Problem Set 3 :The shell

Scott Jin

2017-10-29

Contents

List of Figures

1 Code Listings

Scott Jin——Minishell

```c
/*
 * Mshell.c
 *
 *  Created on: Oct 22, 2017
 *  Author: scott
 *  brief: Mshell (Mini SHell)
 */
#include <sys/wait.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <sys/types.h>
char ** Mshell_split_line(char *line,char ** ReIn,char ** Re,char ** ReErr,char ** ReApp, char ** ReAErr, int *estatus);
int IOredir(const char* path, int fdnew, int oflags, mode_t mode);
int Mshell_processline(char *line, int *estatus,int *errcode);

int main(int argc, char **argv){
  //initialization
  FILE *fdIN;
  char *line=NULL;
  int readin=0,linenum=0,errcode=0,estatus=0;
  size_t buffersize=0;//let getline realloc()
  //check arguments and set fdIN
  if(argc>1){
    if(argc!=2) fprintf(stderr,"Warning:Too␣many␣arguments␣provided:␣only␣first␣one␣will␣be␣processed:%s",argv[1]);
    if((fdIN=fopen(argv[1], "r"))==NULL){
    fprintf(stderr,"Critial␣Error␣in␣opening␣target␣file␣in␣read␣mode:%s:%s\n",argv[1],strerror(errno));
    exit(EXIT_FAILURE);
    }
  }else if(argc==1){
    fdIN=stdin;
    fputs("$␣",stdout);//change PS1
  }
  //reading from target file discripter and processing
  while((readin=getline(&line, &buffersize, fdIN))>=0){//return -1 on EOF or error
    if(fdIN==stdin) fputs("$␣",stdout);
    linenum++;
    if(readin<=1||line[0]=='#'||line[readin-1]!='\n'){  //emptyline,comment,or not newline delimited
      errno=0; //setting errono for error check
      continue; //skip this line
    }
    Mshell_processline(line,&estatus,&errcode);//got line parse and put it into list
      if(estatus!=0){
          fprintf(stderr,"\nError:Execuation␣Error␣existed␣for␣line␣number␣#␣%d:␣likely␣abortted.\n",linenum);
      }
  }
  if(errno!=0){ //error occurs
    fprintf(stderr, "Error␣excuting␣getline()␣for␣line:␣%s,line␣number:%i\n",strerror(errno),linenum);
    return errno;
  }else{
    fprintf(stderr, "End␣of␣file␣read,␣exiting␣shell␣with␣exit␣code:%i\n",errcode);
    printf("\nExecuation␣Completed.\n");
  }
    return errcode;
}
int IOredir(const char* path, int fdnew, int oflags, mode_t mode){
  int fdold;
  if((fdold=open(path,oflags,mode))<0){
    fprintf(stderr,"Warning:Error␣in␣opening␣target␣file␣:%s:%s\n",path, strerror (errno));
    return EXIT_FAILURE;       //skipping redirection
  }
  if (dup2 (fdold, fdnew) < 0) {
    fprintf(stderr, "Warning:Error␣in␣dup2␣target␣file␣discripter:=%d␣dup2()␣failure:␣%s\n",
         fdold, strerror (errno));
    return EXIT_FAILURE;
  }
  if (close(fdold)<0){
    fprintf (stderr, "Warning:Error␣in␣closing␣file(%s):%s[dangling␣file␣discripter␣exits]",
        path, strerror (errno));
    return EXIT_FAILURE;
  }
  return 0;
}
int Mshell_processline(char *line, int* estatus,int *errcode){
  char * ReIn = NULL, * Re = NULL, * ReErr = NULL, * ReApp = NULL, * ReAErr = NULL; //IO
      keys
  char ** tokens=Mshell_split_line(line,&ReIn,&Re,&ReErr,&ReApp,&ReAErr,estatus);
  pid_t pid;
  struct rusage rusage;
  struct timeval t1, t2;
  if (!strcmp (tokens[0], "cd")) {
      if(tokens[1] == NULL){
        if(chdir (getenv ("HOME")) < 0){  //defualt by shell[cd ]
          fprintf (stderr, "ERROR-->cd␣failure␣in␣chdir:␣%s\n", strerror (errno));
          *estatus = 1;
          return EXIT_FAILURE;
        }
      }else{
        if(chdir (tokens[1])<0){
          fprintf (stderr, "ERROR-->cd␣failure␣in␣chdir:␣%s\n", strerror (errno));
          *estatus = 1;
          return EXIT_FAILURE;
        }
      }
    return EXIT_SUCCESS;
  }
  if (!strcmp (tokens[0], "exit")) {
    if(tokens[1] != NULL) *errcode =atoi(tokens[1]);
    if(tokens[2] != NULL) fprintf (stderr, "Warning:␣only␣first␣argument(%s)␣will␣be␣set␣to␣
        the␣error␣code␣for␣command␣exit\n",tokens[1]);
    exit(*errcode); //last errcode unless specified by the commmand
  }
  //get timestamp
  if (gettimeofday(&t1, NULL) < 0) {
      fprintf (stderr, "ERROR:␣gettimeofday␣failure␣for␣command[%s]:␣%s\n", tokens[0],
          strerror (errno));
      *estatus = 1;
      return EXIT_FAILURE;
  }
  int waitStatus,T; // T for time difference, errcode check child return signal
  pid = fork();  //fork
  if (pid == 0) {
  // Child process:IO redirection
    if (ReAErr != NULL) { //aborting if fail
      if (IOredir (ReAErr, 2, O_RDWR | O_APPEND | O_CREAT, 0666)){
        *estatus=1;
        return EXIT_FAILURE;
      }
    } else if (ReErr != NULL)
      if (IOredir (ReErr, 2, O_RDWR | O_TRUNC | O_CREAT, 0666)){
```

```
    *estatus=1;
    return EXIT_FAILURE;
   }
  if (ReApp != NULL) {
   if (IOredir (ReApp, 1, O_RDWR | O_APPEND | O_CREAT, 0666)){
    *estatus=1;
    return EXIT_FAILURE;
   }
  } else if (Re != NULL)
   if (IOredir (Re, 1, O_RDWR | O_TRUNC | O_CREAT, 0666)){
    *estatus=1;
    return EXIT_FAILURE;
   }
  if (ReIn != NULL && IOredir (ReIn, 0, O_RDONLY, 0666)){
   *estatus=1;
   return EXIT_FAILURE;
  }
  if (execvp (tokens[0],tokens)==-1) { //exec
   fprintf (stderr, "ERROR-->execvp␣failure␣for[%s]:␣%s\n", tokens[0], strerror (errno));
   *estatus=1;
   return EXIT_FAILURE;
  }
  exit(EXIT_FAILURE);//should never reach here
 } else if (pid < 0) {
 // Error forking
  fprintf (stderr, "ERROR-->fork␣failure␣for␣[%s]:␣%s\n", tokens[0], strerror (errno));
  *estatus = 1;
  return EXIT_FAILURE;
 } else {
 // Parent process
 if (wait4 (pid, &waitStatus, 0, &rusage) > 0) { //wait for the specific process
  if ((*errcode = WEXITSTATUS (waitStatus)) != 0) *estatus = 1;
  if (gettimeofday(&t2, NULL) < 0) {
      fprintf (stderr, "ERROR:␣gettimeofday␣failure␣for␣command[%s]:␣%s\n", tokens[0], strerror (errno));
   *estatus = 1;
  }
  //Printing all the info
  T = (t2.tv_sec * 1000000 + t2.tv_usec) - (t1.tv_sec * 1000000 + t1.tv_usec);
  fprintf (stderr, "\n[%s]Command␣returned␣with␣return␣code:\t%d\n",tokens[0], *errcode);
  fprintf (stderr,"Consuming␣Time:\n");
  fprintf (stderr, "␣TIME->real:\t%d.%04ds\n", T / 1000000, T % 1000000);
  fprintf (stderr, "␣TIME->usr:\t%ld.%04ds\n", rusage.ru_utime.tv_sec, rusage.ru_utime.tv_usec);
  fprintf (stderr, "␣TIME->sys:\t%ld.%04ds\n", rusage.ru_stime.tv_sec, rusage.ru_stime.tv_usec);
 }else{
  fprintf (stderr, "ERROR:␣wait4␣failure␣for␣[pid=%d␣]:␣%s\n", pid, strerror (errno));
 }//fork exec concluded here;
 }
 return EXIT_SUCCESS;
}
char ** Mshell_split_line(char *line,char ** ReIn,char ** Re,char ** ReErr,char ** ReApp, char ** ReAErr, int *estatus){
 int bufsize = 1024, position = 0;
 char* offset=0;
 char **tokens = malloc(bufsize * sizeof(char*));
 char *token, **tokens_reserve;
 if (tokens==NULL) {
  fprintf(stderr, "Critical␣Error:␣Malloc␣failture-->␣Not␣enough␣space␣left␣for␣processing");
  *estatus=1;
  exit(EXIT_FAILURE);
 }
 line[strlen (line) - 1] = 0; /* remove \n -->ls\n is no a command*/
 token = strtok(line, "␣");
 while (token != NULL) {
  if ((offset=strstr (token, "<")) && offset==token)  *ReIn = token + 1;
```

```
    else if ((offset=strstr (token, ">")) && offset==token)  *Re = token + 1;
    else if ((offset=strstr (token, "2>")) && offset==token) *ReErr = token + 2;
    else if ((offset=strstr (token, ">>")) && offset==token) *ReApp = token + 2;
    else if ((offset=strstr (token, "2>>")) && offset==token) *ReAErr = token + 3;//ignore 2>>
    else tokens[position++] = token;
    if (position >= bufsize) {
        bufsize += 1024;
        tokens_reserve = tokens;
        tokens = realloc(tokens, bufsize * sizeof(char*));
        if (tokens==NULL) {
        free(tokens_reserve);
          fprintf(stderr, "Critical␣Error:␣Malloc␣failure-->␣Not␣enough␣space␣left␣for␣processing");
          *estatus=1;
        exit(-1);
        }
    }
    token = strtok(NULL, "␣");
  }
  tokens[position] = NULL;//append the null terminator
  return tokens;
}
```

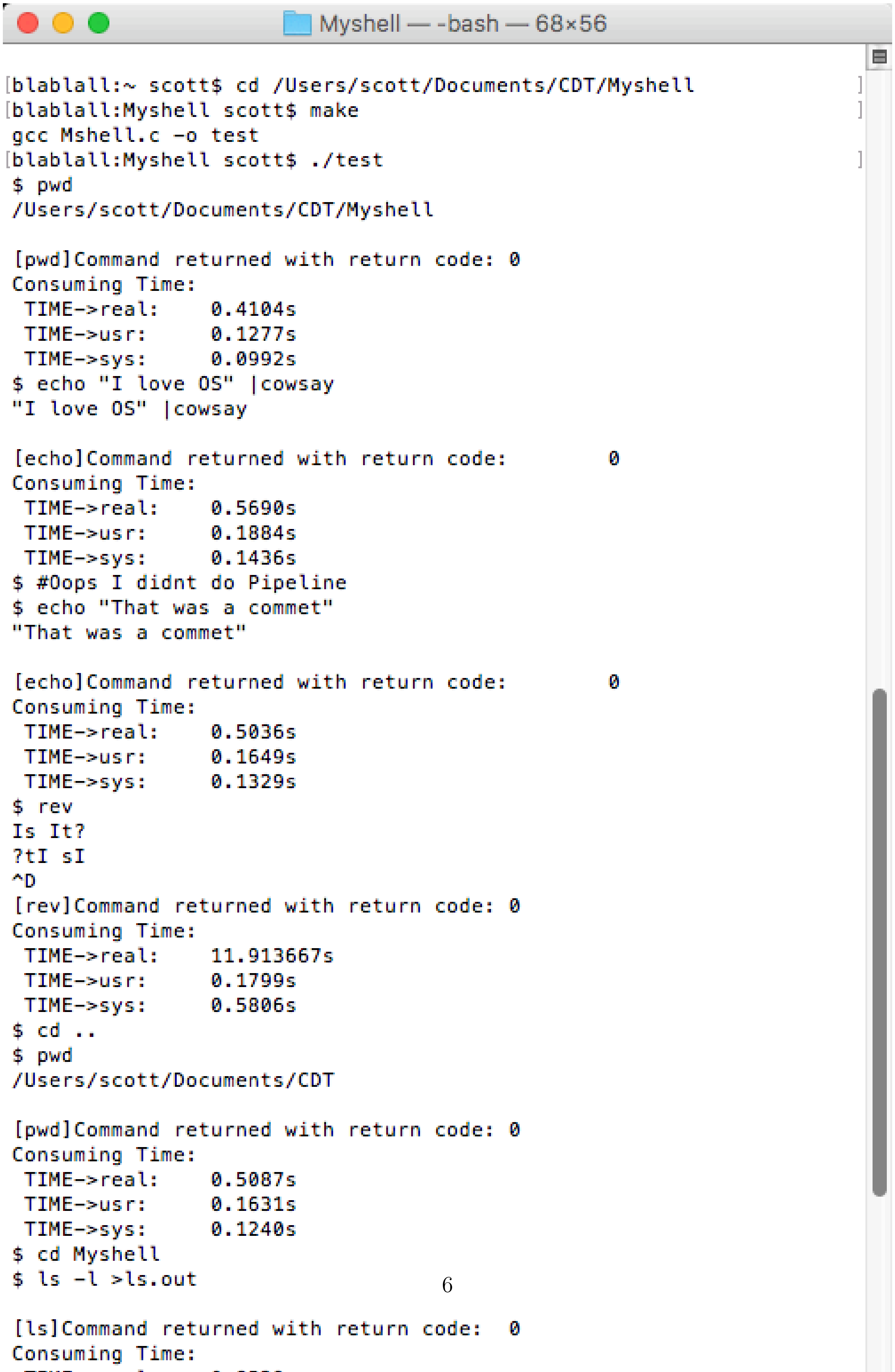
Scott Jin——Testme.sh

```
#! /Users/scott/Documents/CDT/Myshell/test
#This is an example of a shell script that your shell must execute correctly
#notice that lines starting with a # sign are ignored as comments!
#lets say this here file is called testme.sh.  you created it with say
#vi testme.sh ; chmod +x testme.sh
#you invoked it with
#./testme.sh
pwd
ls
cat >cat.out
#at this point, type some lines at the keyboard, then create an EOF (Ctrl-D)
#your shell invoked the system cat command with output redirected to cat.out
cat cat.out
#you better see the lines that you just typed!
exit 123
#after your shell script exits, type echo $? from the UNIX system shell
#the value should be 123.  Since your shell just exited, the following
#bogus command should never be seenEnd of file read, exiting shell with exit code:0
$ $
Execuation Completed.
```

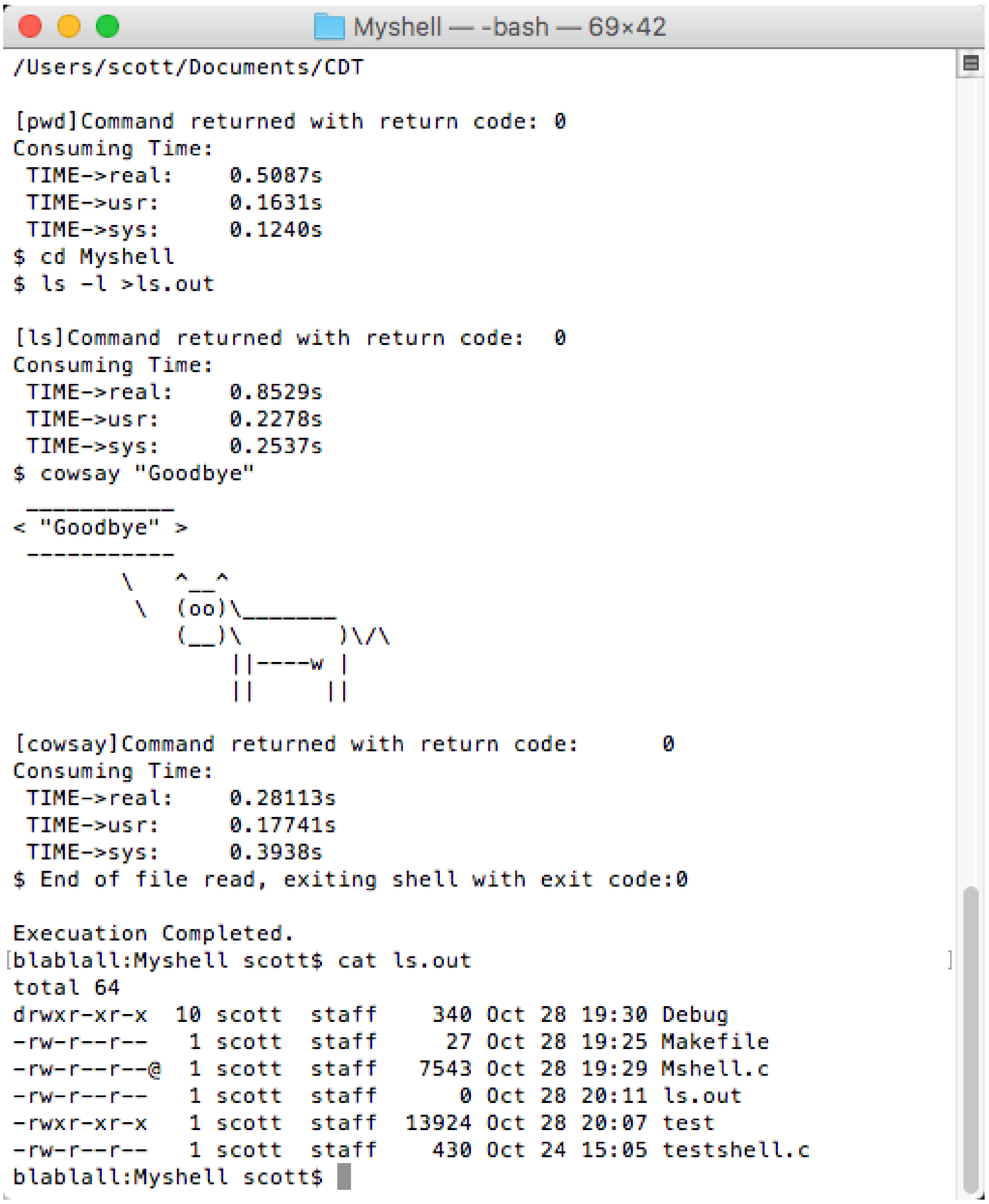
Scott Jin——Testme2.sh

```
#! /Users/scott/Documents/CDT/Myshell/test

#here is another example, say it is called test2.sh
#you invoked it with
#./test2.sh <input.txt
cat >cat2.out
#since you invoked the shell script (via the system shell such as bash)
#with stdin redirected, your shell runs cat which gets stdin from input.txt
exit
#the above exit had no specified return value, so your shell exited with 0
#again, test this with echo $?
```

2 Experimental Screenshots

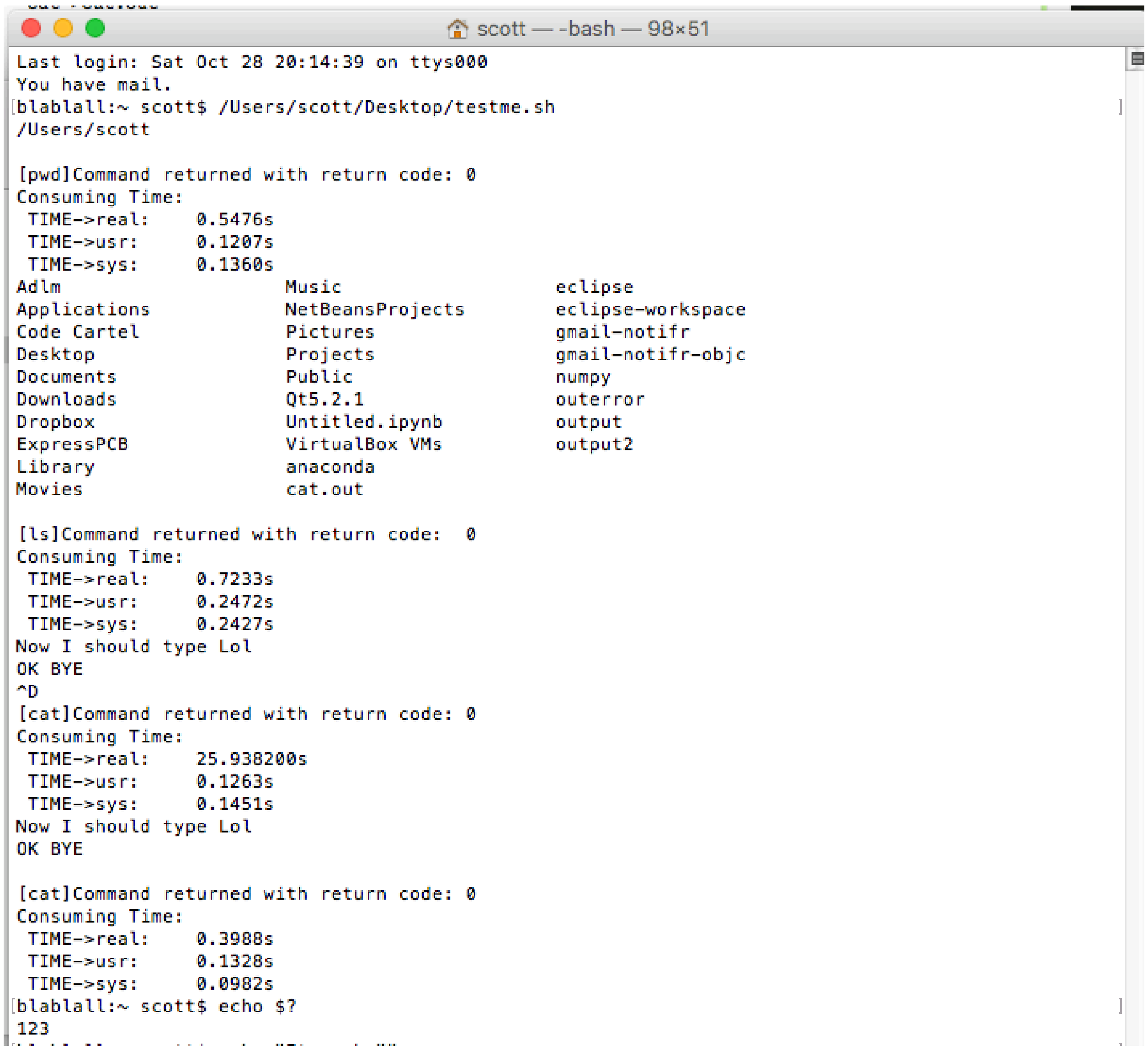


```
[blablall:~ scott$ cd /Users/scott/Documents/CDT/Myshell
[blablall:Myshell scott$ make
gcc Mshell.c -o test
[blablall:Myshell scott$ ./test
$ pwd
/Users/scott/Documents/CDT/Myshell

[pwd]Command returned with return code: 0
Consuming Time:
 TIME->real:    0.4104s
 TIME->usr:     0.1277s
 TIME->sys:     0.0992s
$ echo "I love OS" |cowsay
"I love OS" |cowsay

[echo]Command returned with return code:        0
Consuming Time:
 TIME->real:    0.5690s
 TIME->usr:     0.1884s
 TIME->sys:     0.1436s
$ #Oops I didnt do Pipeline
$ echo "That was a commet"
"That was a commet"

[echo]Command returned with return code:        0
Consuming Time:
 TIME->real:    0.5036s
 TIME->usr:     0.1649s
 TIME->sys:     0.1329s
$ rev
Is It?
?tI sI
^D
[rev]Command returned with return code: 0
Consuming Time:
 TIME->real:    11.913667s
 TIME->usr:     0.1799s
 TIME->sys:     0.5806s
$ cd ..
$ pwd
/Users/scott/Documents/CDT

[pwd]Command returned with return code: 0
Consuming Time:
 TIME->real:    0.5087s
 TIME->usr:     0.1631s
 TIME->sys:     0.1240s
$ cd Myshell
$ ls -l >ls.out

[ls]Command returned with return code:  0
Consuming Time:
```


Myshell — -bash — 69×42

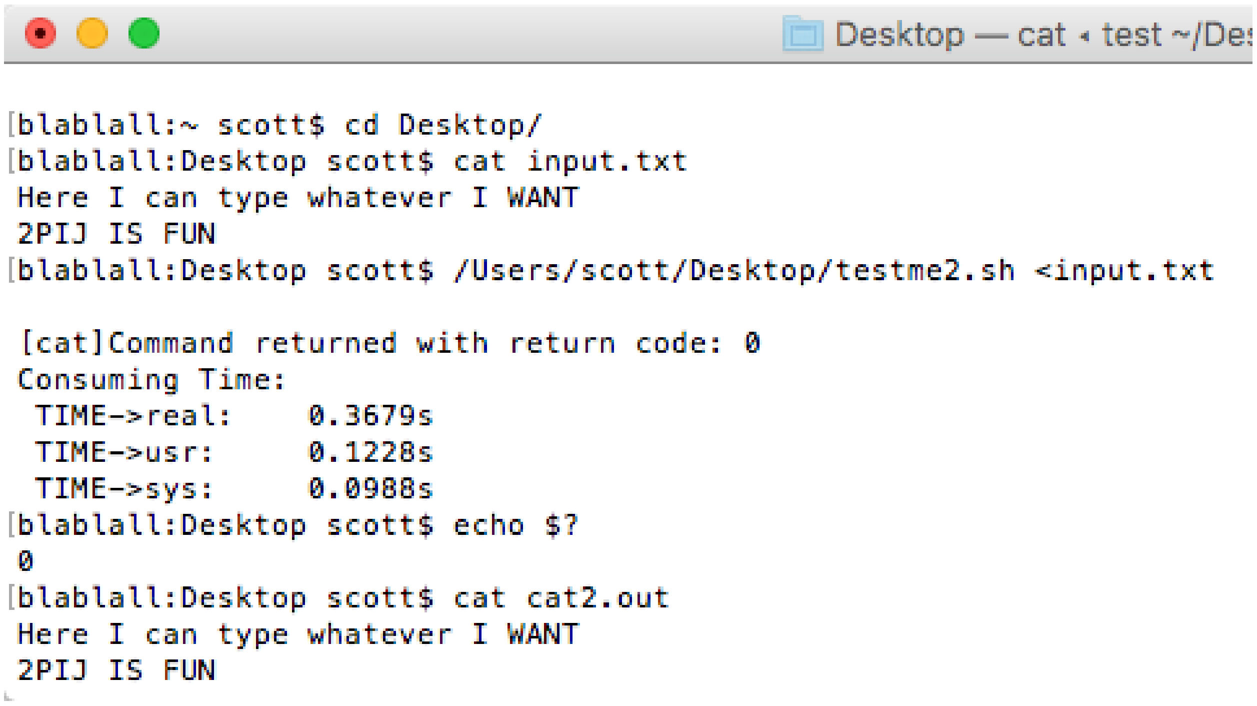
```
/Users/scott/Documents/CDT

[pwd]Command returned with return code: 0
Consuming Time:
 TIME->real:    0.5087s
 TIME->usr:     0.1631s
 TIME->sys:     0.1240s
$ cd Myshell
$ ls -l >ls.out

[ls]Command returned with return code:  0
Consuming Time:
 TIME->real:    0.8529s
 TIME->usr:     0.2278s
 TIME->sys:     0.2537s
$ cowsay "Goodbye"
 ___________
< "Goodbye" >
 -----------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||

[cowsay]Command returned with return code:      0
Consuming Time:
 TIME->real:    0.28113s
 TIME->usr:     0.17741s
 TIME->sys:     0.3938s
$ End of file read, exiting shell with exit code:0

Execuation Completed.
blablall:Myshell scott$ cat ls.out
total 64
drwxr-xr-x  10 scott  staff    340 Oct 28 19:30 Debug
-rw-r--r--   1 scott  staff     27 Oct 28 19:25 Makefile
-rw-r--r--@  1 scott  staff   7543 Oct 28 19:29 Mshell.c
-rw-r--r--   1 scott  staff      0 Oct 28 20:11 ls.out
-rwxr-xr-x   1 scott  staff  13924 Oct 28 20:07 test
-rw-r--r--   1 scott  staff    430 Oct 24 15:05 testshell.c
blablall:Myshell scott$ 
```

Figure 2: Mshell.c



```
Last login: Sat Oct 28 20:14:39 on ttys000
You have mail.
[blablall:~ scott$ /Users/scott/Desktop/testme.sh
/Users/scott

 [pwd]Command returned with return code: 0
Consuming Time:
 TIME->real:    0.5476s
 TIME->usr:     0.1207s
 TIME->sys:     0.1360s
Adlm                    Music                   eclipse
Applications            NetBeansProjects        eclipse-workspace
Code Cartel             Pictures                gmail-notifr
Desktop                 Projects                gmail-notifr-objc
Documents               Public                  numpy
Downloads               Qt5.2.1                 outerror
Dropbox                 Untitled.ipynb          output
ExpressPCB              VirtualBox VMs          output2
Library                 anaconda
Movies                  cat.out

 [ls]Command returned with return code:  0
Consuming Time:
 TIME->real:    0.7233s
 TIME->usr:     0.2472s
 TIME->sys:     0.2427s
Now I should type Lol
OK BYE
^D
 [cat]Command returned with return code: 0
Consuming Time:
 TIME->real:    25.938200s
 TIME->usr:     0.1263s
 TIME->sys:     0.1451s
Now I should type Lol
OK BYE

 [cat]Command returned with return code: 0
Consuming Time:
 TIME->real:    0.3988s
 TIME->usr:     0.1328s
 TIME->sys:     0.0982s
[blablall:~ scott$ echo $?
123
```

Figure 3: Testme.sh



```
[blablall:~ scott$ cd Desktop/
[blablall:Desktop scott$ cat input.txt
 Here I can type whatever I WANT
 2PIJ IS FUN
[blablall:Desktop scott$ /Users/scott/Desktop/testme2.sh <input.txt

 [cat]Command returned with return code: 0
 Consuming Time:
  TIME->real:    0.3679s
  TIME->usr:     0.1228s
  TIME->sys:     0.0988s
[blablall:Desktop scott$ echo $?
 0
[blablall:Desktop scott$ cat cat2.out
 Here I can type whatever I WANT
 2PIJ IS FUN
```

Figure 4: Testme2.sh