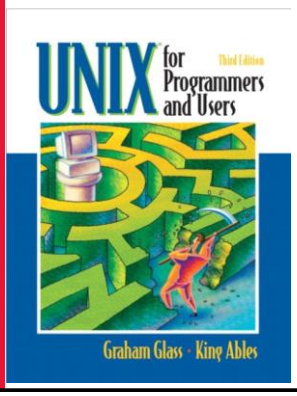
EECS2031 Software Tools

F 2021

Nov 17, 2021 Lecture 19

1

UNIX

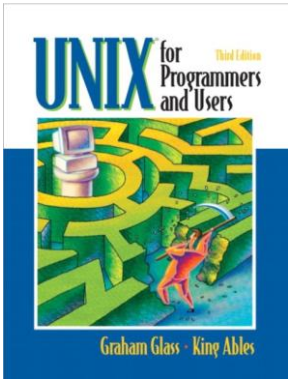


2

Contents

- Overview of UNIX
  - Structures
  - File systems
    - Pathname
    - Security
  - Process:
    - Exit code
    - Pipes

Plan for today



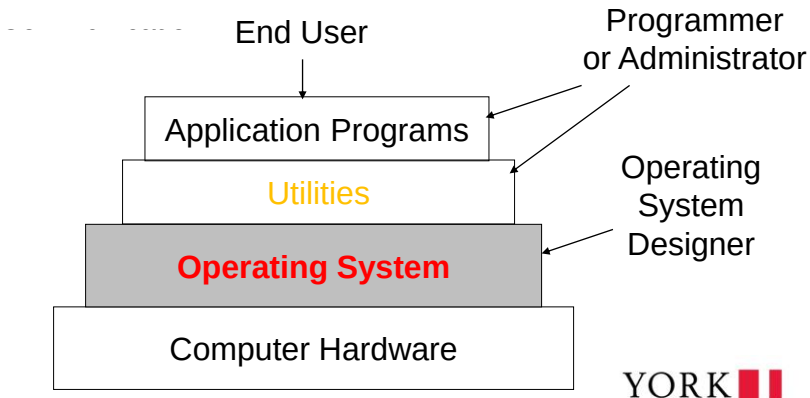
- Utilities/commands
  - Basic
  - Advanced
- Shell and shell scripting language

3

3

Layers and Views of a Computer System

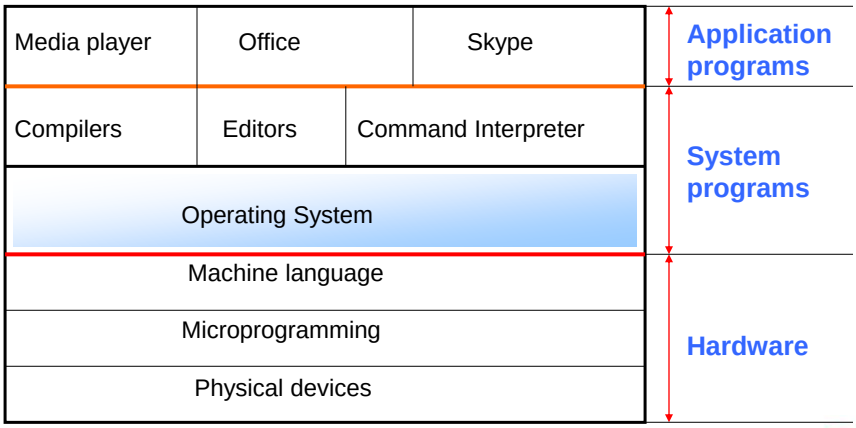
- Computer Systems



4

4

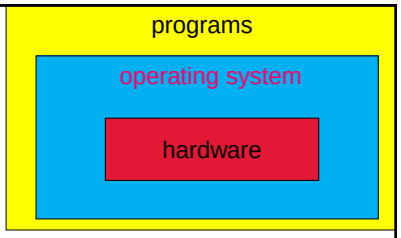
Computer Systems



5

5

Operating Systems



- An operating system
  - a program that controls the execution of application programs and acts as an interface between the user (program) of a computer and the computer hardware.
- An operating system has four major components:
  - process management,
  - memory management
  - the file system
  - input/output
- The most common OS include
  - Windows, MacOS, and UNIX (like).

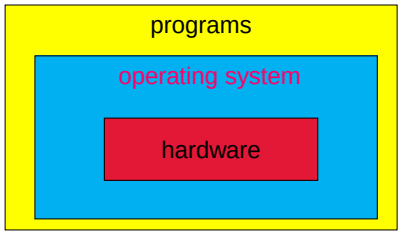


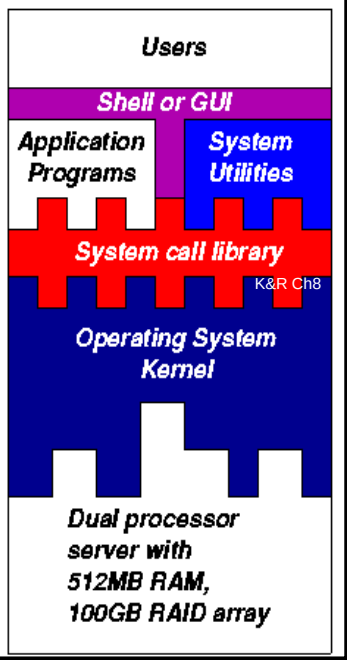
6

6

Operating systems

- General architecture
  - Kernel: deal with hardware
  - HW-independent expose to high level by system call
  - Utilities: small, come with OS
  - Shell: textual command-line interface



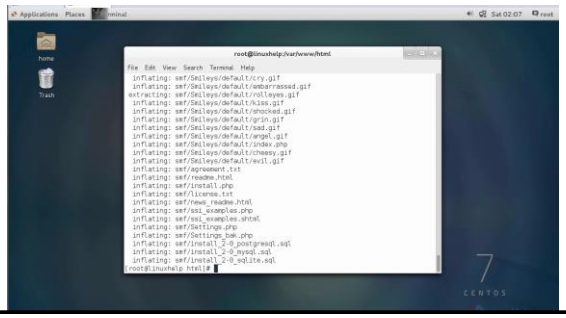


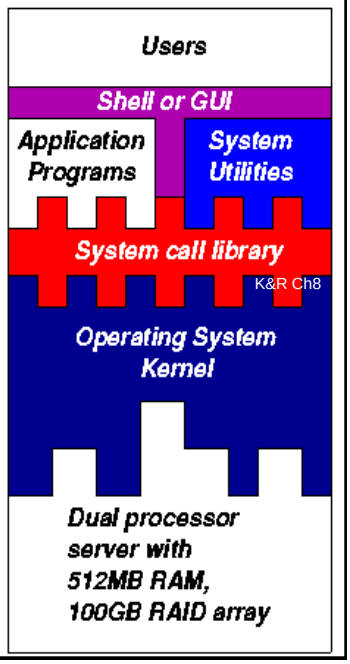
7

7

Operating systems

- General architecture
  - Kernel: deal with hardware
  - HW-independent expose to high level by system call
  - Utilities: small, come with OS
  - Shell: textual command-line interface



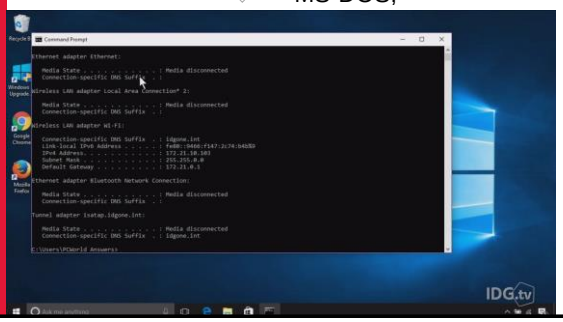


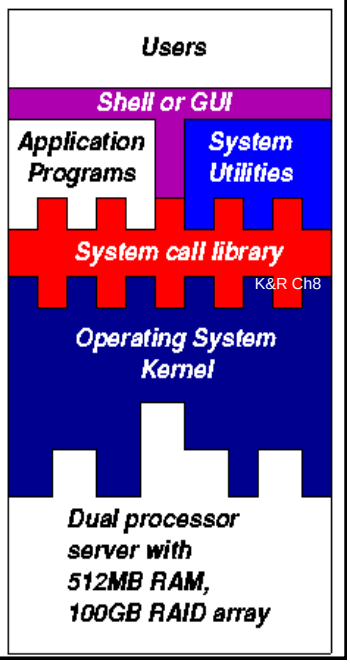
8

Operating systems

- General architecture
  - Kernel: deal with hardware
  - HW-independent expose to high level by system call
  - Utilities: small, come with OS
  - Shell: textual command-line interface

⇩ MS-DOS,



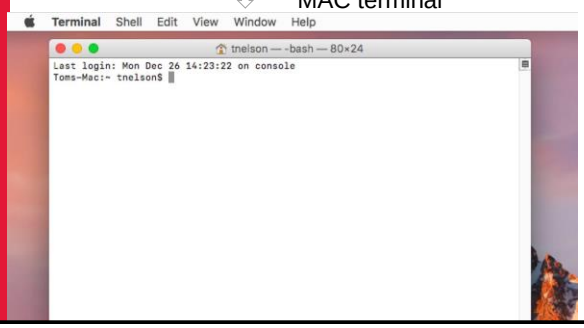


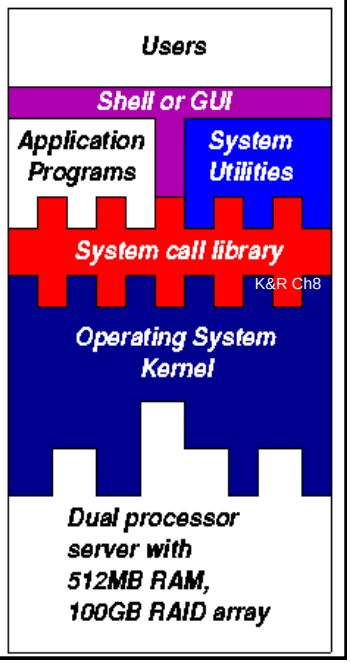
9

Operating systems

- General architecture
  - Kernel: deal with hardware
  - HW-independent expose to high level by system call
  - Utilities: small, come with OS
  - Shell: textual command-line interface

⇩ MAC terminal



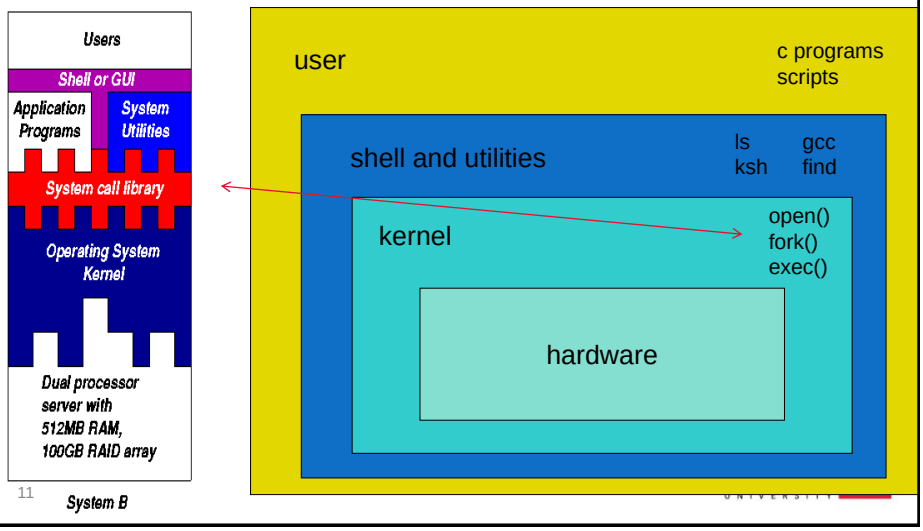


10

Unix System Structure

For your information

Unix is a popular operating system. It is most commonly used in backend applications, on servers, powerful workstations, etc. One of the most well-designed operating systems of its time.



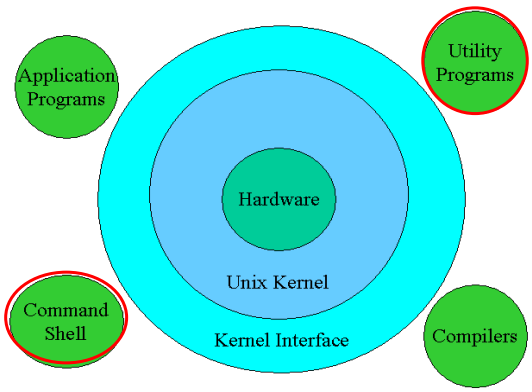
11

11

For your information

Unix System Structure

- Another picture



Unix Architecture

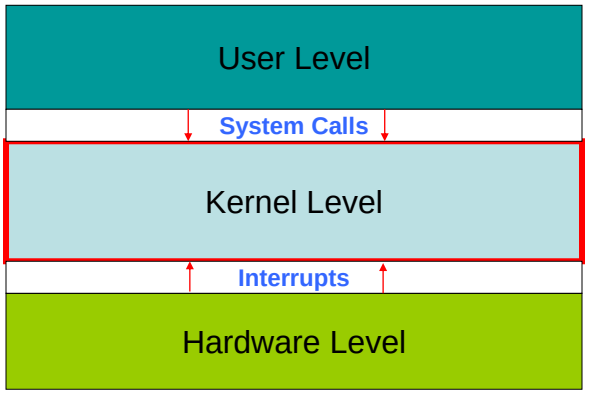
12

12

For your information

Kernel Interactions

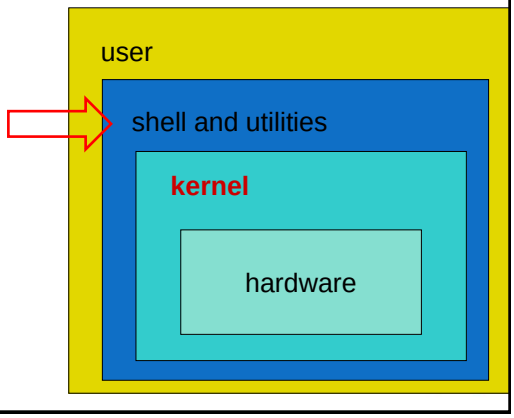
- Processes accesses kernel facilities via system calls
- Peripherals communicate with the kernel via hardware interrupts



K&R Ch 8

Shell and Utilities

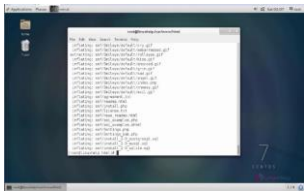
- The rest of the operating system
- Focus of the Unix lectures of this course



UNIX Shell (briefly)

- Shell is the (command line) user interface to the operating system
  - between you and the raw UNIX OS
  - when you log in, you interactively use the shell (old system)

- Functionality:
  - Execute other programs
  - Manage files, processes



- Command-line utilities\shell commands e.g., cd ls mkdir ….

- Scripting
  - A set of shell commands that constitute an executable program: a script
  - Batch file .bat in Windows

```
date
ls
…..
```

YORK UNIVERSITÉ UNIVERSITY

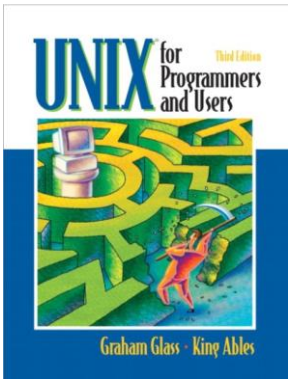
15

15

Contents

- Overview of UNIX
  - Structures
  - File systems
    - Pathname
    - Security
  - Process:
    - Exit code
    - Pipes

today



- Utilities/commands
  - Basic
  - Advanced

- Shell and shell scripting language

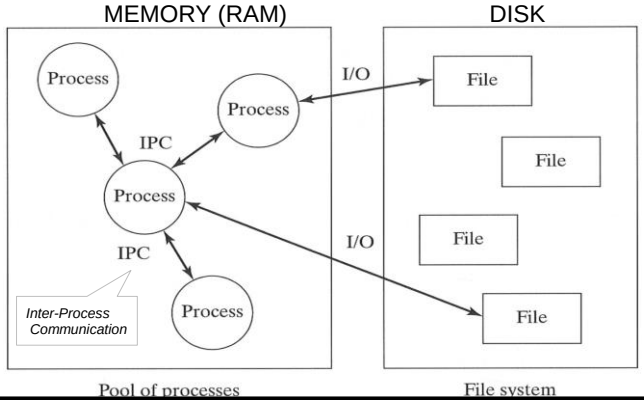
YORK UNIVERSITÉ UNIVERSITY

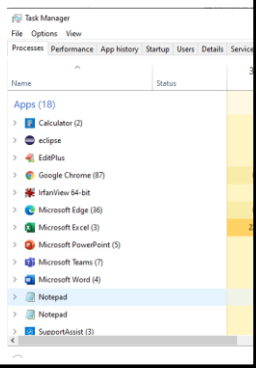
16

16

Files and Processes

- A file is a collection of data that is usually stored on DISK
- When a program is invoked, it is loaded from DISK into MEMORY. When a program is running (in MEMORY), it is called a process.
- Most processes read and write data from files.

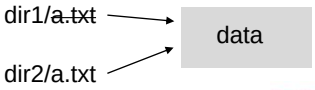




Unix File System

- Files are just sequences of bytes
- Directories are lists of files and other (sub)directories, along with their status:
  - creation date
  - permissions, etc.
- Each directory entry links (points) to a file on the disk
  - moving a file does not actually move any data around.
    - creates link in new location
    - deletes link in old location

Try to move (cut+paste) a 3G movie, see how quick it is

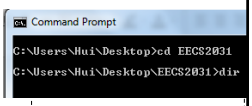


File Pathnames

- Two files in different directories can have the same name. We need pathnames to differentiate between files with the same names located in different directories.

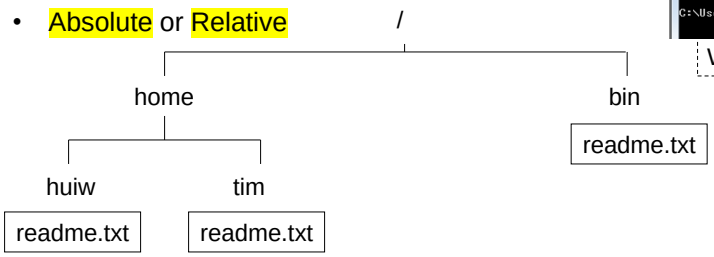
- A pathname is a sequence of directory names that leads you through the hierarchy from a starting directory to a target file.
  - Directory names separated by /

gcc /cs/dept/course/2020-21/S/2031A/submit/lab3/eecs12345/lab3A.c .
cat /home/tim/readme.txt



Windows uses \

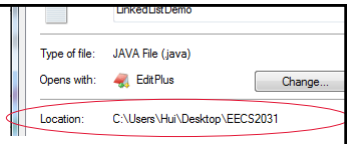
- Absolute or Relative



20

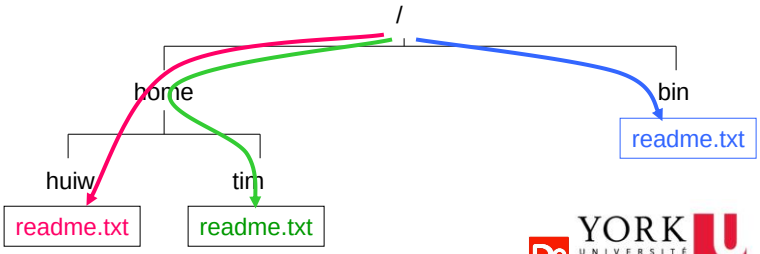
20

File: Absolute Pathnames



- A pathname starts from the root directory of file system is often termed an absolute, or full pathname. / c:\
- Valid from anywhere.

cat /home/huiw/readme.txt ~/readme.txt
cat /home/tim/readme.txt
cat /bin/readme.txt



21

From anywhere, `cat /home/tim/readme.txt`

21

File: Relative Pathnames

- A process may also unambiguously specify a file by using a pathname relative to its current working directory.
- UNIX file system supports the following special fields that may be used when supplying a relative pathname:

Field	Meaning
.	current directory
..	parent directory

Same in DOS

```
cat ./input.txt     cat input.txt
gcc  ./a1.c    or    gcc a1.c
./a.out  <  ../input.txt
cd ..
```

22

22

Relative Pathnames

- Relative Pathnames (from **/home/huiwang**)

cat readme.txt or cat ./readme.txt

cat ../tim/readme.txt ../../home/tim/readme.txt ../../tim/readme.txt

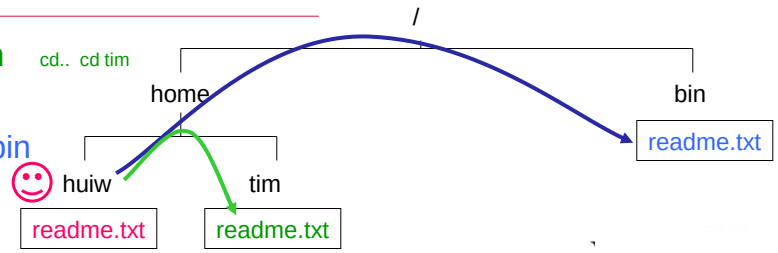
cat ../../bin/readme.txt

rm ../../bin/readme.txt

- Not changing dir
- Stay in huiw

cd ../tim cd.. cd tim

cd ../../bin



choose wisely cd ../../../../../../../../lab1 ???

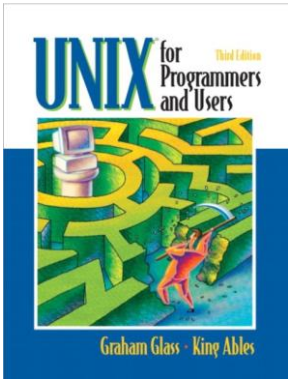
23

23

Contents

- Overview of UNIX
  - Structures
  - File systems
    - Pathname
    - Security
  - Process:
    - Exit code
    - Pipes

today



- Utilities/commands
  - Basic
  - Advanced
- Shell and shell scripting language

24

24

File: Unix Security

- Processes and files have an owner and may be protected against unauthorized access.
- A set of users can form a group. A user can be a member of multiple groups.
- Each user has a primary (default) group.
  - Your primary group is 'ugrad'
  - also 'submit' and 'labtest'

```
red 302 % groups
grad submit faculty labtest
red 303 %
```

25

25

File and Directory Permissions

- UNIX provides a way to **protect files** based on users and groups.

- Three types of permissions:
  - read (r) process may read contents of file
  - write (w) process may write contents of file
  - execute (x) process may execute file

- Three sets/clusters of permissions:
  - permissions for `owner`
  - permissions for `group`
  - permissions for `other`

`-rwxr-xr--`

owner group other

- Same types and sets of permissions as for files apply to directories:
  - read process may read the directory contents (i.e., list files)
  - write process may add/remove files in the directory
  - execute process may open files in directory or subdirectories

26

26

File Permissions (Security)

- File permissions are the basis for file security. They are given in three clusters.

$ ls -l heart.txt

- rw- r-x r-- 1 huiwang faculty 213 Jan 31 00:12 heart.txt

User (owner)	Group	Others
r w -	r - x	r - -

clusters

Each cluster of three letters has the same format:

Read permission	Write permission	Execute permission
r	w	x

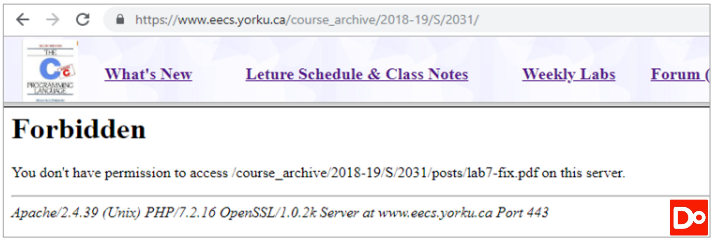
e.g., webfile: others need to have r permission
submit dir: group need to have w permission

27

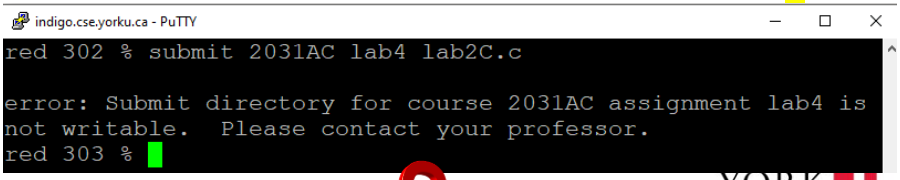
27

Permission examples

webfile: others must has r permission -rwxr-x-wx



submit directory: group must has w permission -rwxr-xr--



28 How to set/change permission?

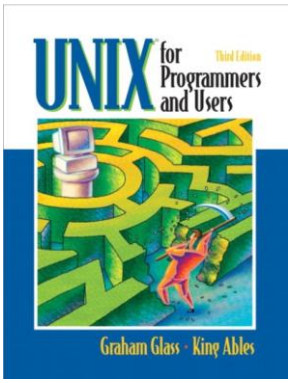
YORK UNIVERSITÉ UNIVERSITY

28

Contents

- Overview of UNIX
  - Structures
  - File systems
    - Pathname
    - Security
  - Process:
    - Exit code
    - Pipes
- Utilities/commands
  - Basic
  - Advanced
- Shell and shell scripting language

today



29

YORK UNIVERSITÉ UNIVERSITY

29

Processes

- Each command/utility involves a process
  - `ls, cd, pwd, gedit, gcc …`
  - Unix can execute many processes simultaneously.

- When a process ends, there is a return value aka exit code associated with the process outcome
  - a non-negative integer. ≥ 0

    0 means success

    anything > 0 represents various kinds of failure

    Opposite to C
  - The return value is passed to the parent process

    Stored in system variable **$?** (Usually used in shell script)

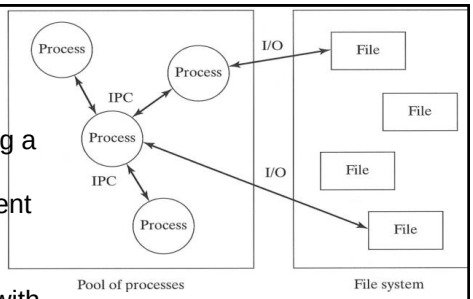
```
sh-4.2$ pwd
/cs/home/huiwang
sh-4.2$ echo $?
0
sh-4.2$ date
Sat Mar 30 09:15:52 EDT 2019
sh-4.2$ echo $?
0
sh-4.2$
```

```
sh-4.2$ cd xxx
sh: cd: xxx: No such file or directory
sh-4.2$ echo $?
1
sh-4.2$ ls xxx
ls: cannot access xxx: No such file or directory
sh-4.2$ echo $?
2
sh-4.2$
```

30

30

Process: Communication



- Processes can communicate using a number of means:
  - passing arguments, environment
  - read/write regular disk files
  - exit values $?
  - inter-process communication with shared queues, memory and semaphores
  - signals
  - pipes
  - sockets

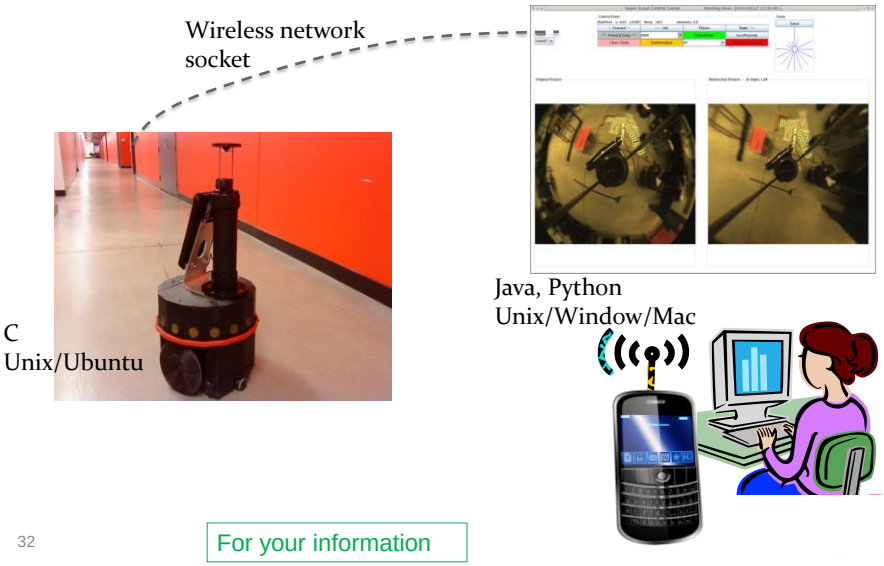
- A pipe is a one-way medium-speed data channel that allows two processes on the **same** machine to talk.

- If the processes are on different machines connected by a network, then a mechanism called a "socket" may be used instead. A socket is a two-way high-speed data channel.

31

31

Socket communication (on different machine)



For your information

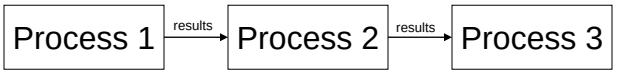
Process communication: Unix Pipes

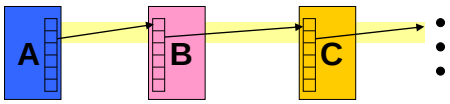
- A special mechanism called a "pipe" built into the heart of UNIX to support cascading utilities.
- A pipe allows a user to specify that the output of one process is to be used as the input to another process.
- Two or more processes may be connected in this fashion, resulting in a "pipeline" of data flowing from the first process through to the last.

Process 1 → results → Process 2 → results → Process 3

Process communication: Unix Pipes

- The nice thing about pipelines is that many problems can be solved by such an arrangement of processes.
- Each process in the pipeline performs a set of operations upon the data and then passes the results on to the next process for further processing.





34

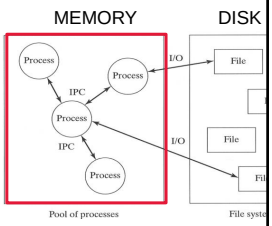
34

Pipeline Example

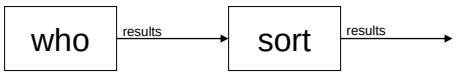
- A utility called **who** outputs an unsorted list of current users. Another utility called **sort** outputs a sorted version of its input.

  $ who

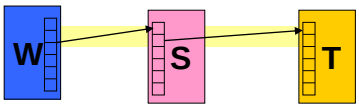
  $ sort input.txt or sort < input.txt



- These two utilities may be connected together with a "pipe" so that the output from **who** passes directly into **sort**, resulting in a sorted list of users. | does this job.



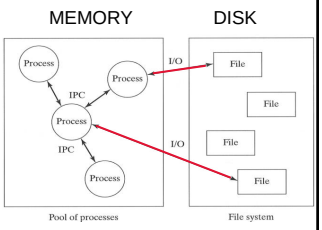
$ who | sort



35

35

Pipe-Equivalent Communication Using a File

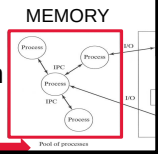


- Could we solve without using a pipe? YES.
- run who, save output into a disk file
- run sort, using this disk file as input

$ who > tmp.txt
$ sort < tmp.txt

- Disadvantages:
  - Unnecessary use of the disk
    - Slower
    - Can take up a lot of space
  - Makes no use of multi-tasking

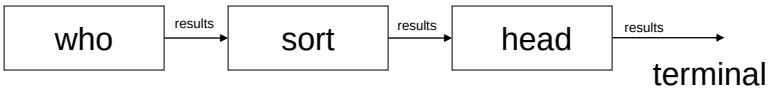
- Pipe is very similar, but does not involve the external device (disk)
  - all mechanisms stay in the realm of the operating system



36 $ who | sort $ who | wc -l $ who | sort -k3
$ who | sort | head -5

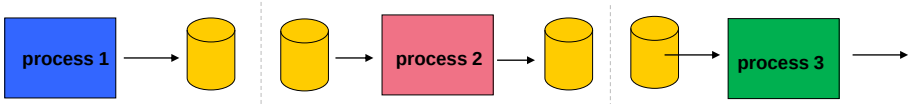
36

More examples



- who | sort | head -5 # list fist 5 in the sorted list

Without pipe



- who > tmp; sort < tmp > tmp2; head -5 < tmp2;

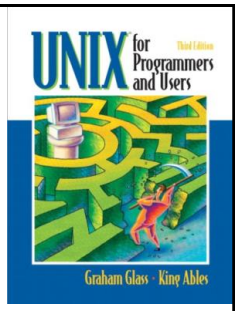
37 ls | more

dir /p or ***dir | more*** in DOS

YORK UNIVERSITÉ UNIVERSITY

37

Contents



- Overview of UNIX
  - Structures
  - File systems
    - absolute vs relative pathname ../../input.txt
    - security -rwx--x--x
  - Process:
    - has return value 0 (success) or > 0 (something wrong)
    - communication: pipes who | sort ls | more
  - Extra readings today
- Utilities/commands
  - Basic
  - advanced
- Shell and shell scripting language

38

38

Unix Versions (extra reading)

For your information

- UNIX is a fairly standard operating system, with two main versions that are slowly merging into one.
- UNIX was created in Bell Laboratories and evolved from that into what is currently known as "System V" UNIX.
- The university of California at Berkeley obtained a copy of UNIX early on in its development and spawned another major version, known as BSD (Berkeley Standard Distribution) UNIX.
- **UNIX international**
  - AT&T, Sun Microsystems, --> System V Release 4.
- **Open Software Foundation**
  - IBM, Digital Equipment Corporation, Hewlett-Packard --> BSD UNIX, called OSF/1.

39

39

(extra reading)

For your information

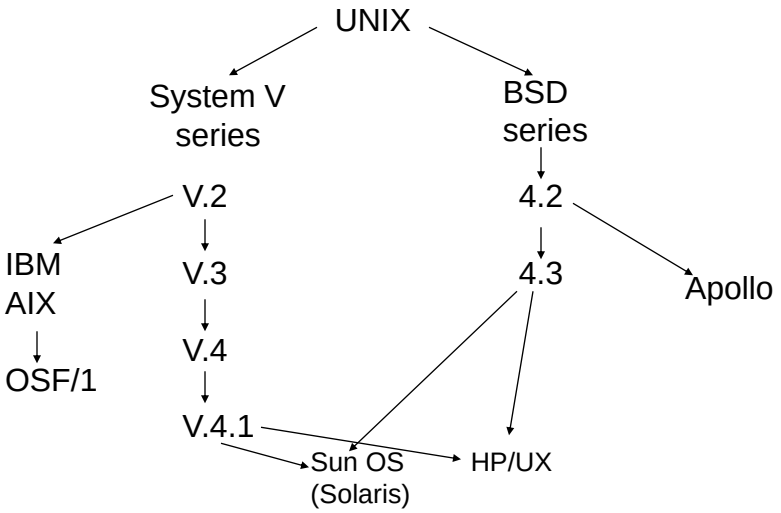


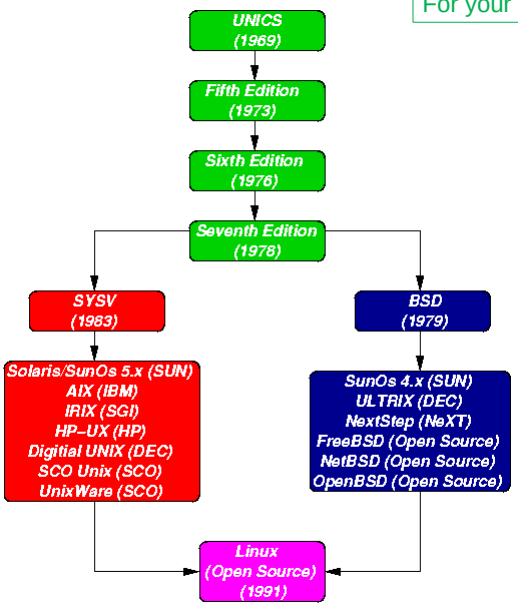
FIGURE 1.6 An abbreviated genealogy of UNIX

40

40

(extra reading)

For your information



Come to my office for a very detailed poster

41

41

(extra reading)

For your information

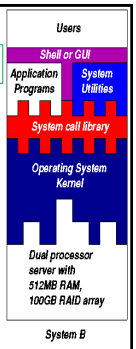
Unix Standards

- Both groups tried to comply with a set of standards set by the **POSIX** (Portable Operating System Interface) committee
- Most of the best features of BSD UNIX have been rolled into most System V-based versions of UNIX.
- UNIX is mostly written in the C language, which makes it relatively easy to port to different platforms.
- This feature is an important benefit and has contributed a great deal to the proliferation and success of UNIX.

(extra reading)

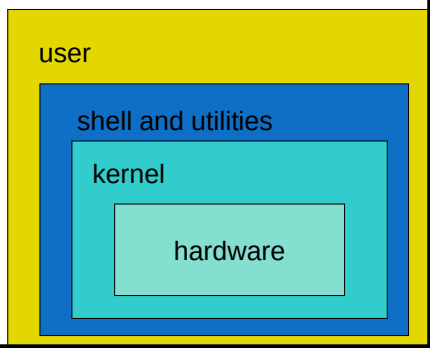
For your information

Linux as an Unix-like OS



- Reimplementation. Adheres to the same POSIX standard as UNIX
  - Looks and acts alike
  - Bebian, Fedora, Red Hat, Ubuntu, SuSE, etc.

- Kernel
- Shells and GUI
  - sh, bash, csh …
  - KDE GNOME
- System utilities
  - ls,cp, wc,more,grep,awk,sed…
- Application programs
  - Emacs, gcc, xfig,latex, soffice



(extra reading)

For your information

What is the difference between Linux and Unix?

- Ans 1: Linux is a unix-like kernel. It is open source.

- Ans2: To put it very generically, Linux is an operating system kernel, and UNIX is a certification for operating systems. The UNIX standard evolved from the original Unix system developed at Bell Labs. After Unix System V, it ceased to be developed as a single operating system, and was instead developed by various competing companies, such as Solaris (from Sun Microsystems), AIX (from IBM), HP-UX (from Hewlett-Packard), and IRIX (from Silicon Graphics). UNIX is a specification for baseline interoperability between these systems, even though there are many major architectural differences between them. Linux has never been certified as being a version of UNIX, so it is described as being "Unix-like." A comprehensive list of differences between Linux and "UNIX" isn't possible, because there are several completely different "UNIX" systems.

44

44

(extra reading)

For your information

Unix file systems

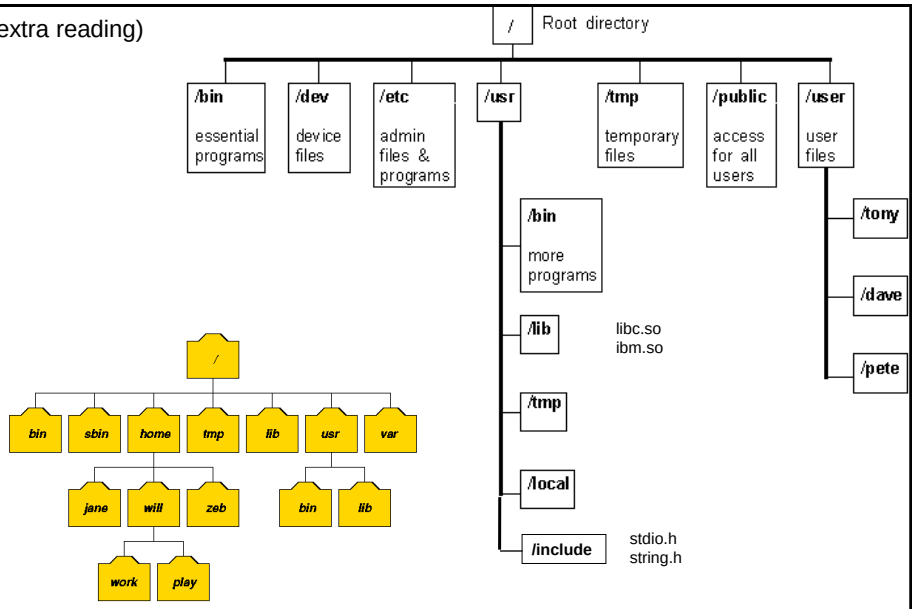
- Unix expands the usual definition of file to include anything from which data can be taken, sent., including io devices (kb, printer,)
  - In Unix, "Everything is a file"

- Four kinds of files
  - Orindary files (regular files) hold info text/ binary files
  - Special files – (device files). Representing physical devices, keyborards, terminals, printers and other peripherals.
  - Directory files (directories). Hold other files and directories
  - Link (pointer) to files hardlink, softlink

45

45

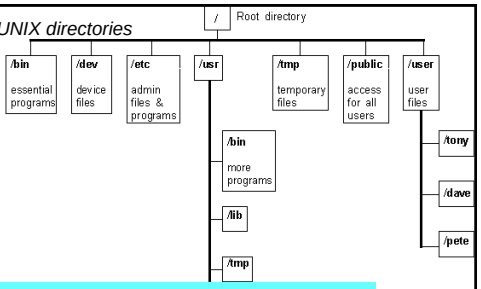
(extra reading)



For your information

(extra reading)

Fig. 2.2: Typical UNIX directories



Directory	Typical Contents
/	The "root" directory
/bin	Essential low-level system utilities
/usr/bin	Higher-level system utilities and application programs
/lib	Program libraries (collections of system calls that can be included in programs by a compiler) for low-level system utilities
/usr/lib	Program libraries for higher-level user programs
/tmp	Temporary file storage space (can be used by any user)
/home or /homes	User home directories containing personal file space for each user.
/etc	UNIX system configuration and information files
/dev	Hardware devices

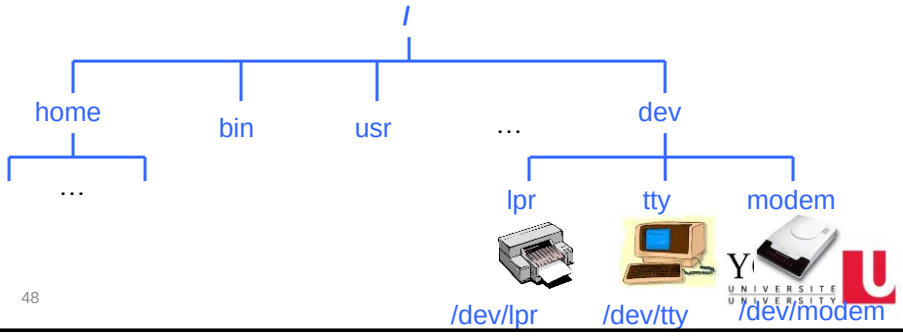
For your information

(extra reading)

For your information

Devices as Special Files

- Besides files, input and output can go from/to various hardware devices.
- Unix Philosophy: Treat these devices as special files!
- Terminals, printers, and other devices are accessible in the same way as disk-based files.



48

48

(extra reading)

For your information

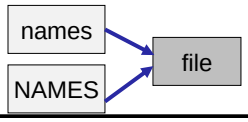
Links

- A *link* is a pointer to a file.
- In fact, in UNIX *all* filenames (directory entry) are just links to a file. Most files only have one link.

```
-rw-r--r--   1 jbond    cs          154 Feb  4 15:00 letter3
-rw-r--r--   1 jbond    cs           64 Feb  4 15:00 names
drwxr-xr-x   1 jbond    cs          512 Feb  4 15:00 secret/
```

- Additional links to a file allow the file to be shared.
- The ln command creates new links.

```
$ ln names NAMES
$ ls -l
total 8
-rw-r--r--   2 jbond    cs           64 Feb  6 18:36 NAMES
-rw-r--r--   1 jbond    cs          154 Feb  4 15:00 letter3
-rw-r--r--   2 jbond    cs           64 Feb  4 15:00 names
drwxr-xr-x   1 jbond    cs          512 Feb  4 15:00 secret/
```



Hard link

49

49

Unix Utilities

- Standard UNIX comes complete with at least 200 small utility programs, usually including:
  - shells,
  - editors,
  - a C compiler,
  - matching with regular expressions,
  - searching utilities,
  - sorting utilities,
  - software development tools,
  - text-processing tools, etc.

Heads-up of utilities

indigo.cse.yorku.ca - PuTTY

```
red 310 % date
Thu Nov 12 14:15:29 EST 2020
red 311 % cal 2020
                                   2020

       January               February                March
Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa
          1  2  3  4                      1    1  2  3  4  5  6  7
 5  6  7  8  9 10 11    2  3  4  5  6  7  8    8  9 10 11 12 13 14
12 13 14 15 16 17 18    9 10 11 12 13 14 15   15 16 17 18 19 20 21
19 20 21 22 23 24 25   16 17 18 19 20 21 22   22 23 24 25 26 27 28
26 27 28 29 30 31      23 24 25 26 27 28 29   29 30 31

        April                  May                   June
Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa
          1  2  3  4                   1  2       1  2  3  4  5  6
 5  6  7  8  9 10 11    3  4  5  6  7  8  9    7  8  9 10 11 12 13
12 13 14 15 16 17 18   10 11 12 13 14 15 16   14 15 16 17 18 19 20
19 20 21 22 23 24 25   17 18 19 20 21 22 23   21 22 23 24 25 26 27
26 27 28 29 30         24 25 26 27 28 29 30   28 29 30
                       31
        July                  August                September
Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa
          1  2  3  4                      1          1  2  3  4  5
 5  6  7  8  9 10 11    2  3  4  5  6  7  8    6  7  8  9 10 11 12
12 13 14 15 16 17 18    9 10 11 12 13 14 15   13 14 15 16 17 18 19
19 20 21 22 23 24 25   16 17 18 19 20 21 22   20 21 22 23 24 25 26
26 27 28 29 30 31      23 24 25 26 27 28 29   27 28 29 30
                       30 31
       October               November              December
Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa   Su Mo Tu We Th Fr Sa
             1  2  3    1  2  3  4  5  6  7          1  2  3  4  5
 4  5  6  7  8  9 10    8  9 10 11 12 13 14    6  7  8  9 10 11 12
11 12 13 14 15 16 17   15 16 17 18 19 20 21   13 14 15 16 17 18 19
18 19 20 21 22 23 24   22 23 24 25 26 27 28   20 21 22 23 24 25 26
25 26 27 28 29 30 31   29 30                   27 28 29 30 31
```

indigo.cse.yorku.ca - PuTTY

```
red 316 % cal 2 2020
    February 2020
Su Mo Tu We Th Fr Sa
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29

red 317 % cal 6 1997
      June 1997
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30

red 318 %
```

Basic utilities/commands

Introduces the following command-line utilities, listed in alphabetical order:

cancel	head	mv
cat	less	newgrp
chgrp	lp	page
chmod	lpr	passwd
chown	lprm	pwd
clear	lpq	rm
cp	lpstat	rmdir
date	ls	stty
file	mail	tail
groups	man	tset
	mkdir	wc
	more	

55

55

Basic utilities/commands

Introduces the following utilities, listed in groups:

General	Directory	File	File print
man	pwd	cat	lp
clear	ls	more less	lpr
echo	cd	head tail	lprm
date	mkdir	cp	lpq
	rmdir	mv	lpstat
		rm	
		wc	
		file	
		chmod	
		chgrp	
		newgrp	
		chown	

56

56

Running a utility/command

- To run a utility, simply enter its name at the prompt and press the Enter key.

  - Up/down arrow for history
  - Tab key for auto complete

$ date # run the utility.
Sun Jul 14 20:10:42 EDT 2019
$ _

clear

- clears your screen

cls in DOS

echo

- print statement in UNIX/LINUX

- echo $?
- echo hello

```
sh-4.2$ echo hello
hello
sh-4.2$
```

Same in DOS

```
C:\Users\huiwang\Desktop>echo hello
hello

C:\Users\huiwang\Desktop>echo we are good
we are good

C:\Users\huiwang\Desktop>_
```

man: online help

- All UNIX systems have a utility called man (short for manual page) that puts this information at your fingertips.

- The manual pages are on-line copies of the original UNIX documentation, which is usually divided into eight sections. They contain information about utilities, system calls, file formats, and shells.

man [section] word
man -k keyword

- The first usage of man displays the manual entry associated with word. If no section number is specified, the first entry that it finds is displayed.

- The second usage of man displays a list of all the manual entries that contain keyword.

59

59

Organization of the manual pages

- The typical division of topics in manual pages (sections) is as follows:

1. **Commands and Application Programs.**
2. System Calls
3. **C Library Functions**
4. Special Files
5. File Formats
6. Games
7. Miscellaneous
8. System Administration Utilities

% `man man`

```
MANUAL SECTIONS
       The standard sections of the manual include:

       1      User Commands

       2      System Calls

       3      C Library Functions

       4      Devices and Special Files

       5      File Formats and Conventions

       6      Games et. Al.

       7      Miscellanea

       8      System Administration tools and Deamons
```

% `man 1 date`

% `man 3 strlen`

60

60