**Chapter 5**

1. Which of the following are valid variable names?

XxXxXx

\_

HOMEDIR

\_date

file\_name

file1

Slimit

1. Suppose that your HOME directory is /users/steve and that you have subdirectories as shown in the following
   1. Write the commands in terms the these variables to
      1. List the contents of the documents directory

$ ls $docs

* + 1. Copy all files from the letters directory to the proposals directory

$ cp $let/\* $prop

* + 1. Move all files whose names contain a capital letter from the letters directory to the current directory

$ mv $let/\*

* + 1. Count the number of files in memos

$ ls $docs/memos | wc –l

* 1. What would be the effect of following commands
     1. $ ls $let/..

Display the contents of $docs, i.e. /users/steve/documents

* + 1. $ cat $prop/sys.A >> $let/no.JSK

Append the contents from /users/steve/documents/proposals/sys.A to /users/steve/documents/letters/no.JSK

* + 1. $ echo $let/\*

display all the files with absolute paths in /user/steve/documents/letters

* + 1. $ cp let/no.JSK $progs

Copy the file /users/steve/documents/letters/no.JSK to /users/steve/programs

* + 1. $ cd $progs

Change the current directory to /users/steve/programs

1. Write a program called nf to display the number of files in your current directory. Type the program and test it out

vi nf

ls –la | grep ^-

1. Write a program called whos to display a sorted list of logged-in users. Just display the username and no other information. Type in program and test it out

vi whos

who | cut -c1-8 | sort

**Chapter 6**

1. What would be the output of following commands
   1. echo \*\*\* error \*\*\*

names test1 u vv zebra error names test1 u v zebra

* 1. echo ‘ls 5 \* 4 > 18 ?’

ls 5 \* 4 > 18 ?

* 1. echo $x

names test1 u vv zebra

* 1. echo What is your name?

What is your name?

* 1. echo $y

?

* 1. echo would you like to play a game?

Would you like to play a game?

* 1. echo “$y”

?

* 1. echo \\*\\*\\*

\*\*\*

* 1. echo $z | wc -l

1

* 1. echo \$$symbol

$>

* 1. echo “$z” | wc -l

3

* 1. echo $\$symbol

$$symbol

* 1. echo ‘$z’ | wc -l

1

* 1. echo “\”

> (promt)

* 1. echo \_$now\_

\_

* 1. echo “\\”

\

* 1. echo hello $symbol out

hello > out

* 1. hello \\

\

* 1. echo “\””

“

* 1. echo I don’t understand

> (prompt)

1. Write the commands to remove all the space characters stored in the shell variable text. Be sure to assign the result back to text. First use tr to do it and then do the same thing with sed.

Using tr

text=`echo $text | tr -d ‘ ‘`

Using sed

hi=`echo $text | sed 's/ //g'`

1. Write the commands to count the number of characters stored in the shell variable text. Then write the commands to count all the alphabetic characters. (Hint: Use sed and wc.) What happens to special character sequences such as \n if they're stored inside text?

echo “$text\n” | wc -c

echo “$text\n” | sed ‘s/[^A-Za-z]//g’ | wc -c

Special characters such as \n are interpreted by echo and produce a single character on shell.

1. Write the commands to assign the unique lines in the file names to the shell variable namelist.

namelist=`sort $names | uniq`

**Chapter 7**

1. Modify lu so that it ignores case when doing the lookup.

grep -i “$1” phonebook

1. What happens if you forget to supply an argument to the lu program? What happens if the argument is null (as in, lu "")?

In both cases, all the contents of phonebook will be displayed

1. grep “^$1” phonebook
2. Write a program called twice that takes a single integer argument and doubles its value

What happens if a noninteger value is typed? What if the argument is omitted?

$ vi twice

echo `expr $1 "\*" 2`

chmod 744 twice

if non-integer value is typed we get error “expr: non numeric argument”

if no argument supplied, we get syntax error

1. Write a program called home that takes the name of a user as its single argument and prints that user's home directory

$ vi home

grep "^$1$" /etc/passwd | cut -d: -f6

$ chmod 744 home

1. Write a program called suffix that renames a file by adding the characters given as the second argument to the end of the name of the file given as the first argument

$ vi suffix

mv $1 $12$

$ chmod 744 suffix

1. Write a program called unsuffix that removes the characters given as the second argument from the end of the name of the file given as the first argument

$ vi unsuffix

newname=$(echo $1 | sed "s/$2//g")

mv $1 $newname

$ chmod 744 unsuffix