OS Assignment 3

Joseph Bentivegna

Prof. Hakner

10/29/2017

Shell Source Code:

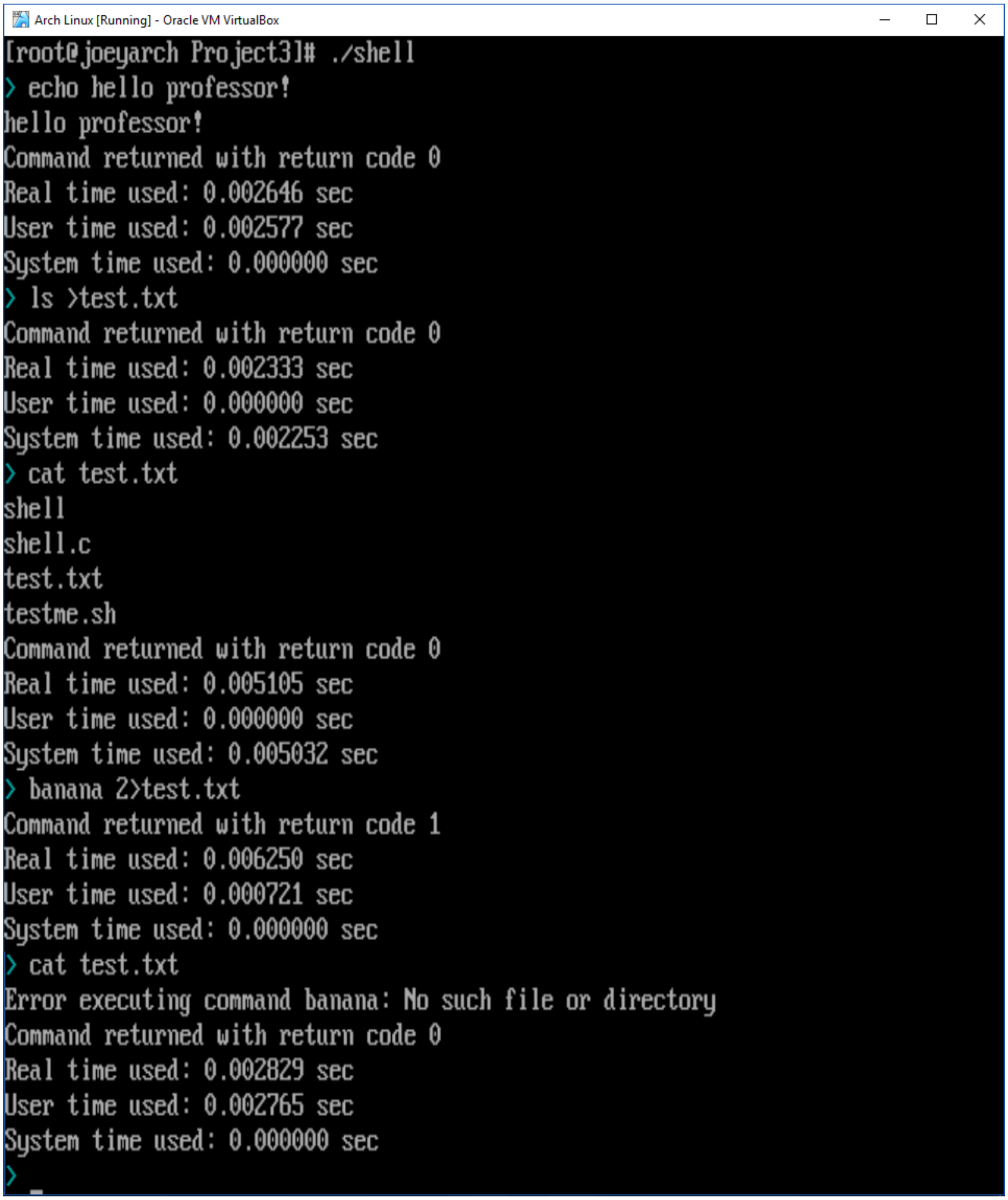
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <time.h>
#define TOKDELIM " \r\n"
#define CYN "\x1B[36m"
#define RESET "\x1B[0m"

void loop();
char *readLine();
int strChecker(char token0, char token1);
char **getArgs(char *line);
char **getRedir(char *line);
int execute(char **args, char **red);
int run(char **args, char **red);
void ioredir(char **red);
int redirect(char *path, int fdn, int options, mode_t mode);
int biCD(char **args);
int biPWD();
int biEXIT(char ** args);

void loop(FILE *source) {
    char *line;
    char **lineArgs;
    char **lineRed;
    int status;

    do {
        printf(CYN "> " RESET);
        line = readLine(source);
        lineArgs = getArgs(line);
        lineRed = getRedir(line);
        status = execute(lineArgs, lineRed);

        free(lineArgs);
        free(lineRed);
    } while (status);
}

char *readLine(FILE *source) {
    char *line = NULL;
    ssize_t bufsize = 0;
    getline(&line, &bufsize, source);
    return line;
}

int strChecker(char token0, char token1) {

    if (token0 == '<') return 1;
    if (token0 == '>') return 1;
    if ((token0 == '2') && (token1 == '>')) return 1;

    return 0;
}

char **getArgs(char *line) {
    int position = 0;
    char **elements = malloc(BUFSIZ);
    char *templine = malloc(BUFSIZ);
    char *token;

    strcpy(templine, line);
    token = strtok(templine, TOKDELIM);
    while (token != NULL) {
        if (strChecker(token[0], token[1]) == 0) {
            elements[position] = token;
            position++;
        }
        token = strtok(NULL, TOKDELIM);
    }
    elements[position] = NULL;
    return elements;
}

char **getRedir(char *line) {
    int position = 0;
    char **elements = malloc(BUFSIZ);
    char *templine = malloc(BUFSIZ);
    char *token;

    strcpy(templine, line);
    token = strtok(templine, TOKDELIM);
    while (token != NULL) {
        if (strChecker(token[0], token[1]) == 1) {
            elements[position] = token;
            position++;
        }
        token = strtok(NULL, TOKDELIM);
    }
    elements[position] = NULL;
    return elements;
}
```

```
int execute(char **args, char **red) {
    if (args[0] == NULL) {
        return 1;
    }
    if (strstr(args[0], "#")) {
        return 1;
    }
    if ((strcmp(args[0], "cd")) == 0) {
        return (biCD(args));
    }
    if ((strcmp(args[0], "pwd")) == 0) {
        return (biPWD());
    }
    if ((strcmp(args[0], "exit")) == 0) {
        return (biEXIT(args));
    }

    return run(args, red);
}

int run(char **args, char **red) {
    pid_t pid, wpid;
    int status;
    struct rusage stats;
    struct timeval start, end;

    gettimeofday(&start, NULL);
    pid = fork();
    if (pid == 0) {
        ioredir(red);
        if (execvp(args[0], args) == -1) {
            fprintf(stderr, "Error executing command %s: %s\n", args[0],
strerror(errno));
        }
        exit(EXIT_FAILURE);
    } else if (pid < 0) {
        fprintf(stderr, "Error in fork process\n");
    } else {
        wpid = wait3(&status, WUNTRACED, &stats);
        gettimeofday(&end, NULL);
        fprintf(stderr, "Command returned with return code %d\n",
WEXITSTATUS(status));
        fprintf(stderr, "Real time used: %ld.%06ld sec\n", (end.tv_sec-
start.tv_sec), (end.tv_usec-start.tv_usec));
        fprintf(stderr, "User time used: %ld.%06ld sec\n",
stats.ru_utime.tv_sec, stats.ru_utime.tv_usec);
        fprintf(stderr, "System time used: %ld.%06ld sec\n",
stats.ru_stime.tv_sec, stats.ru_stime.tv_usec);
    }
    return 1;
}

void ioredir(char **red) {
    int i = 0;
    char *curentry;
    char *curdir;
    while (red[i] != NULL) {
        curentry = red[i];
        if (strstr(curentry, "2>>")) {
            curdir = curentry + 3;
            if (redirect(curdir, 2, O_RDWR|O_APPEND|O_CREAT, 0666)) exit(1);
        } else if (strstr(curentry, ">>")) {
            curdir = curentry + 2;
            if (redirect(curdir, 1, O_RDWR|O_APPEND|O_CREAT, 0666)) exit(1);
        } else if (strstr(curentry, "2>")) {
            curdir = curentry + 2;
            if (redirect(curdir, 2, O_RDWR|O_TRUNC|O_CREAT, 0666)) exit(1);
        } else if (strstr(curentry, ">")) {
            curdir = curentry + 1;
            if (redirect(curdir, 1, O_RDWR|O_TRUNC|O_CREAT, 0666)) exit(1);
        } else if (strstr(curentry, "<")) {
            curdir = curentry + 1;
            if (redirect(curdir, 0, O_RDONLY, 0666)) exit(1);
        }
        i++;
    }
}

int redirect(char *path, int fdn, int options, mode_t mode) {
    int fd;

    if ((fd=open(path, options, mode))<0) {
        fprintf(stderr, "Can't open %s: %s\n", path, strerror(errno));
        return 1;
    }
    if (dup2(fd, fdn)<0) {
        fprintf(stderr, "Can't dup fd= %s: %s\n", fd, strerror(errno));
        return 1;
    }
    if (close(fd)<0) {
        fprintf(stderr, "Can't close %s: %s\n", path, strerror(errno));
        return 1;
    }
    return 0;
}

int biCD(char **args) {
    if (args[1] == NULL) {
        fprintf(stderr, "Error using cd command\n");
    } else {
        if (chdir(args[1]) != 0) {
            fprintf(stderr, "Error changing directory\n");
        }
    }
    return 1;
}

int biPWD() {
    char cwd[BUFSIZ];
    if ((getcwd(cwd, sizeof(cwd))) == NULL) {
        fprintf(stderr, "Error retrieving current directory\n");
    } else {
        printf("Current working dir: %s\n", cwd);
    }
    return 1;
}

int biEXIT(char **args) {
    if (args[1] == NULL) {
        exit(0);
    } else {
        exit(atoi(args[1]));
    }
}

int main(int argc, char **argv) {
    FILE *file;

    if (argc > 1) {
        if ((file = fopen(argv[1], "r")) == NULL) {
            fprintf(stderr, "Error in opening file %s: %s\n", argv[1],
strerror(errno));
            return -1;
        } else {
            loop(file);
        }
    } else if (argc == 1) {
        file = stdin;
        loop(file);
    }
    return 0;
}
```

Script Interpreter Test Source Code:

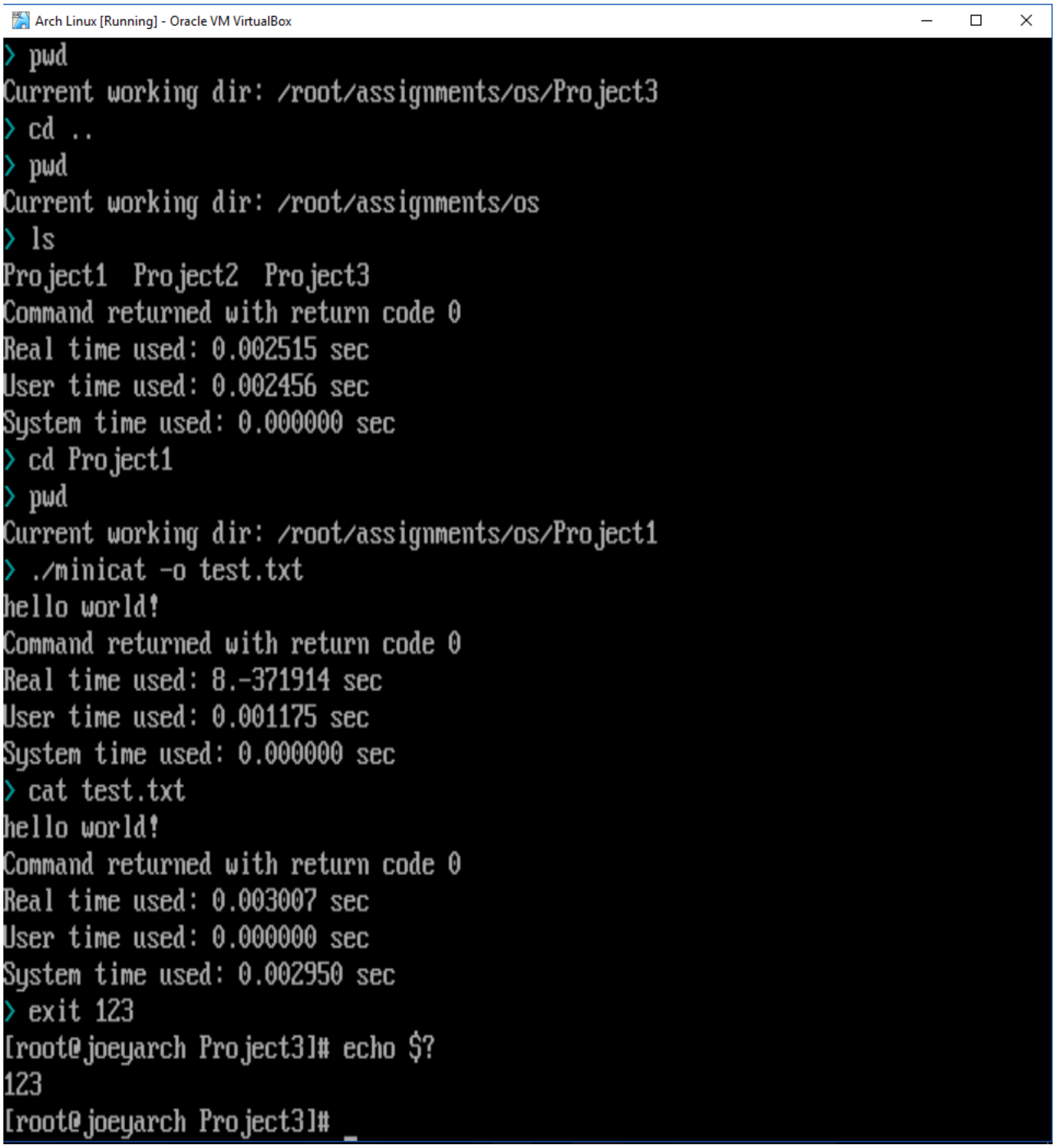
```
#!/root/assignments/os/Project3/shell
cat >cat.out
#this is a comment!
cat cat.out
exit 123
#bogus command should never be seen
```

Shell functioning properly:

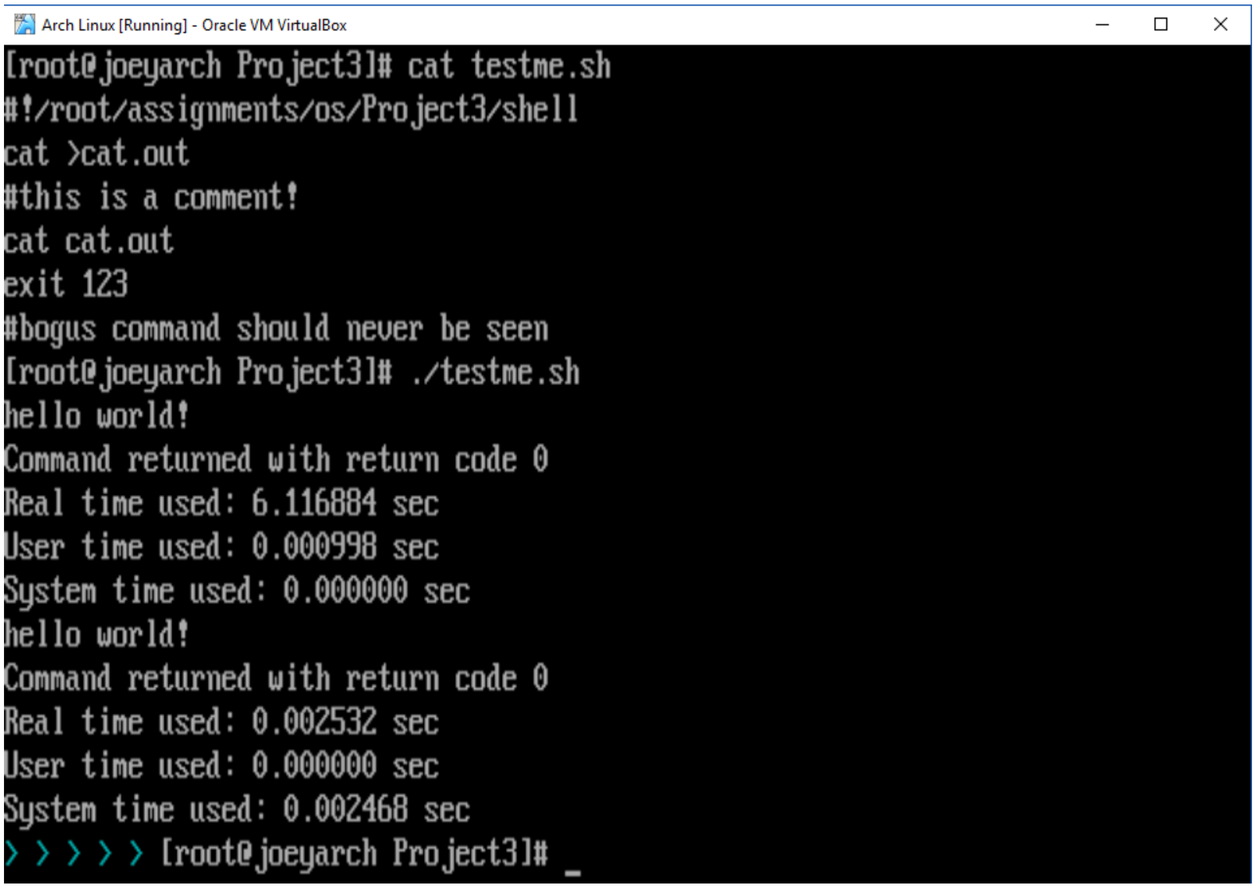


Arch Linux [Running] - Oracle VM VirtualBox

```
[root@joeyarch Project3]# ./shell
> echo hello professor!
hello professor!
Command returned with return code 0
Real time used: 0.002646 sec
User time used: 0.002577 sec
System time used: 0.000000 sec
> ls >test.txt
Command returned with return code 0
Real time used: 0.002333 sec
User time used: 0.000000 sec
System time used: 0.002253 sec
> cat test.txt
shell
shell.c
test.txt
testme.sh
Command returned with return code 0
Real time used: 0.005105 sec
User time used: 0.000000 sec
System time used: 0.005032 sec
> banana 2>test.txt
Command returned with return code 1
Real time used: 0.006250 sec
User time used: 0.000721 sec
System time used: 0.000000 sec
> cat test.txt
Error executing command banana: No such file or directory
Command returned with return code 0
Real time used: 0.002829 sec
User time used: 0.002765 sec
System time used: 0.000000 sec
> _
```


Arch Linux [Running] - Oracle VM VirtualBox

```
> pwd
Current working dir: /root/assignments/os/Project3
> cd ..
> pwd
Current working dir: /root/assignments/os
> ls
Project1  Project2  Project3
Command returned with return code 0
Real time used: 0.002515 sec
User time used: 0.002456 sec
System time used: 0.000000 sec
> cd Project1
> pwd
Current working dir: /root/assignments/os/Project1
> ./minicat -o test.txt
hello world!
Command returned with return code 0
Real time used: 8.-371914 sec
User time used: 0.001175 sec
System time used: 0.000000 sec
> cat test.txt
hello world!
Command returned with return code 0
Real time used: 0.003007 sec
User time used: 0.000000 sec
System time used: 0.002950 sec
> exit 123
[root@joeyarch Project3]# echo $?
123
[root@joeyarch Project3]# _
```


Arch Linux [Running] - Oracle VM VirtualBox

```
[root@joeyarch Project3]# cat testme.sh
#!/root/assignments/os/Project3/shell
cat >cat.out
#this is a comment!
cat cat.out
exit 123
#bogus command should never be seen
[root@joeyarch Project3]# ./testme.sh
hello world!
Command returned with return code 0
Real time used: 6.116884 sec
User time used: 0.000998 sec
System time used: 0.000000 sec
hello world!
Command returned with return code 0
Real time used: 0.002532 sec
User time used: 0.000000 sec
System time used: 0.002468 sec
> > > > > [root@joeyarch Project3]# _
```