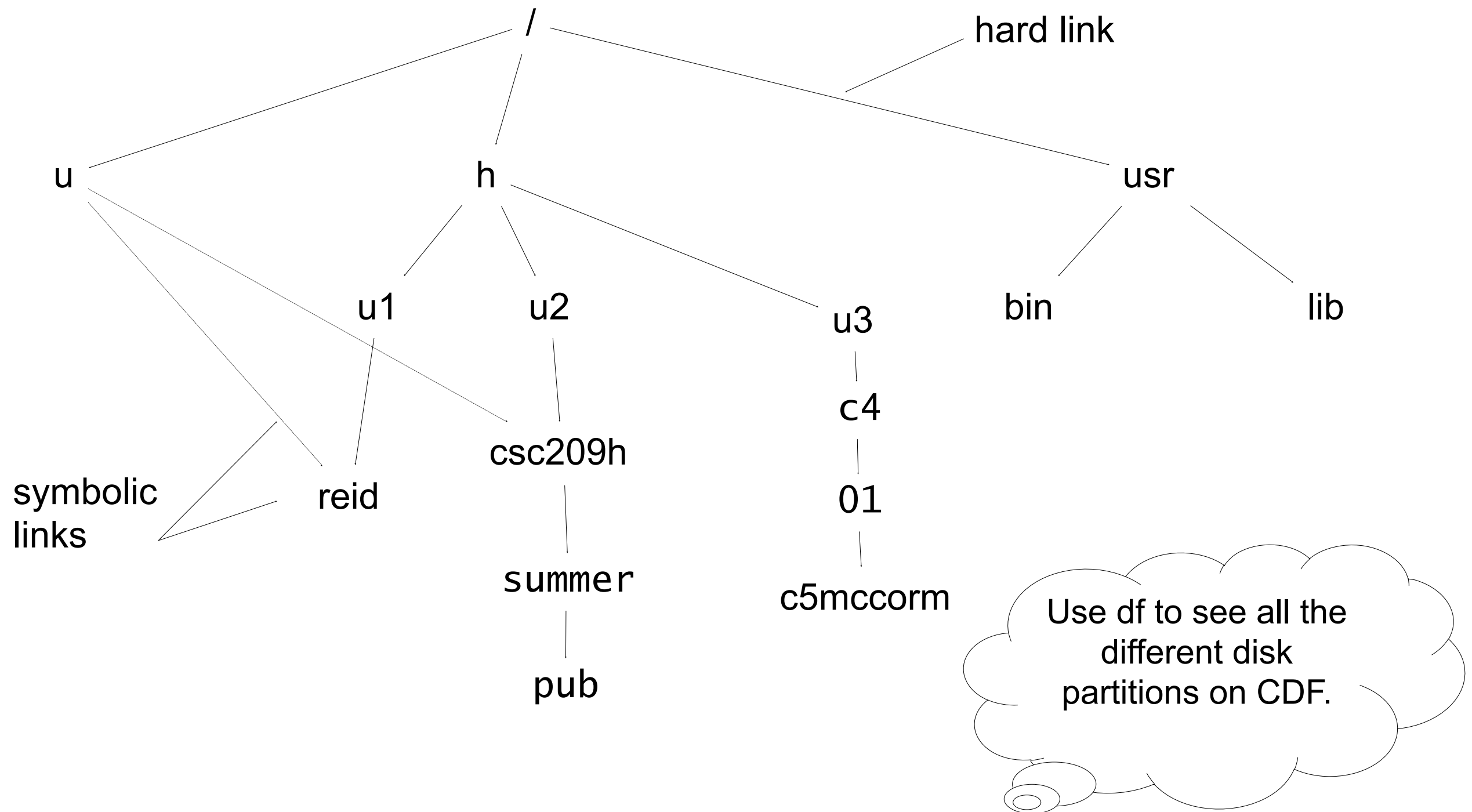
`brw-----`

c = character, b = block

File System Hierarchy

- Everything starts in the “**root**” directory whose name is “/”
- A **directory** is a file that contains directory entries.
- A directory entry maps a file name to an **inode**.
- An inode is the data structure that contains information about a file, including which disk blocks contain the file data.

File System Hierarchy



File Systems and Links

- One file system per disk partition.
- A file system can be **mounted** at any point in the directory tree of another file system.
- An entry in a directory file which specifies an inode is a **hard link**.
- There can be several hard links to a file, but hard links cannot cross file systems.
- A **soft link** (symbolic link) is a small file containing the path name of the linked file or directory.
- Soft links work across file systems.

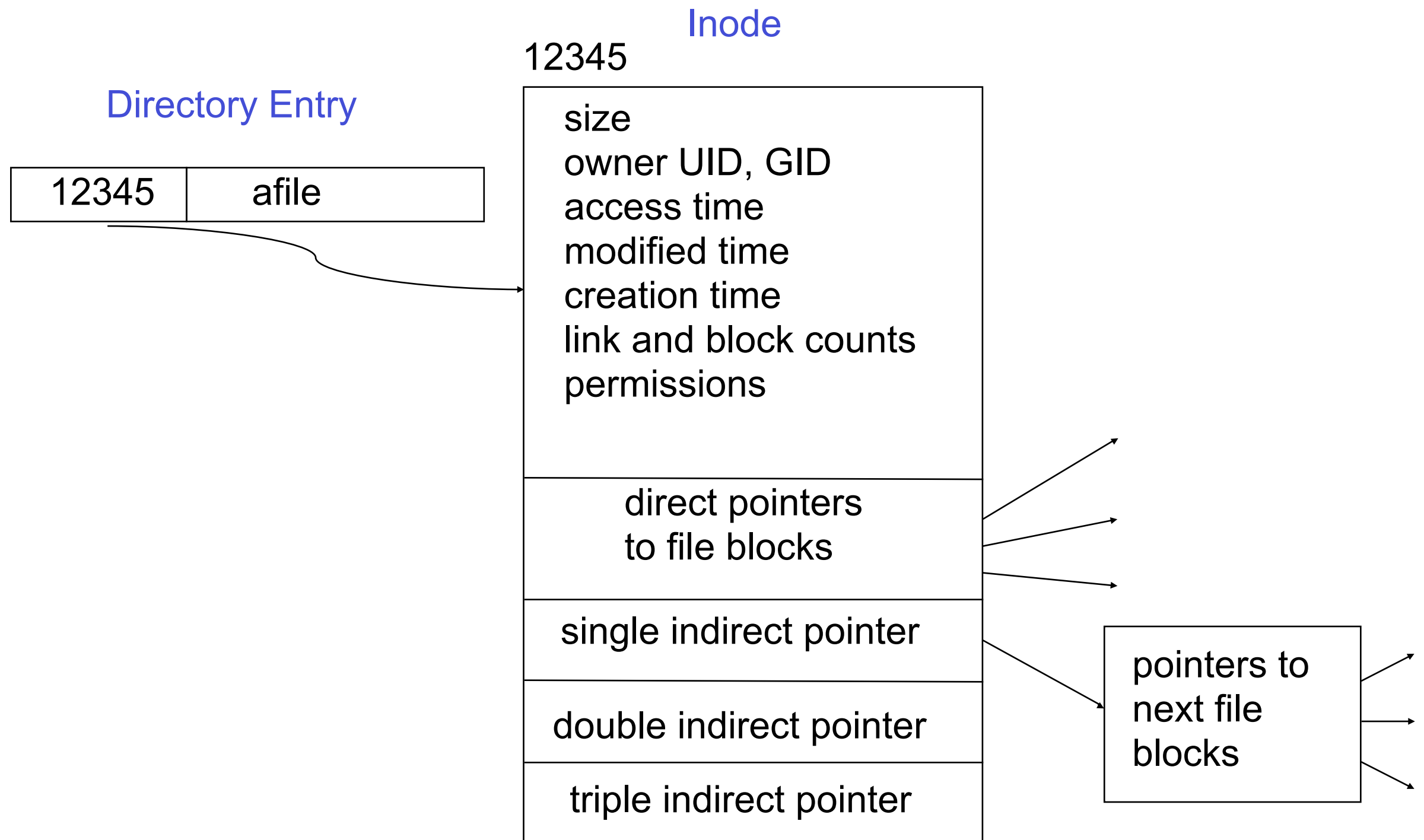
Directories and Links

directory file

2	.
2	..
14	u
46505	home
139412	cdrom
201345	lib

```
% ls -l /
drwxr-xr-x  2 root  root  4096 Nov  8 17:56 bin/
drwxr-xr-x  2 root  root  4096 Aug 10 14:46 cdrom/
drwxrwsr-x  2 root  staff 4096 Feb  8 2002 home/
drwxr-xr-x  6 root  root  4096 Sep  2 15:26 lib/
lrwx----- 1 root  root      6 Sep  2 15:32 u -> /cdf/u/
```

Inodes and Directory Entries

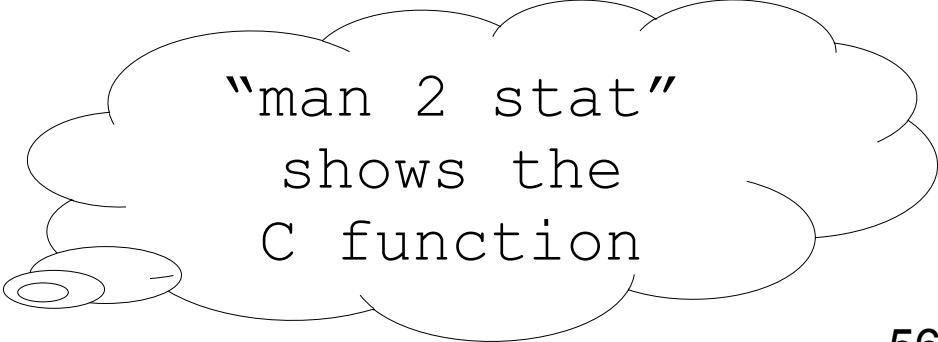


Stat

Kerrisk
Ch 15.1

```
greywolf% stat csc209h
```

```
File: `csc209h'
Size: 512          Blocks: 2          IO
Block: 8192    directory
Device: 16h/22d Inode: 27612          Links: 7
Access: (0755/drwxr-xr-x)  Uid: (    0/
root)   Gid: (  517/ csc209h)
Access: 2010-01-06 11:32:44.293409000 -0500
Modify: 2010-01-04 12:06:15.987312000 -0500
Change: 2010-01-04 12:06:15.987312000 -0500
```



"man 2 stat"
shows the
C function

user group other
-rwxr-xr-x

Permissions

Kerrisk
Ch. 2

```
-rwxr-xr-x 1 reid 0 Jan 6 11:35 allexec*  
-r--r--r-- 1 reid 0 Jan 6 11:35 allread  
dr-xr-xr-x 2 reid 512 Jan 6 11:36 dir-read/  
dr-x--x--x 2 reid 512 Jan 6 11:36 dir-search/  
-rw----- 1 reid 0 Jan 6 11:35 ownerread  
-r--r--r-- 1 reid 0 Jan 6 11:35 readonly
```

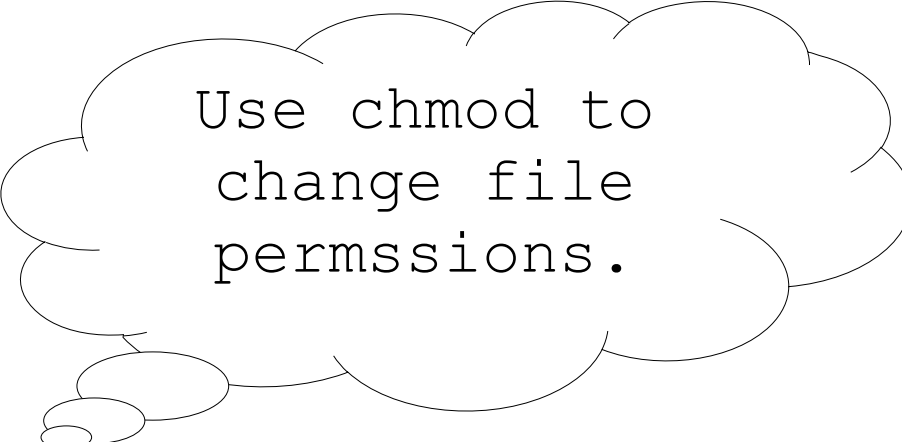
- File permissions
 - read, write, execute – pretty much what you think
- Directory permissions
 - read – you can run ls on the directory
 - write – you can create and delete files in the directory
 - execute – you can “pass through” the directory when searching subdirectories.

Example

```
dr-xr-xr-x 2 reid 512 Jan 6 11:36 dir-read/  
d--x--x--x 2 reid 512 Jan 6 11:43 dir-search/  
---x--x--x 1 reid 6921 Jan 6 11:42 dir-search/cprog*  
-r-xr-xr-x 1 reid 47 Jan 6 11:41 dir-search/shellprog*
```

What is the result of the following:

```
$ ls dir-read  
$ read-only  
$ ls dir-search  
$ dir-search/cprog  
$ cd dir-search  
$ shellprog
```



Use chmod to
change file
permissions.

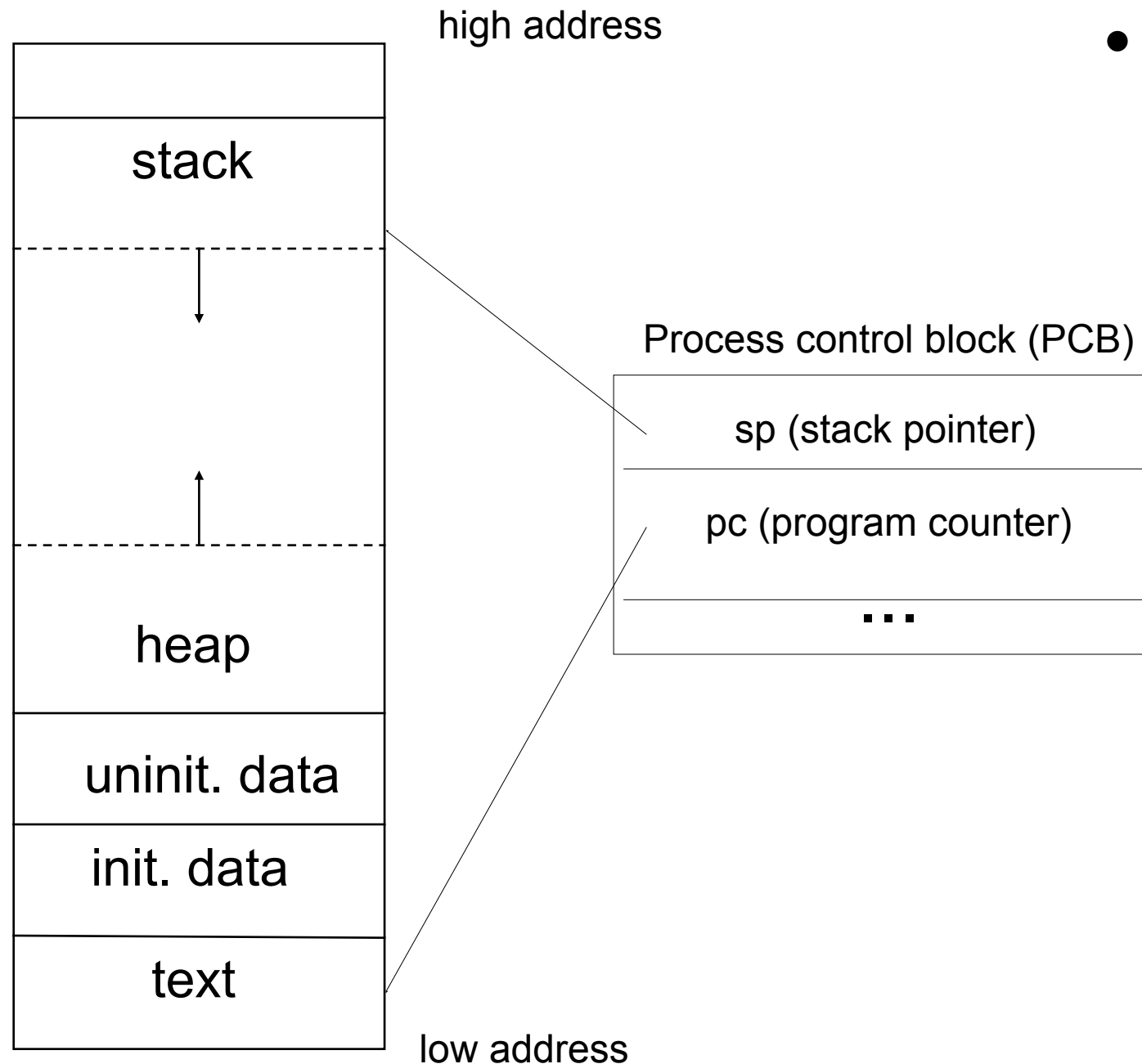
Extra slides

Processes

- A **process** is an executing instance of a program.
- The OS keeps track of information about the process.
 - process ID – a unique non-negative integer
 - process state – “running”, “ready”, “blocked”
 - program counter – which instruction is being executed.
 - a list of open files
 - etc.

Object Files/Executables

- Typical memory layout of programs.



Unix History

- Inspired by Ken Thompson to play Space Travel on his DEC PDP-7 in 1969.
- Thompson wrote the first version of Unix in assembler in one month.
- Dennis Ritchie and Ken Thompson ported an enhanced version to a PDP-11/20 in 1970.
- Ritchie and Rudd Canaday ported a cut down version of the BCPL language to Unix, calling it B.
- The PDP-11 was purchased for text processing.
- The first user was Bell's Patent Department.
- Pipes and C were added in 1971-73

More Unix History

- BTL Lawyers, “License to universities, but no support.”
- This led to extensive sharing.
- University of Toronto on the first mailing list in 1975.
- Software Tools User Group formed in 1978.
- Canadian connection!
 - Bill Reeves, Brian Kernighan, Rob Pike...
- Berkeley Software Distribution grew out of collecting and distributing bug fixes. (Led to FreeBSD, NetBSD)
- Bill Joy started at Berkeley but joined the startup Sun Microsystems in 1982.
- 1991, Linus Torvalds posts a note describing his experimental OS modeled on minix.

Why Unix?

- Available on a number of platforms.
- Multi-user, multi-programmed.
- Shares computer resources sensibly.
- Permits manipulation of files, processes, and programs.
- Allows inter-process and inter-machine communication.
- Permits access to its operating features.