CSC209: Software Tools and Systems Programming

Richard Krueger

Email : krueger@cs.utoronto.ca

Office hours: BA 3234

Administrivia

- Email: krueger@cs.utoronto.ca
  - Email must include your name.
  - Please set up your mail program to use plain text, not html.
  - Email is a formal method of communication:
    - Use proper English.
    - State your question clearly, with enough context.
    - Sign it.

More on email

- Not helpful
  - *I used the makefile you gave us, but my program doesn't compile. What could be wrong?"*
  - *My program gets a seg fault error message, but I don't know why.*
- Much better
  - *When I compile my program, I get the following error message. It seems to indicate that there is a problem with the following lines of code. (cut and paste error messages and code.)*
  - *The debugger tells me that the seg fault I'm getting is on the following line. I don't see what the problem is with this line. (file included below)*

Course Information

- Check the course information sheet (handed out and on the course web page) for
  - Office hours
  - Contact information
  - Assignment schedule
- The course web page is the official source of announcements.

  http://www.cs.utoronto.ca/~csc209h/
- Make sure you have the prerequisites!

Assignments

- A1: Shell programming (Bourne shell)
- A2: Manipulating files and directories (in C)
- A3: Processes (in C)
- A4: Sockets (in C)
- The assignments are best done over a couple of weeks, a few hours at a time.
- All code **must** work on the CDF servers to receive full marks.
- *Don't wait until the last day!*

Submitting Assignments

- You will be using CVS to manage and submit your assignments.
- The repositories will be set up this week.
- You should start learning how to use it as soon as possible.
- *Do not wait until the last minute to try to commit your assignment for the first time.*
- This week's tutorial will cover using CVS.
- See web page for tutorial information.

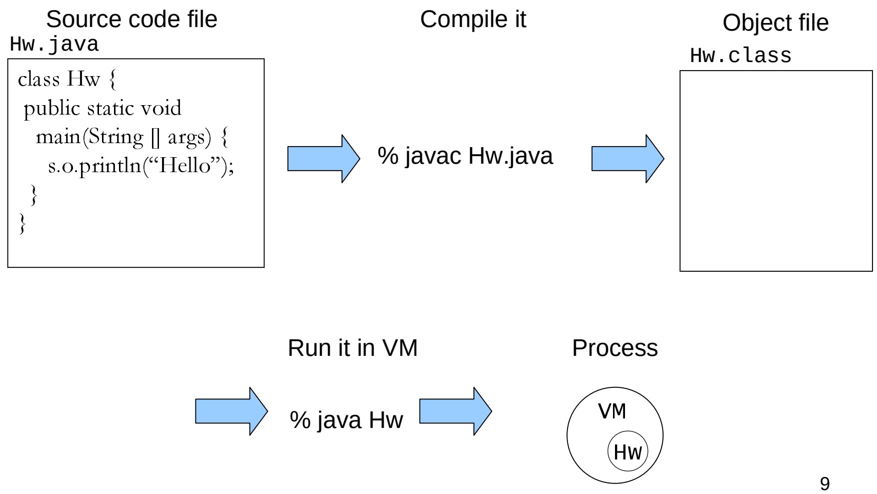
Plagiarism

- "The work you submit must be your own, done without participation by others. It is an academic offense to hand in anything written by someone else without acknowledgement."
- 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 is 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 (be careful)
  - helping each other understand man pages or example code.

The Big Picture (in Java)

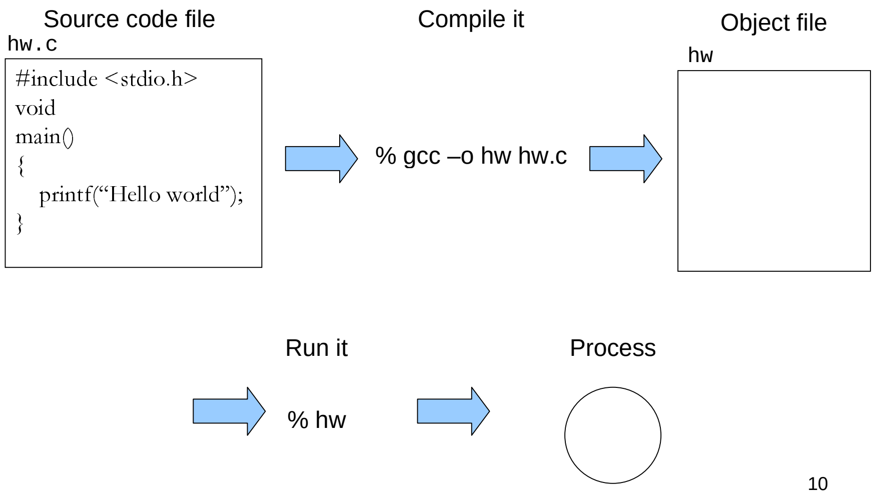


```
class Hw {
 public static void
   main(String [] args) {
     s.o.println("Hello");
  }
}
```

```
% javac Hw.java
```

```
% java Hw
```

The Big Picture



```
#include <stdio.h>
void
main()
{
    printf("Hello world");
}
```

```
% gcc –o hw hw.c
```

```
% hw
```

Source Code Files

hw.c

```
#include <stdio.h>
void
main()
{
    printf("Hello world");
}
```

- What is a file?
- How does the system know where to find `hw.c`?
- What is the meaning of `#include<stdio.h>` ?
- What does `printf` really do?

Compiling a program

```
% gcc –o hw hw.c
```

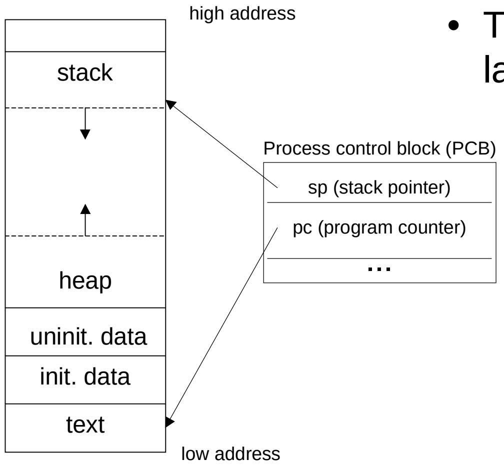
- A compiler is a program translates source code into object (machine) code.
- Here we are running the compiler at the command line.
- A shell is a program that can execute another program.

Shells

```
% gcc –o hw hw.c
```

- The % is a shell prompt.
- Shells
  - accept commands (programs) as input
  - finds the executable
  - interprets the arguments
  - starts executing the command
- Shells also have some “built-in” commands.

Object Files/Executables



- Typical memory layout of programs.

Running a program

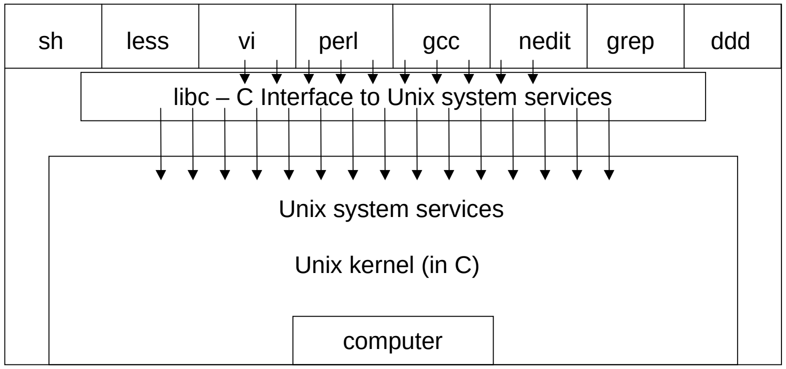
```
% gcc –o hw hw.c

% hw
```

- load a program into memory and hand it off to the OS to run the program.

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.

A different big picture



sh | less | vi | perl | gcc | nedit | grep | ddd

libc – C Interface to Unix system services

Unix system services

Unix kernel (in C)

computer

Course Overview

- Software Tools
  - – Understanding the shell
  - – Shell programming
- Systems Programming
  - – C
  - – files
  - – processes
  - – concurrency
  - – communication

Self Study Topics

- Using CVS - some tutorial coverage
- Using Unix - some tutorial coverage
- Learning an editor – nedit, vi, emacs
- Learning a debugger – ddd is the easiest
- Readings

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.

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 that is a universal interface.

Files and Directories

- “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.

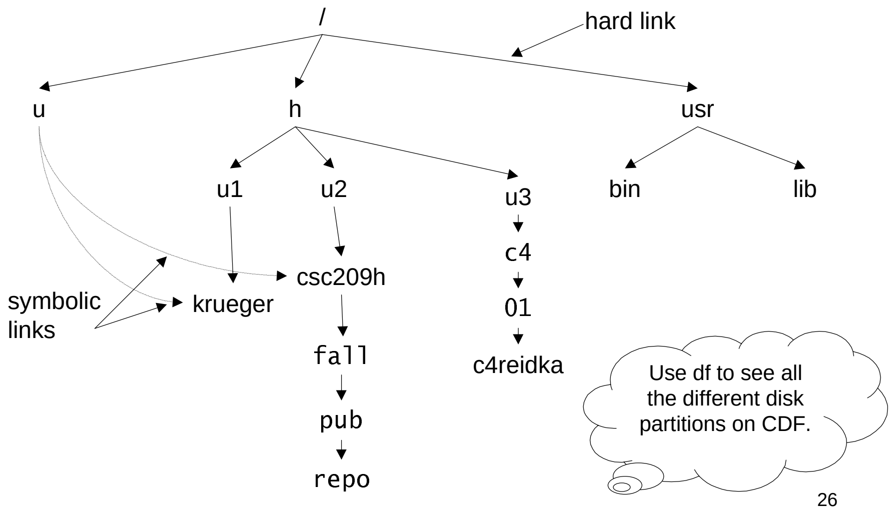
```
crw-------
brw-------
```

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. (Ch 4.3)
- 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



Use df to see all the different disk partitions on CDF.

File Systems and Links

Ch. 4.5, 4.3, 3.2

- 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.

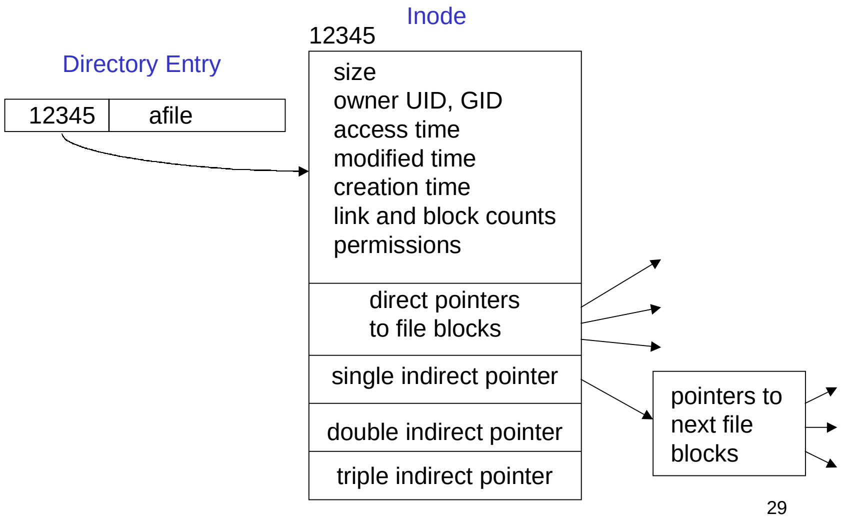
Directories and Links

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



Ch 3.3

Stat

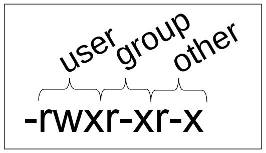
```
eddie% stat csc209h
  File: "csc209h"
  Size: 3584         Allocated Blocks: 8
  Filetype: Directory
  Mode: (0755/drwxr-xr-x)
        Uid: (    0/    root)  Gid: (  517/
  csc209h)
Device: 0/6  Inode: 1055265   Links: 143
  Device type: 0/0
Access: Sun Aug 26 15:00:58 2001
Modify: Mon Jul 23 09:26:51 2001
Change: Mon Jul 23 09:26:51 2001
```

"man 2 stat" shows the C function

Ch 3.1

Permissions

```
-rwxr-xr-x  1 reid  0 Jan 11 22:26   allexec*
-rw-r--r--  1 reid  0 Jan 11 22:23   allread
-rw-------  1 reid  0 Jan 11 22:23   ownerread
-r--r--r--  1 reid  0 Jan 11 22:23   readonly
drwxr-xr-x  1 reid  0 Jan 11 22:23   dir-read
drwx--x--x  1 reid  0 Jan 11 22:23   dir-search
```



- 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

```
-r--r--r--  1 reid  0 Jan 11 22:23   readonly
dr-xr-xr-x  1 reid  0 Jan 11 22:23   dir-read
d--x--x--x  1 reid  0 Jan 11 22:23   dir-search
---x--x--x  1 reid  0 Jan 11 22:23   dir-search/xfile
```

What is the result of the following:

```
$ ls readonly
$ readonly
$ ls dir-search
$ dir-search/xfile
$ cd dir-search
```

Use chmod to change file permssions.