

Linux命令解释程序设计与实现实验报告

|  |  |
| --- | --- |
| 学 号： |  |
| 姓 名： |  |
| 学 院： |  |
| 提交日期： |  |

1. **实验目的**

探索、理解并掌握操作系统命令解释器的设计原理和实现机制，基于Linux内核进行相应命令解释程序的设计和实现，并在 Linux 操作系统平台上加以测试验证。

1. **实验内容**

分析、设计与实现基于Linux内核的命令解释程序(Shell)，主要包括系统环境变量的设置和初始化、系统命令提示符显示、命令辨别解析（区分内部命令与外部命令及不同内部命令)、典型内部命令（譬如显示指定目录下文件列表、显示文本文件内容、文件拷贝、文件删除、空文件创建、日期设置/显示）处理等功能，并在 Linux操作系统上测试验证。

1. **实验要求**

Linux命令解释程序功能设计要求:

(1）选取和设计实现一组内部命令(五条以上);

(2）外部命令执行采用直接调用exec系统调用的方式来实现;

(3）至少一条内部命令采用直接调用相应系统调用的方式来实现;

(4）系统环境变量(至少包括用户主目录HOME和可执行程序搜索路径目录PATH)支持;

(5）在 Linux 操作系统上启用（或替换原命令解释程序Shell）并测试验证。

**4.实验环境**

虚拟机软件：VMware

操作系统：Ubuntu

编译器：gcc

编辑器：vim

**5.实验分析**

Shell直译为壳，它在Linux系统里可以认为是操作系统的外壳，为用户和系统内核中间的一个接口。它既有着连接用户和系统内核的能力，也可以保护系统内核不受非法操作的干扰，防止操作系统收到破坏。在我们用户输入一个指令时，shell作为一个命令行解释器，将我们的指令进行解释，再决定如何提交给内核去执行。

Shell里可以既可以处理内部指令，也可以处理外部指令。对于实际使用的用户来讲，他们不必关心于输入的指令是内部指令还是外部指令，但我们这次实验要完成的内容包括了内外部指令的处理。

好的一点是，这些基础的指令并不复杂，在封装好之后我们可以采取直接调用或者间接调用去完成此次实验，并且如果判断到输入的指令不在我们所实现的指令之中的话，可以直接返回错误，提示没有这条命令。

**6.具体实现的内容**

1.内部命令：pwd,cd,rename,exit,clear,help,echo(其中两条为直接系统调用实现)

2.外部命令：ls,cp

3.指示命令：包括提示输出，判别指令合法性以及处理指令结果提示。

**7.实现思路以及关键点分析**

（1）本次实验我认为的难点和关键点并不在于指令内部的设计，因为部分指令可以系统调用，部分指令的实现起来所用到的数据结构或者算法并不复杂。我认为最重要的是在于如何达到执行命令的效果。

经过查阅资料可知，Linux系统中自带的system（command）函数是可以完成这个效果的，但通过资料显示和实际的实验可以发现，这个函数很容易出错，并且不是很安全，所以这就需要我们设计一个命令分析并且输入的函数，然后调用这个函数，最终进入系统实现。

我所设计的这个函数明明为analysis\_command()函数就是为了实现这个功能。通过简单的readline()获取用户输入的字符串，然后进行分析。这个函数最后通过使用exec函数的使用来实现作用。（此处的exec不是一个函数，而是以exec开头的一类函数）。

这部分具体代码如下：



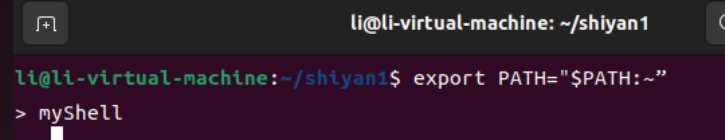
（2）主函数部分

主函数部分设计思路大致为：最初预打印命令提示符，然后等待接收用户命令，之后判断并处理命令。整体的逻辑与调用的函数都十分明确，代码自然顺理成章：

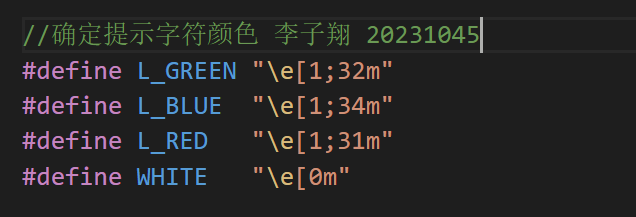


**8.功能测试**

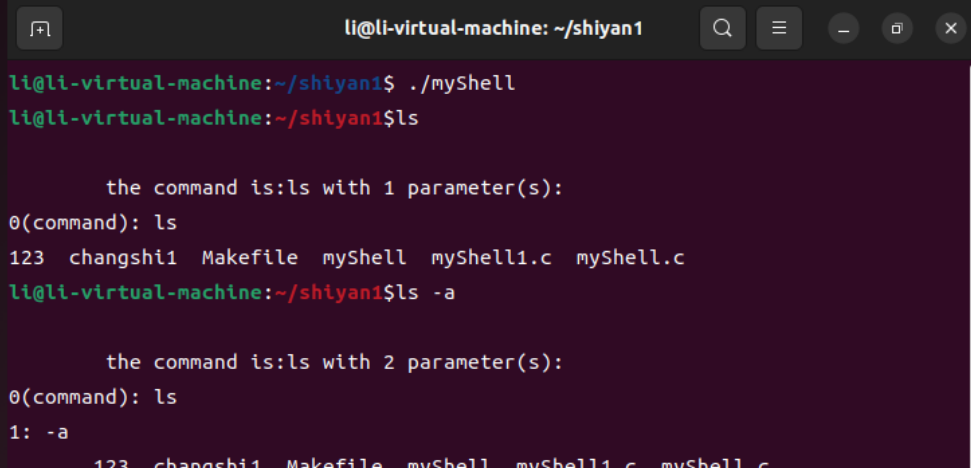
测试之前，首先用export命令使得文件添加到环境变量之中：

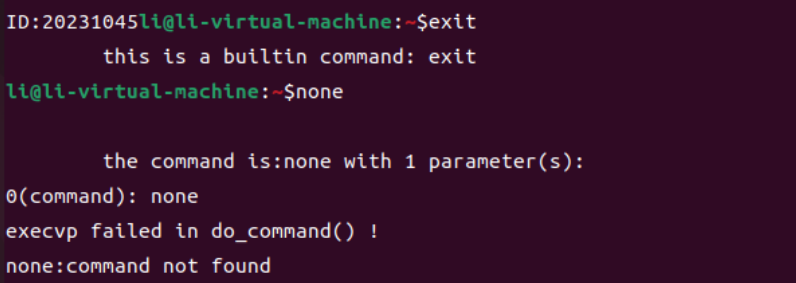
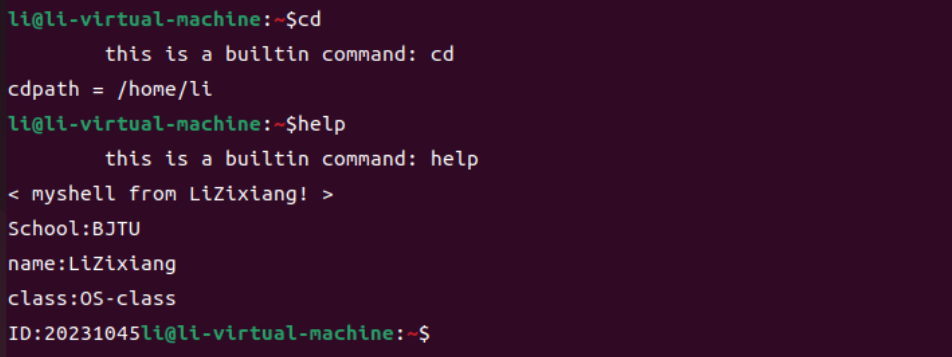
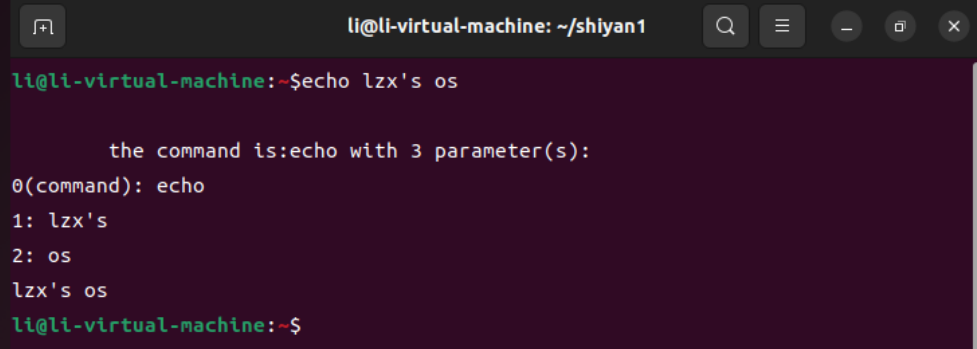
****

然后测试中途，发现命令提示符字体的颜色与系统的相同，不便区分，于是更改了命令提示符的颜色：



然后进行指令的测试：





**9.心得体会**

本次实验是操作系统课程的一个基础实验，我通过了解Linux中shell的作用以及用法，结合自己对shell功能的认识，逐步编写了一个类似于shell的程序，并且通过这个程序实现了期望实现的指令。

这次实验中，我遇到了很多问题。从最初始的虚拟机的环境的搭建，到了解Linux所需要的命令行输入，最后编写程序，并编写Makefile文件使得成功运行程序。特别是在myshell程序的编写过程中，我开始对于指令的编写毫无头绪，在尝试并成功编写了一个指令之后，之后的指令也通过类似的方法同样可以编写出来。最后在修改了提示界面之后整个程序基本圆满完成。

通过这次实验，我学习到了很多。首先更加深刻地理解了操作系统的命令解释器的设计原理和具体实现的方法，再通过自己对Linux相关指令的设计和实现之后，掌握的更加娴熟。并且，这次实验也锻炼了我测试和完善程序的能力，通过一步一步的修改，使之达到了预计的效果，收获颇多。