Week 6
More on Options

13 February 2019

CS 35L Lab 4

Jeremy Rotman

Announcements

- Assignment #5 is due Saturday by 11:55pm
- For Assignment #10
  - **Email me to tell me what story you are choosing**
  - Here is the link to see what stories people have signed up for already
    - Choose a story at least one week before you present

Outline

- ➔ Parsing Options
- ➔ fstat
- ➔ Homework 5

Questions?

C Command Line Arguments

- Main function is given two arguments
  - argc
    - The number of arguments passed to your program
  - argv
    - The array of arguments passed to your program
    - argv[0] is the name of the program

C Option Parsing

- int getopt(int argc, char* const* argv, const char* options)
    - Asks for number of arguments, the arguments, and the options
    - Options holds the flags you might expect to receive
        - It is a string of character flags, e.g. “abc” means options a, b, and c
        - By adding a ‘:’ after a character it implies that character should have an argument after it, e.g. “a:bc” means option a has an argument
- int optopt
    - If getopt receives an unknown option, it sets this to be that option
- char* optarg
    - Variable set by getopt to point to the argument of the option

Example

```
int fflag = 0;
int nflag = 0;
while ((c = getopt(argc, argv, “fn”)) != -1)
    switch(c) {
        case ‘f’:
            fflag = 1;
        case ‘n’:
            nflag = 1;
        case ‘?’:
            fprintf (stderr, "Unknown option `-%c'.\n",
optopt);
    }
```

Bash Options

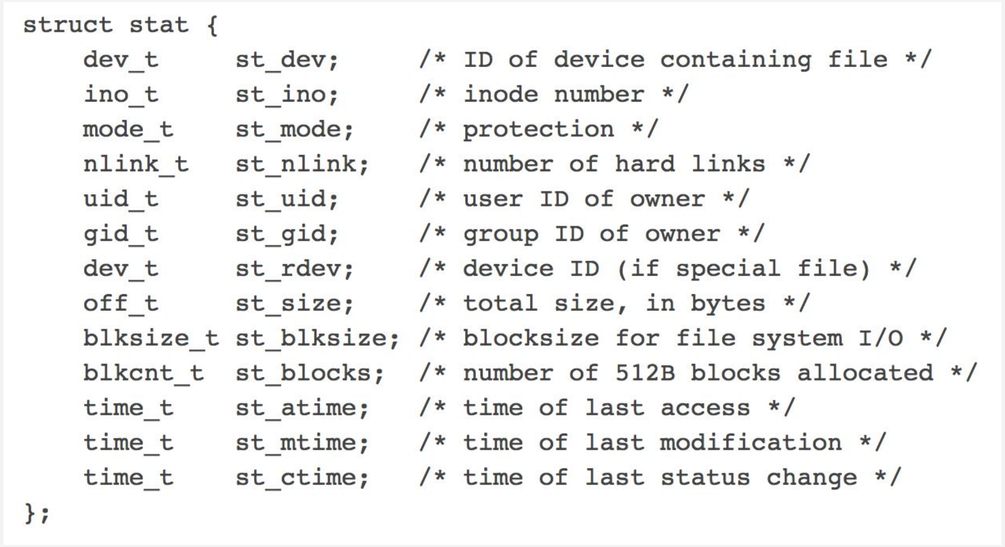
- ➔ Relatively similar to C getopt
- ➔ getopts OPTSTRING VARNAME [ARGS…]
    - VARNAME is the variable getopts gets stored in
    - OPTSTRING is essentially the same as the option string in C
    - One major difference
        - A colon at the beginning of the string changes the error reporting mode
        - Default is verbose, a colon at the beginning switches it to silent
    - Verbose vs. Silent
        - Verbose
            - Invalid option: VARNAME is set to ?, OPTARG is unset
        - Silent
            - Invalid option: VARNAME is set to ?, OPTARG is set to the invalid option

Example

```
while getopts “:f” opt; do
    case ${opt} in
    f)
        FFLAG=true
        ;;
    \?)
        echo “Invalid Option ${OPTARG}” >&2
        ;;
    esac
done
```

fstat

- ➔ Returns information about a file
    - ◆ Returns a stat struct



```
struct stat {
    dev_t     st_dev;     /* ID of device containing file */
    ino_t     st_ino;     /* inode number */
    mode_t    st_mode;    /* protection */
    nlink_t   st_nlink;   /* number of hard links */
    uid_t     st_uid;     /* user ID of owner */
    gid_t     st_gid;     /* group ID of owner */
    dev_t     st_rdev;    /* device ID (if special file) */
    off_t     st_size;    /* total size, in bytes */
    blksize_t st_blksize; /* blocksize for file system I/O */
    blkcnt_t  st_blocks;  /* number of 512B blocks allocated */
    time_t    st_atime;   /* time of last access */
    time_t    st_mtime;   /* time of last modification */
    time_t    st_ctime;   /* time of last status change */
};
```

Homework 5

- sfrobu
  - Rewrite sfrob using system calls
  - Should behave like sfrob but
    - If stdin is a regular file it should initially allocate enough memory to hold all data in file at once
      - Should still work with growing file
    - Implement an option, -f
      - Makes it ignore case
  - Relevant system calls
    - read, write, and fstat
  - Compare sfrob and sfrobu performance with the time command

Homework 5

- Additionally, write sfrobs
  - Shellscript that uses tr and sort to sort encrypted files
    - Use a pipeline
    - Should **not** make temporary files
  - Also allows a -f option to ignore case

Questions?