**青 岛 科 技 大 学**

**课程设计（Unix）设计报告**

**shell编程用户管理**

**题 目 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

杨爱光

**指导教师\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

梁炜婧

**学生姓名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**学生学号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1408080136

信息科学技术学

计算机科学与技术

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_院（部）\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_专业\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_班**

开发1

**\_\_\_\_\_\_年 \_\_\_月 \_\_\_日**

9

7

2018

|  |  |  |  |
| --- | --- | --- | --- |
| 课程设计地点 | 弘毅楼D410、411 | 课程设计时间 | 2018.5.21­—2018.7.15 |
| 课程设计  任务及要求 | 能够综合运用所学知识，在Unix/Linux平台下，使用某种开发工具和相关技术，针对Unix/Linux应用领域如运营维护、服务器设计等案例进行需求分析、方案设计和测试评估；并能够运用工程化方法进行系统的开发和设计，并能规范撰写设计报告；熟悉行业相关规范和岗位职责，培养自主分析和解决实际问题的能力。 | | |
| 指导教师评语 | 指导教师签字： | | |
| 验收（答辩）意见 | 验收（答辩）人员签字： | | |
| 系（教研室）  意 见 | 教研室主任签字： | | |
| 成 绩  （百分制） |  | | |

目录

[1设计要求 1](#_Toc518917644)

[2系统需求分析 2](#_Toc518917645)

[2.1 menu.sh文件实现功能 2](#_Toc518917646)

[2.1.1查询功能 2](#_Toc518917647)

[2.2 add.sh文件实现功能 2](#_Toc518917648)

[2.3 delete.sh文件实现功能 2](#_Toc518917649)

[2.4退出 3](#_Toc518917650)

[3系统设计 4](#_Toc518917651)

[3.1功能图 4](#_Toc518917652)

[3.2流程图 4](#_Toc518917653)

[3.2.1 menu菜单 4](#_Toc518917654)

[3.2.2 add模块 5](#_Toc518917655)

[3.2.3 delete模块 5](#_Toc518917656)

[4系统数据结构 7](#_Toc518917657)

[4.1文件结构 7](#_Toc518917658)

[4.2判断passwd文件是否存在的函数 7](#_Toc518917659)

[5系统实现 8](#_Toc518917660)

[5.1 menu文件 8](#_Toc518917661)

[5.2 add文件 11](#_Toc518917662)

[5.3 delete文件 16](#_Toc518917663)

[6系统测试 19](#_Toc518917664)

[6.1功能测试 19](#_Toc518917665)

[6.1.1 显示当前记录和格式化显示 19](#_Toc518917666)

[6.1.2 显示用户名和用户ID及格式化显示 19](#_Toc518917667)

[6.1.3 查询特定用户信息 20](#_Toc518917668)

[6.1.4 添加用户 20](#_Toc518917669)

[6.1.5 删除用户 20](#_Toc518917670)

[6.1.6 查询密码 20](#_Toc518917671)

[6.2鲁棒性测试 21](#_Toc518917672)

[6.2.1 输入错误 21](#_Toc518917673)

[6.2.2 添加用户测试 21](#_Toc518917674)

[7总结 23](#_Toc518917675)

[参考文献 24](#_Toc518917676)

1. 设计要求

掌握linux操作系统中shell脚本编程技术中的：变量定义、变量使用、循环控制结构、条件选择结构、用户交互、文件读写、文本内容分析以及linux重要命令的使用等。以验证学生对于shell脚本编程的掌握情况。要求学生能够利用shell脚本编程实现较为基本的功能。

Linux作为多用户、多任务的操作系统，其系统资源是所有用户共享的。对用户的管理是操作系统的重点。LINUX课程设计要用shell编程实现，将理论学习付诸实践。

编写一个shell程序，能够对系统的用户进行增加、删除、查找、显示特定用户信息等功能。

1. 系统需求分析
   1. menu.sh文件实现功能

显示系统所有功能的菜单界面，供用户从中选择多操作选项的菜单以便用户从中选择。用户通过键盘输入功能对应的数字标号来选择某一项功能。

1).在屏幕上显示当前所有用户的记录

2).在屏幕上显示当前所有用户（经过格式化和排序的）的记录

3).只在屏幕上显示用户名和用户ID

4).只在屏幕上显示（经过格式化和排序的）用户名和用户ID

5).查询并显示特定用户的记录

6).往passwd文件里增加新的用户记录

7).从passwd文件里删除某个用户记录

q).退出

当用户做了选择且所选的操作完成之后，可以再次显示主菜单以供用户做下一步选择。用while循环和case条件语句配合来完成此操作。

* + 1. 查询功能

通过menu菜单的1、2、3、4、5功能来实现查询功能。具体区别是：

显示所有用户信息：将passwd文件中的所有内容逐行分页输出，简单的以空格为每项内容之间的分割，没有格式化对齐每项内容。

显示所有用户信息（经格式化后）：将passwd文件中的所有内容逐行分页输出，以制表符为每项内容之间的分割，格式化对齐每项内容。

只显示用户名和UID：将passwd文件中的所有内容经过筛选后只留下用户名和用户UID，再逐行分页输出，简单的以空格为每项内容之间的分割，没有格式化对齐每项内容。

只显示用户名和UID（经格式化后）：将passwd文件中的所有内容经过筛选后只留下用户名和用户UID，再逐行分页输出，以制表符为每项内容之间的分割，格式化对齐每项内容。

通过关键字查找包含关键字的记录，并用特殊颜色强调关键字。

2.2 add.sh文件实现功能

当用户选择菜单页中的功能6时调用add.sh文件来完成添加新用户的功能。提示用户输入新用户的用户名、密码、UID、GID、注释、登录shell，并根据输入的用户名在/home下生成对应的用户工作目录/home/username。添加成功则显示添加成功信息，添加失败则显示添加失败的信息。最后询问是否继续添加用户（y/n）？根据用户的选择返回菜单或继续添加。

需要对用户输入进行校验：密码

2.3 delete.sh文件实现功能

当用户选择菜单页中的功能7时调用delete.sh文件来完成删除用户的功能。提示用户输入要删除的用户的唯一UID，若该用户存在显示其在passwd中的该条记录信息。提示用户确定删除（y/n）？根据用户的选择执行相应的操作。提示用户是否继续删除（y/n）?根据用户的选择返回菜单获继续删除用户。

* 1. 退出

当用户在键盘中输入q/Q时，提示用户确定退出（y/n）?根据用户的操作选择返回菜单或者退出系统。

1. 系统设计

3.1功能图

图 3-1系统功能图

显示当前所有用户的记录

显示当前所有用户经格式化后的记录

显示用户名和用户ID

显示格式化后用户名和用户ID

查找指定用户信息

添加用户信息

删除用户信息

用户信息主菜单

3.2流程图

* + 1. menu菜单

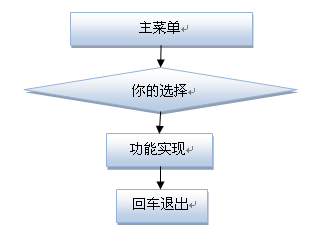


图 3-2 menu菜单

* + 1. add模块

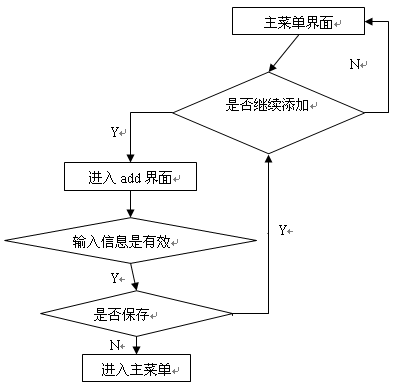


图 3-3 add模块

* + 1. delete模块

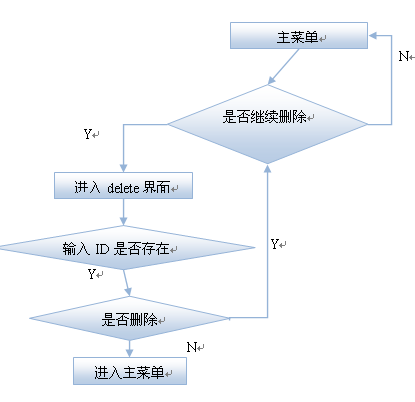
****

图 3-4 delete模块

1. 系统数据结构
   1. 文件结构

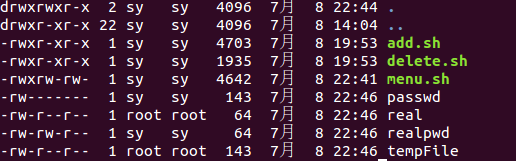


图 4-1 文件结构

* 1. 判断passwd文件是否存在的函数

CheckDataSourceFileExist()

{

if [ -f $dataSource ]

then

#文件存在

return 0;

else

#文件不存在

clear;

echo -n "警告 【$dataSource】 不存在！请确认！";

read ;

return 1;

fi

}

1. 系统实现
   1. menu文件

#! /bin/bash

dataSource='./passwd';

homePath='.';

addUsers="./add.sh";

deleteUsers="./delete.sh";

CheckDataSourceFileExist()

{

if [ -e $dataSource ]

then

#文件存在

return 0;

else

#文件不存在

clear; #清空标准输入

echo -n "警告 【$dataSource】 不存在！请确认！"; #-n代表不换行输出

read ; #读取选项存入特殊环境变量REPLY中而不是choice，继续while

return 1;

fi

}

choice="F";

while [ $choice != "Q" -a $choice != "q" ]

do

CheckDataSourceFileExist;

if [ $? -ne 1 ] #调用check函数返回是否为1，－ne 1即不等于1，即文件存在

then

clear;

echo " 用户信息管理主菜单";

echo "===========================================================";

echo " 1.显示当前所有记录";

echo " 2.格式化显示当前所有记录";

echo " 3.显示用户名和用户ID";

echo " 4.格式化显示用户名和ID";

echo " 5.查询特定用户信息";

echo " 6.添加新用户";

echo " 7.删除用户";

echo " 8.查看密码";

echo " Q.退出";

echo -n "你的选择:";

read choice;

#作用是清除由于read到string.Empty的时候引发的bash参数过多的错误。

if [ -z $choice ] #字符串长度为0是true

then

choice="empty";

fi

clear;

if [ $choice = "empty" ];

then

echo "选项尚未选择！";

else

case $choice in

1) CheckDataSourceFileExist;

if [ $? -ne 1 ] #不等于1，即文件存在

then

echo "当前的所有的用户信息如下所示:";

echo " 序号 用户名 密码 ID GID 说明 工作目录 登录Shell";

cat -n $dataSource | tr ":" " " | more;

#调用变量$,tr将：分隔符替换为空格，more命令是一个基于vi编辑器文本过滤器，它以全屏幕的方式按页显示文本文件的内容Enter（向下翻滚一行），空格（向下滚动一屏），Q（退出命令）该命令一次显示一屏文本，满屏后停下来，并且在屏幕的底部出现一个提示信息，给出至今己显示的该文件的百分比：--More（更多）--（XX%）

fi;;

2) CheckDataSourceFileExist;

if [ $? -ne 1 ]

then

echo "当前的所有的用户信息如下所示:";

echo -e " 序号 用户名\t密码\tID\tGID\t说明\t工作目录\t登录Shell"

cat -n $dataSource | awk -F ":" '{print $1 "\t" $2 "\t" $3 "\t" $4 "\t" $5 "\t" $6 "\t" $7 }' | more;

fi;;

#sort k 按照第一列将passwd文件排序，awk文本分析工具，把文件逐行（以\n）的读入,-F ":"将冒号作文本分割域输出第一个域,第二个(即用户名\t密码\tUID\tGID\t注释\t主目录\t登录shell)

3) CheckDataSourceFileExist;

if [ $? -ne 1 ]

then

echo "当前的所有的用户的用户名和用户ID如下所示:";

echo -e "　　　　序号 用户名,ID";

awk -F ":" '{print $1 "," $3}' $dataSource　| cat -n | more;

fi;;

4) CheckDataSourceFileExist;

if [ $? -ne 1 ]

then

echo "当前的所有的用户的用户名和用户ID如下所示:";

echo -e " 序号 用户名 \t ID";

awk -F ":" '{print $1 "\t " $3}' $dataSource | cat -n | more;

fi;;

5) CheckDataSourceFileExist;

if [ $? -ne 1 ]

then

keyWords="";

while [ -z $keyWords ] #字符串长度为0是true

do

echo -n "输入搜索关键词:";

read keyWords;

if [ -z $keyWords ]

then

echo "搜索关键词不能为空，请重新输入！";

fi

done

CheckDataSourceFileExist;

if [ $? -ne 1 ]

then

grep -i -n --color=auto $keyWords $dataSource; #-i不区分大小写,grep -n显示行号和cat不一样,关键字加颜色

if [ $? -eq 1 ] #找到grep命令返回0，否返回1，文件不存在返回2

then

echo "很遗憾，【$dataSource】文件中，并不存在与 $keyWords 一致的信息。";

fi

fi

fi;;

6) $addUsers;;

7) $deleteUsers;;

8)echo " 序号 用户名：密码";

cat -n "./realpwd" ;;

Q) printf "程序已经退出。";;

q) printf "程序已经退出。";;

\*) echo $choice "：此选项不是默认提供的功能。请确认。";; #default

esac

fi

echo -n "请按任意键返回菜单："; #-n不换行

read ; #存入特殊环境变量REPLY中而不是choice，继续while，所以输入什么都可以enter,q,都会继续。只有在case菜单中选择q是赋值给choice,case之后的选择都是这句，给了默认的

fi

done

* 1. add文件

clear;

isContinue="y";

userName="";

while [ $isContinue = "y" -o $isContinue = "Y" ]

do

#用户名处理，输入一致到字符串长度不为0为止，字母打头，数字加字母，判断是否已存在

userName="";

while [ -z $userName ] #长度为空true

do

echo -n "用户名:";

read userName;

if [ -z $userName ] #长度为空true

then

echo "错误！用户名是不能为空的，请重新输入！";

continue;

fi

if [ `expr match $userName "[a-zA-Z][0-9a-zA-Z]\*"` -ne `expr length $userName` ] #expr match返回匹配到的字符串的长度，exprlength求字符串长度。

then

echo "错误！用户名的只能由非数字打头的字符和数字组成，请重新输入！";

userName="";

continue;

fi

grep -q $userName "./passwd";

if [ $? -eq 0 ]

then

echo "错误！用户名已存在，请重新输入！";

userName="";

continue;

fi

done

#密码处理：1.验空，3.密码长度要6位，由大小写字母、数字和控制字符组成，2.验两次输入的一致性，

passWord="";

passWordAgain="";

while [ -z $passWord ]

do

echo -n "密码:"; #不换行输出

read passWord;

if [ -z $passWord ] #长度为0true

then

echo "错误！密码是不能为空的，请重新输入！";

continue;

fi

if [ `expr length $passWord` -ne 6 ]

then

echo "密码长度为6位，请重新输入!";

passWord="";

continue;

fi

if [ `expr match $passWord "[0-9a-zA-Z]\*"` -ne `expr length $passWord` ]

then

echo "密码由大小写字母、数字和控制字符组成，请重新输入!";

passWord="";

continue;

fi

echo -n "请在输入一次密码：";

read passWordAgain;

if [ $passWordAgain != $passWord ]

then

echo "两次输入的密码不一样，请重新输入！";

passWord="";

continue;

fi

done

#用户UID输入处理,UID为数字、一般非超级用户的ID是100-500

uID="";

while [ -z $uID ]

do

echo -n "用户UID:";

read uID;

if [ -z $uID ]

then

echo "错误！用户UID是不能为空的，请重新输入！";

continue;

fi

if [ `expr match $uID "[0-9]\*"` -ne `expr length $uID` ] #输入的密码在[0-9]正则表达式中比配的长度是否和输入的密码长度一致，校验是否都是0-9中数字

then

echo "错误！用户的UID必须为数字，请重新输入！";

uID="";

continue;

fi

if [ $uID -lt 100 -o $uID -gt 500 ] #-o或，小于100或大于500错

then

echo "错误！一般非超级用户的ID范围为100～500，请重新输入!";

uID="";

continue;

fi

grep -q $uID "./passwd"; #-q抑制输出

if [ $? -eq 0 ]

then

echo "错误！用户UID已存在，请重新输入！";

uID="";

continue;

fi

done

#用户组GID处理，是500-60000

gID="";

while [ -z $gID ]

do

echo -n "用户组GID:";

read gID;

if [ -z $gID ]

then

echo "错误！用户GID是不能为空的，请重新输入！";

continue;

fi

if [ `expr match $gID "[0-9]\*"` -ne `expr length $gID` ]

then

echo "错误！用户的GID必须为数字，请重新输入！";

gID="";

continue;

fi

if [ $gID -lt 500 -o $gID -gt 60000 ]

then

echo "错误！用户组的ID范围为500～60000，请重新输入!";

gID="";

continue;

fi

done

echo -n "注释:";read note;

#登录shell:bash,sh,csh,ksh

shellVersion="";

while [ -z $shellVersion ]

do

echo -n "请输入以下登录SHELL（bash,sh,csh,ksh）中一个:";

read shellVersion;

if [ $shellVersion != "bash" -a $shellVersion != "sh" -a $shellVersion != "csh" -a $shellVersion != "ksh" ] #-a与

then

echo "输入的Shell类型【$shellVersion】不在本系统支持范围内，请重新输入！";

shellVersion="";

continue;

fi

done

echo -n "用户工作目录:";

mkdir /home/$userName;

if [ $? -eq 0 ]

then

echo "/home/$userName成功创建！";

mdirec="/home/$userName";

fi

userInfo="$userName:x:$uID:$gID:$note:$mdirec:$shellVersion";

echo $userInfo >> $dataSource | sudo sh; #>>追加方式重定向

sudo sh -c "echo $userInfo >> ./passwd";

#因为sudo得到的权限并不随重定向符号输出，所以我们可以先将当前shell弄得有root的权限，然后再进行相关操作

echo "$userName:$passWord" >> "./realpwd"; #建立一个查看密码的文件

if [ $? -eq 0 ]

then

echo "$userName用户信息添加成功！";

else

echo "$userName用户信息添加失败！";

fi

echo -n "是否继续添加其他用户？？（y/n）";

read isContinue;

done

* 1. delete文件

clear;

dataSource='./passwd'; #为什么这个文件不认全局变量？

isContinue="y";

while [ $isContinue = "y" -o $isContinue = "Y" ] #-o逻辑或

do

echo -n "输入用户名:";

read userName;

if [ -z "$userName" ]

then

echo "用户名不能为空，请重新输入！";

isContinue="Y";

continue;

fi

grep -q $userName $dataSource;

if [ $? -eq 0 ] #找到返回0，没找到返回1

then

echo "用户信息为：";

grep $userName $dataSource;

echo -n "是否删除？（y/n）";

read isContinue;

else

echo "你输入的 【$userName】 用户不存在！";

echo -n "是否继续删除其他用户？？（y/n）";

read isContinue;

continue;

fi

if [ $isContinue = "y" -o $isContinue = "Y" ]

then

echo -n "请输入密码：";

read password;

fi

realpwd=`(grep $userName "./realpwd" | awk -F ":" '{print $2}')`;

echo $realpwd;

if [ "$realpwd" != "$password" ]

then

echo "密码输入错误";

break;

else

rowID1=$(grep -n $userName $dataSource | awk -F ":" '{print $1}'); #取得要删除的用户的记录行号

sed -e "$rowID1 d" $dataSource > tempFile; #删除某行:sed数字+d命令,将删除后的全部内容输出重定向到tempfile文件

cat tempFile > $dataSource; #再将tempfile文件写回（不包括删除了的用户）passwd文件

rmdir "/home/$userName";

rowID2=$(grep -n $userName "./realpwd" | awk -F ":" '{print $1}');

sed -e "$rowID2 d" "./realpwd" > real; #删除的密码文件中对应行

cat real > "./realpwd";

#rm tempFile;

fi

if [ $? -eq 0 ]

then

echo "记录删除！";

else

echo "系统错误，删除失败！";

fi

echo -n "是否继续删除其他用户？？（y/n）";

read isContinue;

if [ -z $isContinue ]

then

isContinue="N";

fi

done

1. 系统测试
   1. 功能测试

对实现程序的功能进行测试，测试程序是否实现目标功能。包括显示当前所有记录、格式化显示当前所有记录、显示用户名和用户ID、格式化显示用户名和ID、查询特定用户信息、添加新用户、删除用户、查看密码等。

* + 1. 显示当前记录和格式化显示

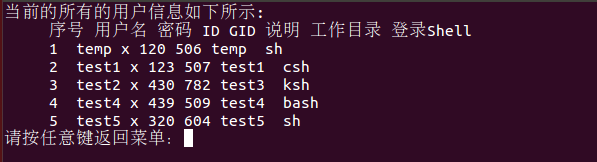


图6-1 显示当前所有用户信息

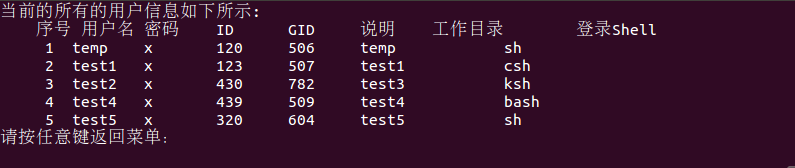


图6-2 格式化显示用户信息

6.1.2 显示用户名和用户ID及格式化显示

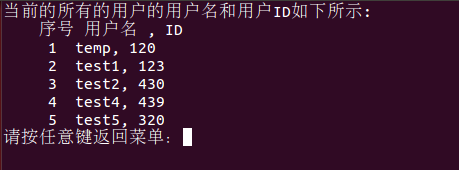


图6-3 显示用户名和用户ID

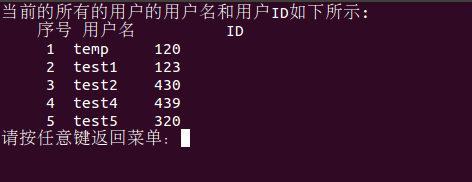


图6-4 格式化显示用户名和用户ID

6.1.3 查询特定用户信息

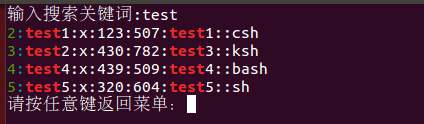


图6-5 查询特定用户信息

6.1.4 添加用户

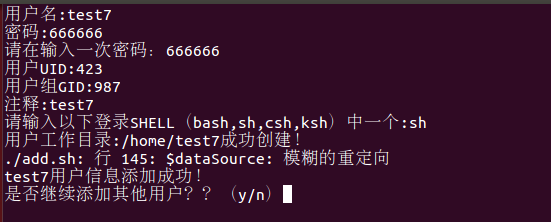


图6-6 添加新用户

6.1.5 删除用户

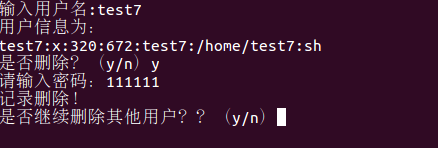


图6-7 删除用户

6.1.6 查询密码



图6-8 查询密码

* 1. 鲁棒性测试

6.2.1 输入错误



图6-9 菜单下输入错误

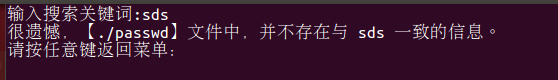


图6-10 搜索时输入错误

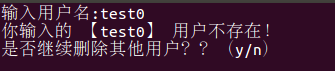


图6-11删除时输入错误

如图6-9,6-10和6-11所示，在输入错误的情况下，程序会提示输入错误，并进行下一步操作的提示。

6.2.2 添加用户测试

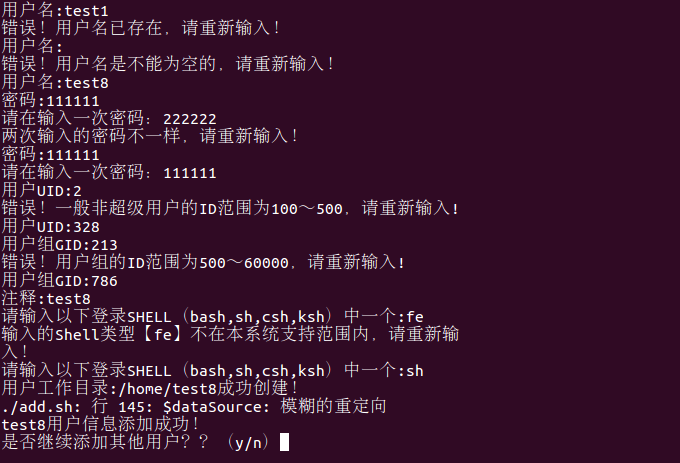


图 6-12 添加用户测试

图6-12 展示了添加用户时产生的不同情况，包括用户名的规定，UID的规定和GID的规定登录shell的规定，如果输入错误则提示用户正确的输入格式。

1. 总结

本次项目实现了shell编程的模拟Ubuntu系统的用户管理。在Ubuntu下用户的增删改查主要是通过编辑修改/etc/passwd文件来实现的。/etc/passwd文件有七个字段：分别是用户名：密码：UID：GID：注释：工作目录：登录shell。本项目通过新建的passwd文件来模拟/etc/passwd文件，实现用户的增删改查。

项目完成了查询并格式化输出所有用户的详细信息，查询并格式化输出用户名和UID，根据用户输入关键字来查询信息，添加用户即向passwd文件中写入用户信息，并在/home下建立工作目录。删除用户即删除用户在passwd文件中信息。除此外本项目还实现了对关键字不同颜色输出，添加用户时写入passwd文件的密码是加密的x。项目添加了一项功能，查看用户的真实密码。当删除用户时也会把真实密码一同删掉。

通过此项目我对Linux的操作系统有了更深一步的了解，同时从头学习了shell编程的语法，又掌握了一门新的脚本语言。还熟悉了grep，awk，cat，more，echo等命令，shell编程中流程控制语句while，if，case的用法，条件表达式，变量的使用加$，文件的输入输出重定向。

通过本项目用户可以更加直观的了解系统中自己或其他用户的情况，实现对用户操作的可视化方便用户管理。

不足之处，界面没有优化比如居中对齐，用户输入密码时没有对其进行隐藏。

**参考文献**

[1] 菜鸟教程linux教程之shell编程

http://www.runoob.com/linux/linux-tutorial.html

[2] 百度文库：shell编程--用户信息管理(模拟)

https://wenku.baidu.com/view/3e9ea06f48d7c1c709a14506.html

[3] 慕课网shell编程之变量

https://www.imooc.com/learn/336

[4] CSDN博客

https://blog.csdn.net/qing101hua/article/details/78982470