



CMSC 106

Lecture Set #13 –

File Input/Output

Set Started:

Friday, December 3, 2010



Terminating execution

```
void exit(int status);
```

- prototype in `stdlib.h`
- "immediately" ends execution when called
- status is viewable by the shell
 - `exit(0)`; generally means OK
 - `exit(1)`; or any nonzero generally means error encountered
 - can use "`echo $?`" in tcsh to see the exit status of the last program executed
- can use constants `EXIT_SUCCESS` and `EXIT_FAILURE` instead of 0/1



Types of streams

- Text streams are composed of lines of text, each terminated by a newline
 - there are library functions to deal with text streams in three ways:
 - character-oriented I/O
 - line-oriented I/O
 - formatted I/O
- Binary streams are composed of just plain data

3



The type **FILE** *

- Variables of type **FILE** * are used to represent open streams
- Three predefined streams for every program:
 - **stdin**: standard input (redirect with <)
 - **stdout**: standard output (redirect with >)
 - **stderr**: standard error (redirect both stdout and stderr with >& -- in csh or tcsh only!)

4



Stream I/O overview

1. Declare a **FILE *** variable for the file
2. Use **fopen()** to open the file
3. Read or write (or both!)
4. Use **fclose()** when done

5



Opening files

```
FILE *fopen(char *filename, char *mode);
```

- arguments are strings
- **mode** specifies access mode:
 - "r" - read; file must already exist
 - "w" - write; if file exists, it is overwritten
 - "a" - append; if file does not exist, it's created
 - there are other modes ...
- returns **NULL** on failure

6



Opening files, example

```
#include <stdio.h>

int main() {
    FILE *fp = fopen("infile.txt", "r");
    if (fp == NULL) {
        perror("can't open infile.txt");
        exit(EXIT_FAILURE);
    }
    /* read the file and close when done */
    return EXIT_SUCCESS;
}
```

7



Closing files

```
int fclose(FILE *fp);
```

- closes a file, flushing output if necessary
- returns 0 on success
- Failure does occur - the closing operation can be interrupted by the OS, or other, worse things

8



Reading and Writing to Text Files

```
int fprintf(FILE *fp,  
            formatString, values);  
int fscanf(FILE *fp,  
            formatString, addresses);
```