**操作系统课程设计实验报告**

——实验一：shell实验

负责人姓名：李文涛

学号：14061217

日期：2016.4.9

**小组成员**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 姓名 | 学号 | 实验分工 |
| 1 | 李文涛 | 14061217 | 实验一 |
| 2 | 张东鑫 | 14061153 | 实验二 |
| 3 | 李康宇 | 14061172 | 实验三 |
| 4 | 岳明哲 | 14061173 | 实验四 |

目录

[1.实验目的 4](#_Toc446001831)

[2.需求说明 4](#_Toc446001832)

[2.1基本要求 4](#_Toc446001833)

[2.2 提高要求 4](#_Toc446001834)

[2.3 完成情况 4](#_Toc446001835)

[3.设计说明 5](#_Toc446001836)

[3.1 程序流程图 5](#_Toc446001837)

[3.2基本要求实现说明 5](#_Toc446001838)

[3.3 提高要求实现说明 5](#_Toc446001839)

[4.收获和感想 5](#_Toc446001840)

# 1.实验目的

1.学习Linux相关软件工具的使用

2.熟悉使用Linux中YACC工具进行语法分析的基本方法

3.运用man帮助手册查询相关命令

4.理解并发程序的同步问题

5.学习POSIX/UNIX系统调用的使用

6.掌握进程控制和进程间通信的方法

# 2.需求说明

## 2.1基本要求

1.程序能够正常运行

2.能够执行fg，bg, cd, history, exit等内部命令

3.能够执行外部程序命令，命令可以带参数

4.使用I/O 重定向

5.支持前后台作业，提供作业控制功能，包括打印作业的清单，改变当前运 行作业的前台/后台状态，以及控制作业的挂起，终止和继续运行

## 2.2 提高要求

1.尝试对YACC语法分析的文法进行进一步的修改与完善

2.尝试在Linux下将Lex和YACC结合起来进行词法和语法分析

3.对其他常用的内部命令进行实现，并可以尝试考虑对通配符的支持和实现

4.实现对管道的支持

5.考虑并实现组合键Ctrl+C命令

## 2.3 完成情况

完成了以下功能：

1.基本要求

2.组合键Ctrl+C

3.实现了kil(在user-sh里面体现为mykill)，echo(在user-sh里面体现为 myecho)这两个内部命令

4.利用YACC和lex实现了一个支持四则运算的计算器

# 3.设计说明

## 3.1 程序流程图

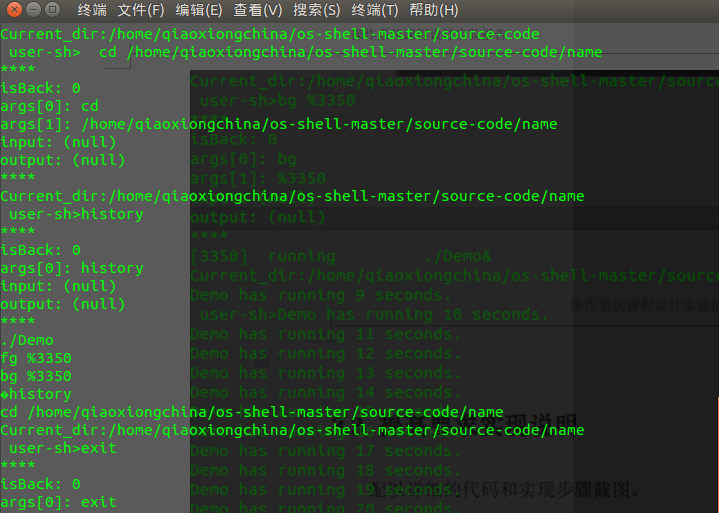
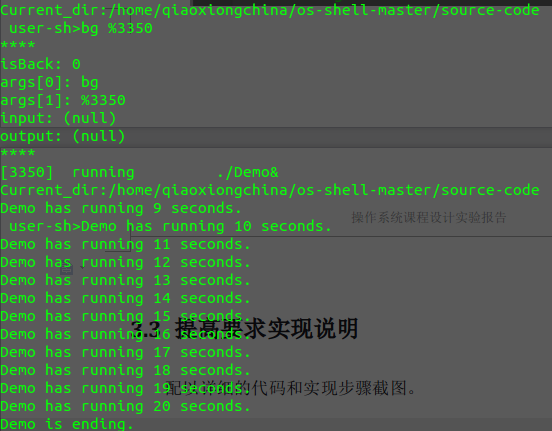
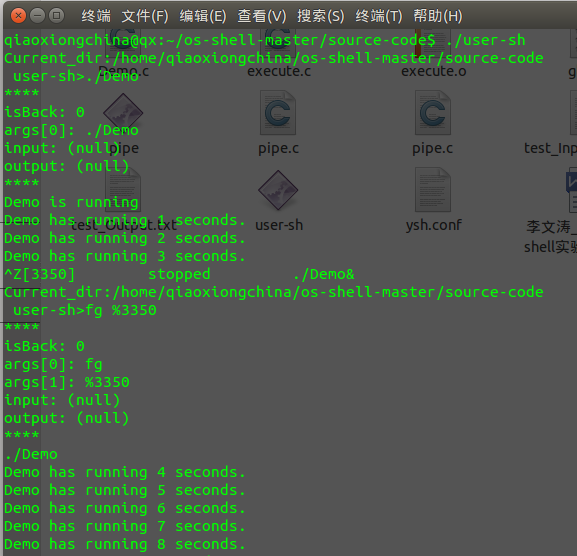
1.先对环境进行初始化

2.对输入的命令进行解析

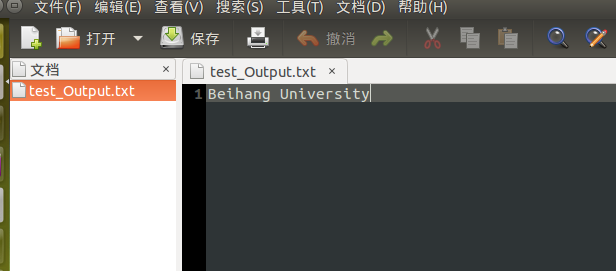
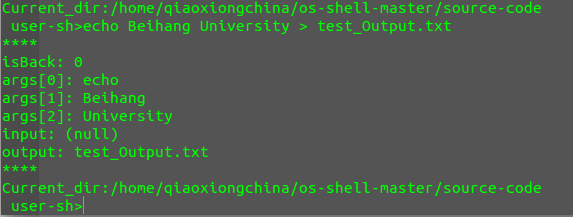
3.判断输入的命令是什么，然后调用相关的函数执行命令

## 3.2基本要求实现说明

1.程序的正常运行以及内部命令的实现



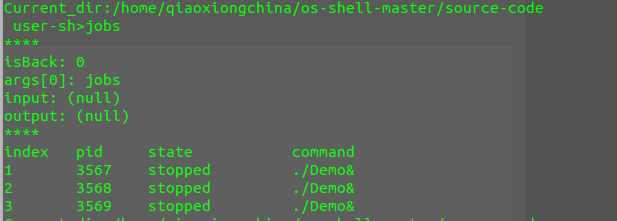
2.I/O重定向



3.前后台作业



作业清单



## 3.3 提高要求实现说明

1.实现Ctrl+C

void ctrl\_C(){

Job \*now = NULL;

if(fgPid == 0){ //前台没有作业则直接返回

return;

}

now = head;

while(now != NULL && now->pid != fgPid)

now = now->next;

if(now == NULL){ //未找到前台作业，则根据fgPid添加前台作业

now = addJob(fgPid);

}

//修改前台作业的状态及相应的命令格式，并打印提示信息

strcpy(now->state, STOPPED);

now->cmd[strlen(now->cmd)] = '&';

now->cmd[strlen(now->cmd) + 1] = '\0';

printf("[%d]\t%s\t\t%s\n", now->pid, now->state, now->cmd);

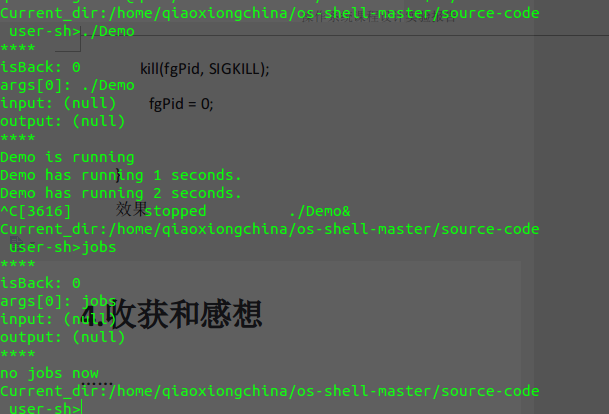
//发送SIGSTOP信号给正在前台运作的工作，将其停止

kill(fgPid, SIGKILL);

fgPid = 0;

}

效果



2.实现了kil(在user-sh里面体现为mykill)，echo(在user-sh里面体现为 myecho)这两个内部命令

1) mykill

else if(strcmp(cmd->args[0], "mykill") == 0){// mykill 命令

temp = cmd->args[1];

if( temp != NULL){

pid = str2Pid(temp,0,strlen(temp));

kill(pid,SIGKILL);

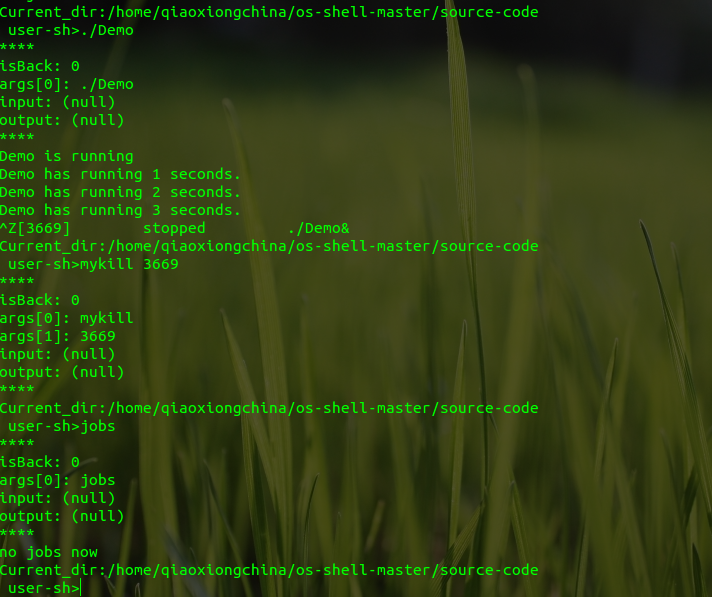
}

else{

printf("mykill; ilegal parameter, right form:mykill <int>\n");

}

效果



2)myecho

else if (strcmp(cmd->args[0], "myecho") == 0){//myecho命令

if( cmd->args[1] != NULL) {

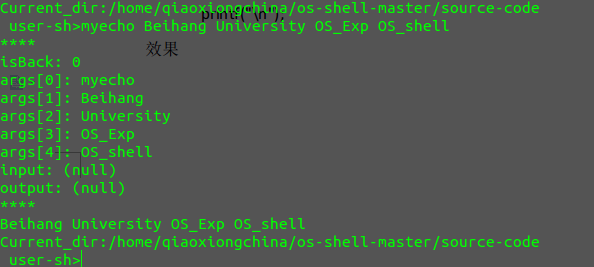
for(i = 1; cmd->args[i] != NULL; i++)

printf("%s ", cmd->args[i]);

}

printf("\n");

效果



4.在Linux下将Yacc和lex结合起来进行词法和语法分析——实现了简单 的加减法计算器

1) cal.lex

%{

#include "cal.tab.h"

#include <stdio.h>

#include <string.h>

extern int yylval;

%}

%%

[0-9]+ {yylval = atoi(yytext); return NUMBER;}

[ \t] {;}

\n {return 0;}

. {return yytext[0];}

%%

int yywrap(){

return 1;

}

2)cal.y

%token NAME NUMBER

%%

statement: NAME '=' expression

| expression { printf("=%d\n", $1 );}

;

expression: expression '+' NUMBER { $$ = $1 + $3;}

| expression '-' NUMBER { $$ = $1 - $3;}

| NUMBER { $$ = $1;}

;

%%

int main() {

yyparse();

return 0;

}

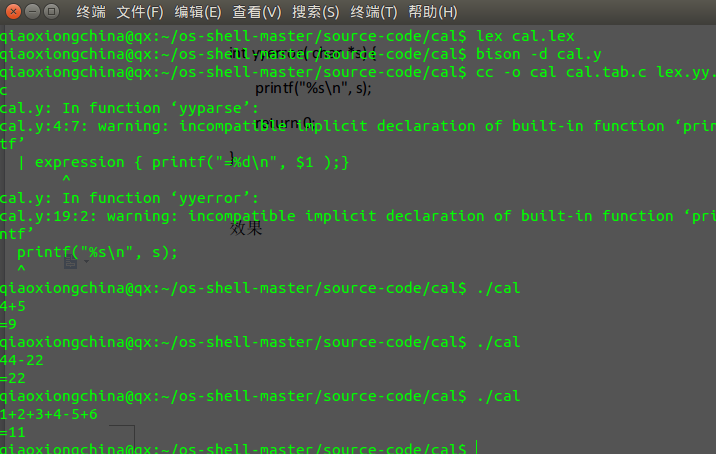
int yyerror( char \*s) {

printf("%s\n", s);

return 0;

}

效果



# 4.收获和感想

1.收获：懂得了很多Linux的知识，对进程等概念有了更加深入形象的认识，学 会了如何在Linux下编程，学会了如何与团队成员合作

2.感想：由于是初次接触Linux和Shell，上手比较慢，而且由于其他课程的压力较大，不能将大部分的时间投入与OS实验，所以提高要求只完成了一半。这给了我们一个教训就是：凡事宜早不宜迟，在自己不会的情况下要多向他人讨教。