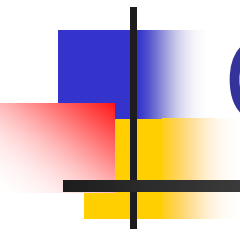
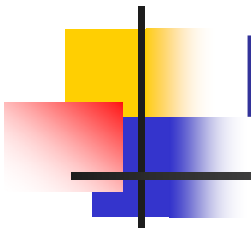


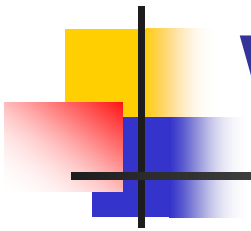
Chapter 3: File I/O

CMPS 105: Systems Programming
Prof. Scott Brandt
T Th 2-3:45
Soc Sci 2, Rm. 167



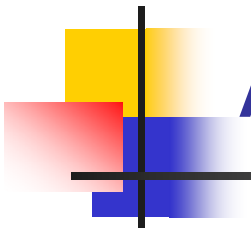
First

- Questions?
- Programming Assignment 1
- Programming Assignment 2
- Class in general?



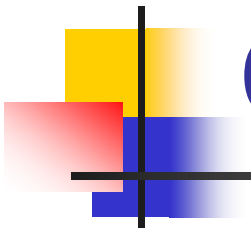
What is a file?

- Data storage
- Byte stream
- Named
- Non-volatile
- Shared
- Protected
- ...



Accessing Files

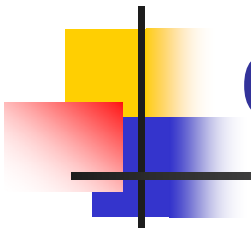
- System manages disk
- Abstraction and manages sharing
- Protection
  - Can only access via system calls
  - Permissions
- Abstraction
  - creat(), open(), close(), read(), write(), stat()



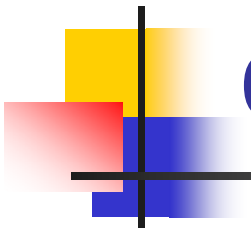
Creat and open

- Creat creates a new file
- Open opens or creates a file
- Both return a file descriptor
  - Index into kernel table
  - Entry contains relevant info
- May fail if
  - No file
  - No permission
  - ...

open()

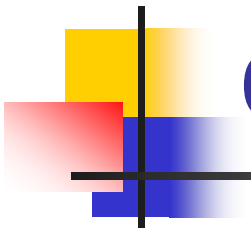


- Sys/types.h, sys/stat.h, fcntl.h
- int open (const char *pathname, int oflag, ..., /* mode_t mode */);
- Pathname may be absolute or relative
- Oflag = O_RDONLY, O_WRONLY, O_RDRW, O_APPEND, O_CREAT, O_EXCL, O_TRUNC, O_NOCTTY, O_NONBLOCK, O_SYNC
- Returns error status



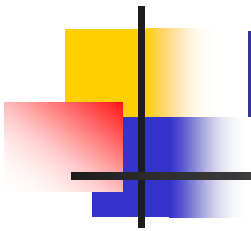
creat()

- Sys/types.h, sys/stat.h, fcntl.h
- int creat(const char *pathname, mode_t mode);
- Pathname
- Mode specifies permissions (section 4.5)
- Returns error status



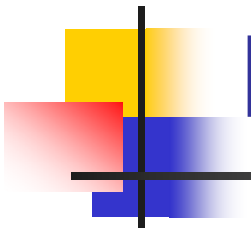
close(int fildes)

- Unistd.h
- Int close(int filedes);
- Closes an open file
- Takes a file descriptor as a parameter
- All open files are closed automatically when a process terminates
- Returns error status



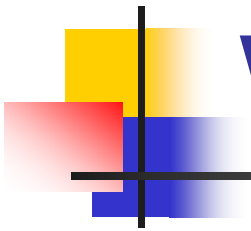
lseek()

- Sys/types.h, unistd.h
- off_t lseek(int fildes, off_t offset, int whence)
- Whence: SEEK_SET, SEEK_CUR, SEEK_END
- Returns current position



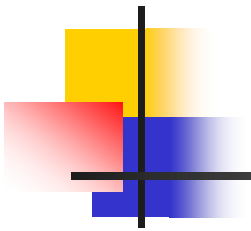
Read()

- Unistd.h
- Ssize_t read(int filedes, void *buff, size_t n_bytes);
- Returns number of bytes read or 0 (end of file) or -1 (error)



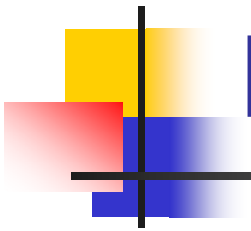
Write()

- Unistd.h
- Ssize_t write(int filedes, const void *buff, size_t nbytes)
- Returns number of bytes written or error (file size or disk full)



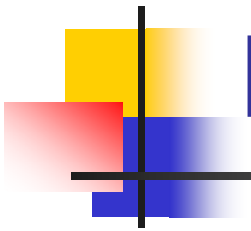
I/O Efficiency (section 3.9)

- Different buffer sizes
- Bigger buffer is better (page 57)
  - Discuss why?
- Discuss other efficiency concerns
  - What is the problem with big buffers
  - Sparse files, small files, big files
  - What other issues exist?



File Sharing

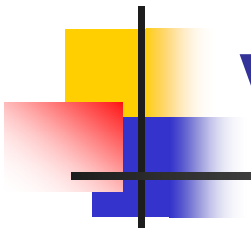
- Process table
  - Table of open file descriptors
    - File descriptor flags
    - A pointer to the file table entry
- File table of all open files
  - The file status flags (read, write, append, sync, nonblocking, ...);
  - The current file offset
  - A pointer to the v-node table entry



File Sharing (cont.)

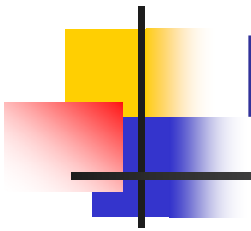
- V-node structure
  - Type of file
  - Pointers to functions that operate on it
  - i-node info
  - File size

Discuss what happens when various operations occur

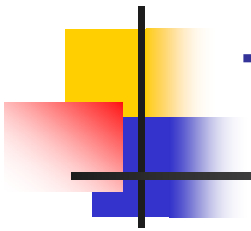


- Open
- Close
- Read
- Write
- Lseek
- Issues: position, size, modification time, creation time, contents, ...

Dup and dup2

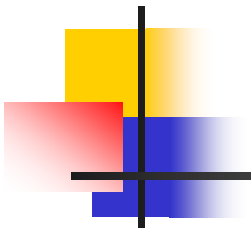


- Unistd.h
- Copy a file descriptor
- Int dup(int fildes);
  - Uses lowest numbered available file descriptor
- Int dup2(int fildes, int fildes2);
  - If fildes2 is open, it is closed first
- Can be used to reassign stdin and stdout
- Atomic



fcntl

- Sys/types.h, unistd.h, fcntl.h
- Change the properties of an open file
- Int fcntl(int fildes, int cmd, ... /*int arg*/);
- Cmds: F_DUPFD, F_GETFD, F_SETFD, F_GETFL, F_SETFL, F_GETOWN, F_SETOWN



Ioctl()

- Everything else
- Int ioctl