Theo Song
Prof. Hakner
ECE 357
10/23/2019

Problem 1 - Shell Script Invocation

1) /bin/sh

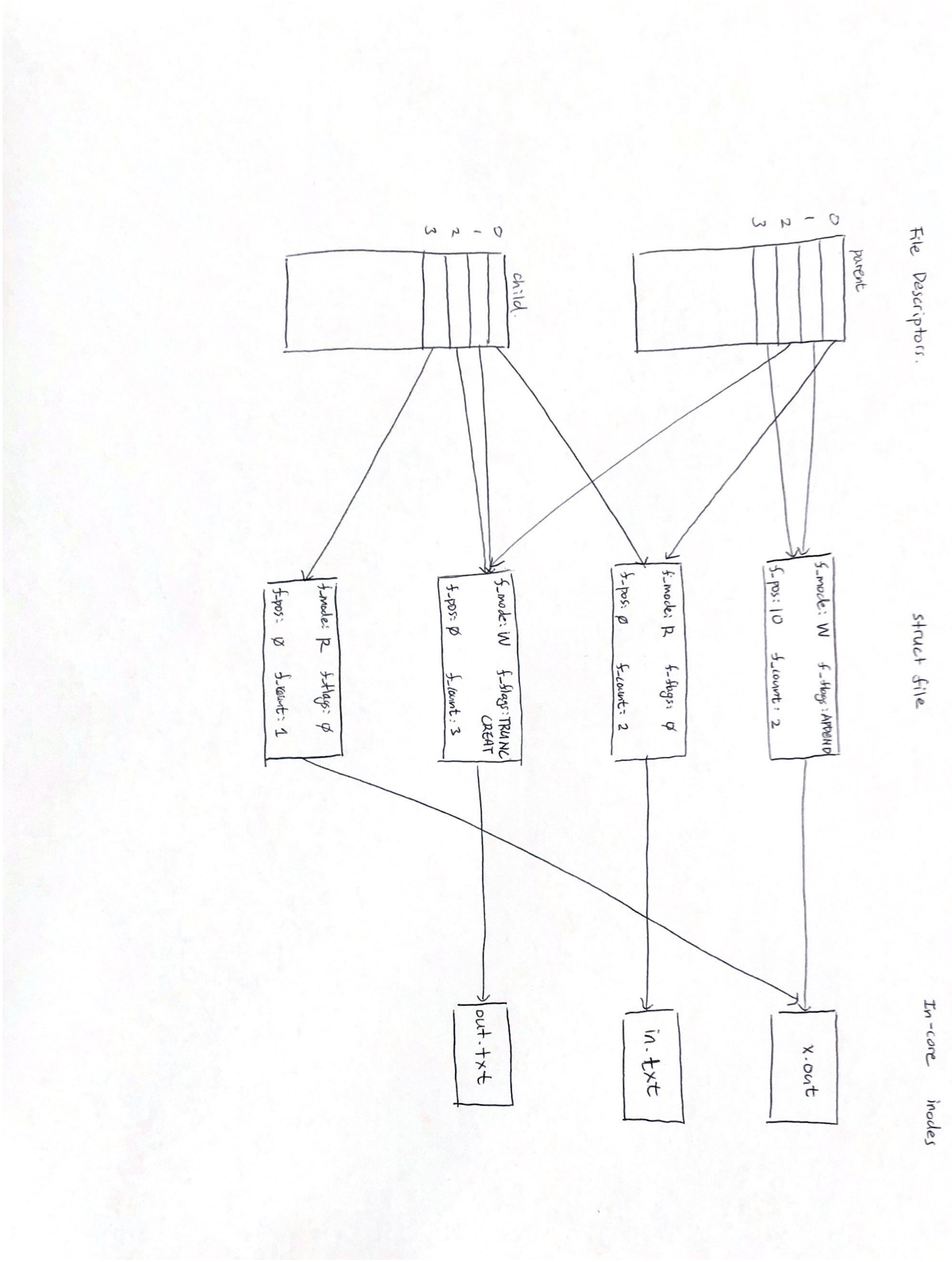
2) argc: 5

   argv: {sh, ./script.sh, f2.c, f3.c, f4.c, NULL}

3) The pid 123 calls a wait() system call and waits until the child process has executed and exits, returning the exitstatus to the parent process.

4) As the shell script is executed, the first line, which is “ls -l”, will be appended by the argument “foobar”, making the command “ls -l foobar” run in /bin/sh. However, since there is no file or directory named foobar, it will return an error of 2, meaning “if serious trouble”, according to the man page.

Problem 2 - File Descriptor Tables


File Descriptors.
parent
0
1
2
3
child
0
1
2
3
struct file
f_mode: W f_flags: APPEND f_pos: 10 f_count: 2
f_mode: R f_flags: ∅ f_pos: ∅ f_count: 2
f_mode: W f_flags: TRUNC CREAT f_pos: ∅ f_count: 3
f_mode: R f_flags: ∅ f_pos: ∅ f_count: 1
In-core inodes
x.out
in.txt
out.txt

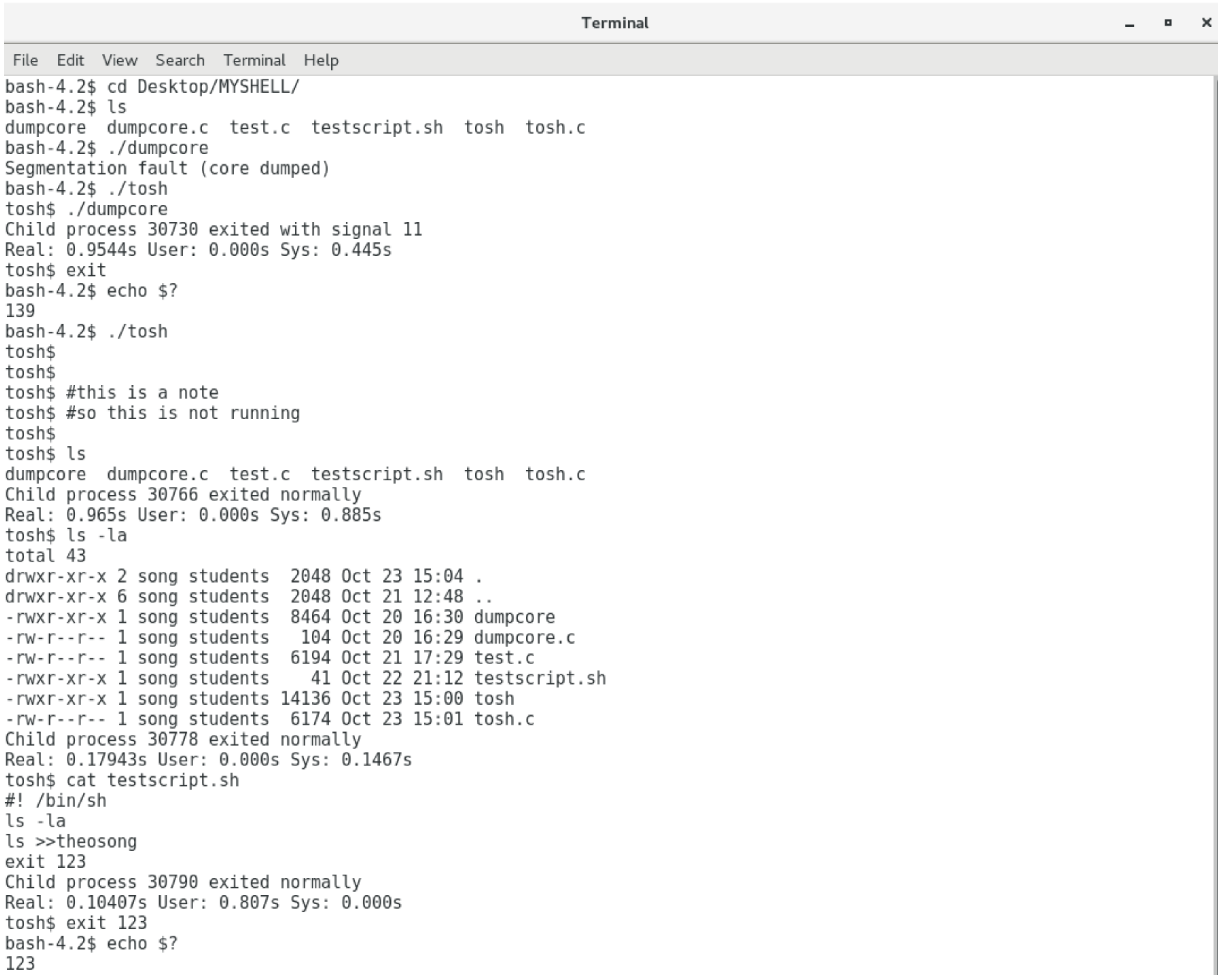
Program 3: Simple Shell Program

<Source Code>

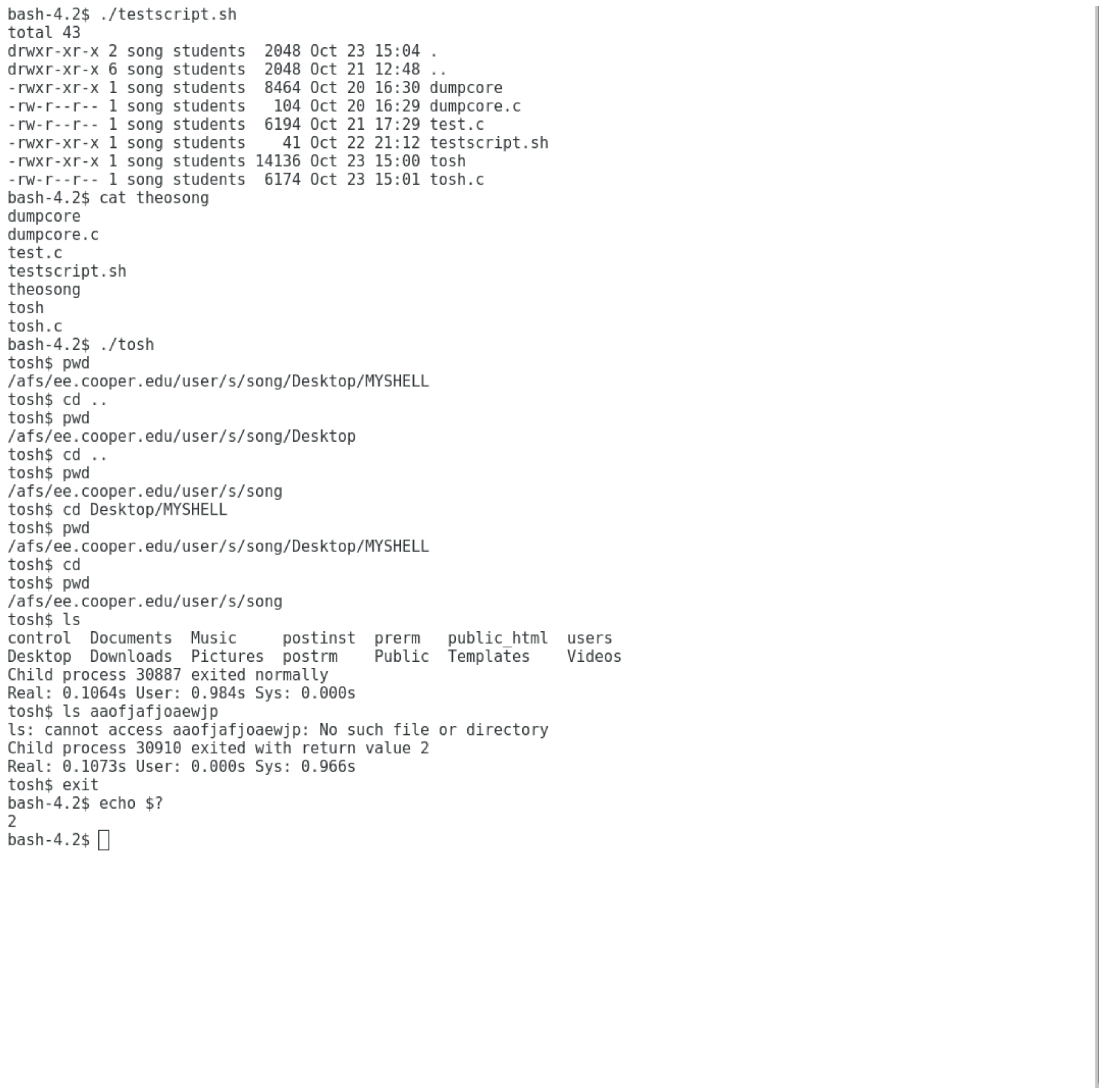
```
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>
#include <sys/types.h>
#include <sys/resource.h>
#include <sys/time.h>


void to_pwd(){
   char dir[BUFSIZ] = {};
   if(getcwd(dir, sizeof(dir))==NULL)
      perror("Error: pwd");
   else
      if((write(2,dir,sizeof(dir)))==-1)
         perror("Error: pwd: write");
      printf("\n");
}

void to_cd(char* path){
   char* tmppath = path;

   if(tmppath == NULL) tmppath = getenv("HOME");

   if(chdir(tmppath) != 0)
      perror("Error: cd");
}

void to_exit(char* exitstatus, int status){
   int exitnum;
   if(exitstatus == NULL){
      exit(status);
   }
   else{
      if(strcmp(exitstatus, "0") == 0){
         exit(0);
      }
```

```
        else{
            exitnum = strtol(exitstatus,NULL,10);
            exit(exitnum);
        }
    }
}

int parse(char* line, char* tokens[], char* redir[]){
    char* token;
    int i = 0, j = 0;
    token = strtok(line, " \r\n");

    while(token!=NULL){
        tokens[i] = token;
        i++;
        token = strtok(NULL, " \r\n");
    }

    for(int c=0; c<i; c++){
        for(int l=0; l<2; l++){
            if(tokens[c][l] == '<' || tokens[c][l] == '>'){
                redir[j] = tokens[c];
                j++;
                break;
            }
        }
    }
    redir[j++] = NULL;
    tokens[i++] = NULL;

    return i-j;
}

void redirect(char* redir){
    int fd;
    char* filename;
    if(redir[0] == '2'){
        if(redir[1] == '>' && redir[2] == '>'){
            filename = strndup(redir+3, strlen(redir));
            if((fd = open(filename, O_WRONLY|O_CREAT|O_APPEND, 0666))==-1){
                perror("Error: open");
                exit(1);
            }
        }
        else if(redir[1] == '>'){
            filename = strndup(redir+2, strlen(redir));
            if((fd = open(filename, O_WRONLY|O_CREAT|O_TRUNC, 0666))==-1){
                perror("Error: open");
                exit(1);
            }
        }
        if((dup2(fd, 2))==-1){ perror("Error: dup"); exit(1);}
        if((close(fd))==-1){ perror("Error: close"); exit(1);}
    }
    else if(redir[0] == '<'){
        filename = strndup(redir+1, strlen(redir));
        if((fd = open(filename, O_RDONLY))==-1){
            perror("Error: open");
            exit(1);
        }
        if((dup2(fd, 0))==-1){ perror("Error: dup"); exit(1);}
        if((close(fd))==-1){ perror("Error: close"); exit(1);}
    }
    else if(redir[0] == '>'){
        if(redir[1] == '>'){
            filename = strndup(redir+2, strlen(redir));
            if((fd = open(filename, O_WRONLY|O_CREAT|O_APPEND, 0666))==-1){
                perror("Error: open");
                exit(1);
            }
        }
        else{
            filename = strndup(redir+1, strlen(redir));
            if((fd = open(filename, O_WRONLY|O_CREAT|O_TRUNC, 0666))==-1){
                perror("Error: open");
                exit(1);
            }
        }
        if((dup2(fd, 1))==-1){ perror("Error: dup"); exit(1);}
        if((close(fd))==-1){ perror("Error: close"); exit(1);}
    }
    else{ perror("Error: wrong redirection command"); exit(1);}

    free(filename);
}

```

```
int to_process(char* tokens[], char* redir[], int wordnum, int status){
    char* arg[wordnum+1];
    for(int i=0; i<wordnum; i++){
        arg[i] = tokens[i];
    }
    arg[wordnum] = NULL;

    struct rusage ru;
    struct timeval tic, toc;

    gettimeofday(&tic, NULL);
    pid_t pid = fork();
    if(pid == -1){
        perror("Error: fork");
    }
    else if(pid == 0){//child
        int j=0;
        while(redir[j]){
            if(j==3) break;
            redirect(redir[j]);
            j++;
        }
        if((execvp(tokens[0], arg))==-1){
            perror("Error: exec");
            exit(127);
        }
    }
    else{//parent
        if((wait3(&status, 0, &ru))==-1)
            perror("Error: wait3");
        else{
            gettimeofday(&toc, NULL);
            if(WIFEXITED(status) && !WEXITSTATUS(status))
                fprintf(stderr, "Child process %d exited normally\n", pid);
            else if(WIFEXITED(status) && WEXITSTATUS(status)){
                fprintf(stderr, "Child process %d exited with return value %d\n", pid, WEXITSTATUS(status));
                status = WEXITSTATUS(status);
            }
            else if(WIFSIGNALED(status)){
                fprintf(stderr, "Child process %d exited with signal %d\n", pid, WTERMSIG(status));
            }
            fprintf(stderr, "Real: %ld.%.3ds ", toc.tv_sec-tic.tv_sec, toc.tv_usec-tic.tv_usec);
            fprintf(stderr, "User: %ld.%.3ds ", ru.ru_utime.tv_sec, ru.ru_utime.tv_usec);
            fprintf(stderr, "Sys: %ld.%.3ds\n", ru.ru_stime.tv_sec, ru.ru_stime.tv_usec);
        }
    }
    return status;
}

int main(int argc, char* argv[]){
    FILE* file;
    size_t len = 0;
    ssize_t charnum = 0;
    int status;

    if(argc > 1){
        if((file = fopen(argv[1], "r"))==NULL){
            perror("Error: fopen");
            exit(EXIT_FAILURE);
        }
    }
    else if(argc == 1)
        file = stdin;

    char* line = NULL;

    while(charnum!=-1){
        if(file == stdin)
            printf("tosh$ ");
        charnum = getline(&line, &len, file);
        if(charnum == -1) exit(0);
        if(line[0]=='#'||line[0]=='\n') continue;
        char* tokens[charnum];
        char* redir[charnum];

        int wordnum = parse(line,tokens,redir);

        if((strcmp(tokens[0], "exit"))==0||line == NULL)
            to_exit(tokens[1], status);
        else if((strcmp(tokens[0], "pwd"))==0)
            to_pwd();
        else if((strcmp(tokens[0], "cd"))==0)
            to_cd(tokens[1]);
        else
        status = to_process(tokens, redir, wordnum, status);
    }

    if((fclose(file)!=0)&& argc>1){
        perror("Error: fclose");
        exit(EXIT_FAILURE);
    }
    return 0;
}

```



```
bash-4.2$ cd Desktop/MYSHELL/
bash-4.2$ ls
dumpcore  dumpcore.c  test.c  testscript.sh  tosh  tosh.c
bash-4.2$ ./dumpcore
Segmentation fault (core dumped)
bash-4.2$ ./tosh
tosh$ ./dumpcore
Child process 30730 exited with signal 11
Real: 0.9544s User: 0.000s Sys: 0.445s
tosh$ exit
bash-4.2$ echo $?
139
bash-4.2$ ./tosh
tosh$
tosh$
tosh$ #this is a note
tosh$ #so this is not running
tosh$
tosh$ ls
dumpcore  dumpcore.c  test.c  testscript.sh  tosh  tosh.c
Child process 30766 exited normally
Real: 0.965s User: 0.000s Sys: 0.885s
tosh$ ls -la
total 43
drwxr-xr-x 2 song students  2048 Oct 23 15:04 .
drwxr-xr-x 6 song students  2048 Oct 21 12:48 ..
-rwxr-xr-x 1 song students  8464 Oct 20 16:30 dumpcore
-rw-r--r-- 1 song students   104 Oct 20 16:29 dumpcore.c
-rw-r--r-- 1 song students  6194 Oct 21 17:29 test.c
-rwxr-xr-x 1 song students    41 Oct 22 21:12 testscript.sh
-rwxr-xr-x 1 song students 14136 Oct 23 15:00 tosh
-rw-r--r-- 1 song students  6174 Oct 23 15:01 tosh.c
Child process 30778 exited normally
Real: 0.17943s User: 0.000s Sys: 0.1467s
tosh$ cat testscript.sh
#! /bin/sh
ls -la
ls >>theosong
exit 123
Child process 30790 exited normally
Real: 0.10407s User: 0.807s Sys: 0.000s
tosh$ exit 123
bash-4.2$ echo $?
123
```



```
bash-4.2$ ./testscript.sh
total 43
drwxr-xr-x 2 song students  2048 Oct 23 15:04 .
drwxr-xr-x 6 song students  2048 Oct 21 12:48 ..
-rwxr-xr-x 1 song students  8464 Oct 20 16:30 dumpcore
-rw-r--r-- 1 song students   104 Oct 20 16:29 dumpcore.c
-rw-r--r-- 1 song students  6194 Oct 21 17:29 test.c
-rwxr-xr-x 1 song students    41 Oct 22 21:12 testscript.sh
-rwxr-xr-x 1 song students 14136 Oct 23 15:00 tosh
-rw-r--r-- 1 song students  6174 Oct 23 15:01 tosh.c
bash-4.2$ cat theosong
dumpcore
dumpcore.c
test.c
testscript.sh
theosong
tosh
tosh.c
bash-4.2$ ./tosh
tosh$ pwd
/afs/ee.cooper.edu/user/s/song/Desktop/MYSHELL
tosh$ cd ..
tosh$ pwd
/afs/ee.cooper.edu/user/s/song/Desktop
tosh$ cd ..
tosh$ pwd
/afs/ee.cooper.edu/user/s/song
tosh$ cd Desktop/MYSHELL
tosh$ pwd
/afs/ee.cooper.edu/user/s/song/Desktop/MYSHELL
tosh$ cd
tosh$ pwd
/afs/ee.cooper.edu/user/s/song
tosh$ ls
control  Documents  Music     postinst  prerm   public_html  users
Desktop  Downloads  Pictures  postrm    Public  Templates    Videos
Child process 30887 exited normally
Real: 0.1064s User: 0.984s Sys: 0.000s
tosh$ ls aaofjafjoaewjp
ls: cannot access aaofjafjoaewjp: No such file or directory
Child process 30910 exited with return value 2
Real: 0.1073s User: 0.000s Sys: 0.966s
tosh$ exit
bash-4.2$ echo $?
2
bash-4.2$ 
```