CS241 #23 Files Pipes Seeks II

**1. How does the C library wrap a filedescriptor?**

1. // How the C library wraps a filedescriptor
2. typedef struct \_FILE { /\* Simplified!\*/
3. int fd;
4. void\* buffer; // reduce # of write() calls
5. size\_t capacity;
6. size\_t size;
7. int buffering; // \_IONBF / \_IOLBF / \_IOFBF
8. } FILE;
10. FILE\* fdopen(int fd, char\* rwmode) {
11. ?



16. }
17. void fputs(char\*str, FILE\* file) {
18. ?

**fprintf(FILE\*, format,...)**

uses write (buffering reduces number of writes => performance)

**fseek(FILE\*, offset, whence)** SEEK\_CUR | SEEK\_SET|SEEK\_END

uses lseek(int fd, off\_t offset, int whence);

**long pos=ftell(FILE\*)** uses return lseek(fd,0,SEEK\_CUR)

**2. Challenge: Implement C library function rewind(FILE\*)**

Hint: You will need fflush and lseek or fseek (that will flush for you)

1. // Use the struct above to extract the filedes
2. void rewind(FILE\* f) {

5. }

**3. Reading & Writing binary data**

**fread** ( void \* ptr, size\_t size, size\_t nitems, FILE \* stream);

**fwrite** ( void \* ptr, size\_t size, size\_t nitems, FILE \* stream);

1. int num\_pts;
2. typedef struct { float x,y,z } p\_t;
3. p\_t\* points;
4. void load\_point\_cloud() {
5. FILE\* f = fopen("points.dat","r");
6. fread( &num\_pts , \_\_\_\_\_\_\_\_\_\_\_\_\_ , 1, f);
7. points = calloc( sizeof p\_t, num\_pts);
8. ?

Error handling/What could go wrong? Why #include<stdint.h> and using uint32\_t be better?

**4. Challenge: Read in the first half of a file as C string**

Hints: fopen, fseek, ftell, fread, malloc, fclose may be useful

1. char\* half(char\*filename) {

**5. Implement fflush**

Hint: Use & reset the FILE's output buffer, write will be useful

1. void fflush(FILE\*f) {

**6.** Amdahl's law.

My problem takes 100 seconds to calculate an answer? 25 seconds of that is non-||izable.

How can I make it run twice as fast ?

**7.** Pipe Exam Practice. Putting it all together

Write a complete program to perform the following. The parent process will copy the contents (4KB at a time) of a file 'input.txt' into stdin of the child process which exec's a bash shell

Assume *read* and *write* always complete. dup2 may be useful.