**操作系统课程设计实验报告**

——实验一：shell实验

负责人姓名：黎柏文

学号：14061168

日期：2016.4.8

**小组成员**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 姓名 | 学号 | 实验分工 |
| 1 | 黎柏文 | 14061168 | 整体设计、lex/yacc语法分析 |
| 2 | 黄顺达 | 14061159 | 信号量ctrl+c |
| 3 | 祝星魁 | 14061165 | 管道实现 |
| 4 | 吕文超 | 14061174 | 内部命令 |

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（如gcc、gdb和make）

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

程序能够正常运行

能够执行***fg、bg、cd、history、exit***等内部命令

能够执行外部程序命令，命令可以带参数

使用I/O重定向

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、中止和继续运行

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结合起来使用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与实现

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.基本要求所有功能

2.提高要求里面除了通配符以外的所有功能

实验完成过程

1.阅读助教给的样例代码，搞清楚逻辑关系

2.自己尝试一些例子，熟悉open,read,write,dup等函数，信号量的使用等

3.整体设计：将程序架构分成两大部分：词法分析和命令执行

设计数据结构：命令的结构、作业列表的结构、程序调用关系等

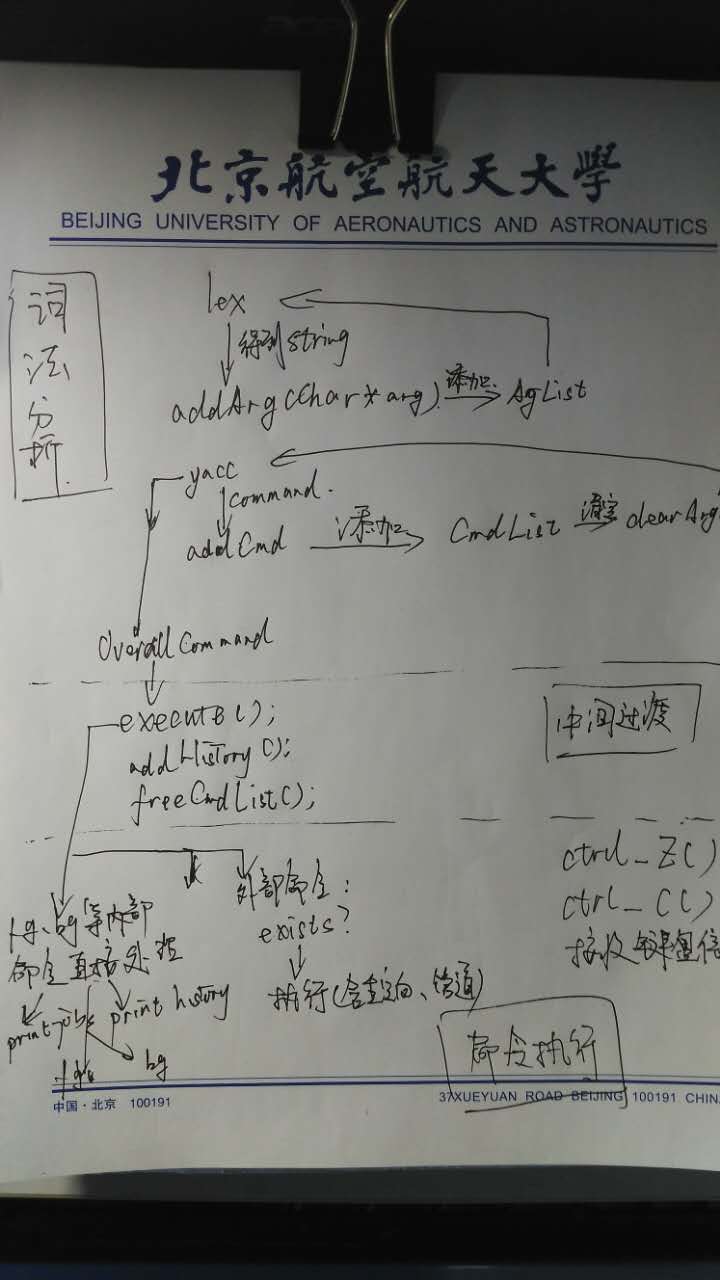
4.讨论和分工

5.编写程序和测试：首先编写yacc和lex进行词法分析的部分，编写global.h和execute.c的基本内容，初步形成整体结构，编写compile.sh(我们没有使用makefile的方法而是直接写了脚本来编译)；然后编写执行命令的部分，从简单的开始入手，先实现能执行简单命令，然后实现区分前后台，然后实现接收键盘信号，接着实现重定向，最后实现管道

6.讨论总结和书写实验报告

3.设计说明

3.1 程序流程图



3.2基本要求实现说明

命令执行主函数：execute()

外部命令执行函数：execOuterCmd()

内部命令执行函数：printHistory(),printJobs(),fg(),bg()

作业相关函数：addJob(),rmJob()

3.3 提高要求实现说明

yacc和lex进行词法分析：mysh.lex，mysh.y

管道的实现：通过增加两个文件pipeintmp和pipeouttmp让检测到管道的时候从这两个里面读写，每个指令读之前先rename一下pipeintmp和pipeouttmp以实现读取上一条指令写的内容

内部命令的增加：增加了命令pwd，通过调用函数getcwd来实现

ctrl+C组合键的实现：增加了接收信号量SIGINT的函数来实现ctrl+Z

4.收获和感想

这次实验中我是主要负责人，负责主要设计和分工，也是代码贡献量最多的那个所以可以说对整个实验都了如指掌，收获也很多。

首先是具体的收获：

1。在使用lex和yacc的过程中我感觉这个东西很适合进行词法分析，可以指定严格的语法规范并且在遇到不同情况的时候调用不同函数，还是很方便的。并且就像yacc的名字”Yet Another Compiler Compiler”所说，这东西应该会很适合用来写编译器，并且这货还能递归地定义语法规则！虽然了解还不是很多但已经感受到这东西的威力了

2.语言和系统的关系：通过这次实验，我感受到编程语言是和操作系统紧密结合在一起的，Unix/Linux是用C语言来写的，所以在这上面写C语言程序、调用相关系统调用是非常自然而顺畅的

3.有关进程的知识：我通过这次实验对操作系统的进程之间的关系有了更加具体的理解，再结合上OO课上开始写多线程的程序，我愈发地感受到多线程/进程的巨大威力了

然后是有关学习方法和一些抽象的编程知识的收获：

1.在接触一样新的东西的时候，应该先从简单例子入手，等自己熟悉了明白了怎么回事了，再去编写大型的软件，这样才比较自然而不会有恐惧感

2.程序应当做好封装、降低耦合度、写好注释。我这次写分了两周来写，上一周花了1天，这一周花了2天，中间时隔一个星期，幸亏我的注释写得还算完善，而且在上一周写yacc和lex的时候已经完整地对那一部分程序进行了测试，不然我真的无从下手

**5.遇到的问题及解决办法**

**中间遇到了许多问题，但大都可以通过几种方法解决：**

**1.百度**

**2.问同学**

**3.将问题分解成简单的情况，解决了再回到复杂的情况中去**