OS Assignment 1

Akshat Gupta
2021515
CSAI, BTech 2021-25

Internal Commands:

1. **echo** - By default, print the given argument followed by a newline characte (“\n”)
   echo -n
   Omits the newline character at the end of the arguments to be printed
   echo -E
   Disregards all escape characters in the string passed as argument to the echo function
   Test cases:
   - echo “hello world” → hello world\n
   - echo -n “hello world” → hello world
   - echo -E “hello \cworld” → hello \cworld\n

   Error Handling
   If no argument is passed to echo function, it prints only newline character.
   If any option other than -n or -E is passed, shell prints an error message.
2. **pwd** - By default, prints the path of the current working directory
   pwd -L
   Prints the logical path of the current working directory, if symbolic link is created between the working directory and an external directory
   pwd -P
   Prints the resolved path of the current working directory (resolved implies all symbolic links are resolved)
   Test Cases
   - pwd → /Users/akshatgupta/Desktop
   - pwd -L → /Users/akshatgupta/Desktop
   - pwd -P → /Users/akshatgupta/Desktop
   - pwd /Users/akshatgupta/ → error

   Error Handling
   If pwd is passed with any argument other than the specified options, shell prints an error message for the same.
3. **cd** - Takes an argument which specifies the absolute/relative path of the directory to change to
   cd ~
   Calls cd on the home directory i.e. switches to the home directory

cd -
Same as calling cd .. i.e. switches to the parent directory
Test Cases
- cd /Users/akshatgupta/Desktop/IIITD → switches to IIITD directory on Desktop
- cd ~ → Switches to /Users/akshatgupta (Home dir)
- cd - → Switches to /Users/akshatgupta/Desktop (Parent dir)

Error Handling
If cd is passed without an argument, shell prints an error message.
If cd is passed with any option as argument other than the above specified the shell prints an error message for the same; if cd is passed with an argument that is the directory name that does not exist at the specified path, shell attempts to cd into the directory, and if failed, prints an error message for the same.

External Commands (using Fork):

1. **ls** - Lists all non-hidden files and directories in the current working directory or the specified directory (passed as an argument to the ls command)

   ls -a
   Prints all hidden files and directories along with the default printed ones
   ls -r
   Prints all non-hidden files and directories in the reverse order to the default-executed command
   Test Cases
   - ls → (prints all files and directories in current working directory)
   - ls -a → (prints all hidden files as well such as "." & "..")
   - ls -r → (prints the same output as "ls", but in reverse order)
   - ls /Users/akshatgupta/Desktop → (prints all non-hidden files and directories in the Desktop directory)
   - ls -C /Users/akshatgupta/Desktop → error (since -C not supported)

   Error Handling
   If ls is passed with an option other than the ones specified above, shell prints an error message for the same. If ls is passed with more than one directory as argument, prints an error message (since shell only supports listing for one directory at a time).
2. **cat** - Prints the contents of a file passed as argument to the cat function

   cat -n
   Numbers all lines being printed (including blank lines) starting at 1
   cat -b

Numbers all non-blank lines starting at 1

Test Cases

- cat /Users/akshatgupta/Desktop/test.txt → (prints contents of test.txt e.g. hello\n\nworld)
- cat -n /Users/akshatgupta/Desktop/test.txt → 1 hello\n2\n3 world
- cat -b /Users/akshatgupta/Desktop/test.txt → 1 hello\n\n2 world

Error Handling

If file passed as argument to cat does not exist, shell throws an error. If any option other than the ones specified above are passed to cat, shell prints an error message for the same.

3. **date** - Prints the current local date and time

date -n

-n is an obsolete flag i.e. it is ignored which means the output format is the same as the default format

date -R

-R is a formatter flag i.e. it prints the date and time in a certain format specifically the RFC 2822 date and time format (“%a, %d %b %Y %T %z”)

Test Cases

- date → Wed Oct 26 15:30:19 IST 2022
- date -n → Wed Oct 26 15:30:19 IST 2022
- date -R → Wed, 26 Oct 2022 15:30:19 +0530
- date -j → error (since this option is not supported by shell)

Error Handling

If any argument other than the options specified above is passed with the date command, the shell prints an error message for the same.

4. **rm** - Remove the file/directory passed as argument to the rm command

rm -i

Asks for confirmation before deleting the specified file (passed as argument)

rm -v

“Verbose” i.e. lists the names of the files as they are deleted/removed

Test Cases

- rm /Users/akshatgupta/Desktop/test.txt → (removes test.txt, if it exists)
- rm -i /Users/akshatgupta/Desktop/test.txt → (asks for confirmation of deletion, if given yes, removes test.txt if it exists)
- rm -v /Users/akshatgupta/Desktop/test.txt → (removes test.txt if it exists, and prints “test.txt” to stdout while removing
- rm -f /Users/akshatgupta/Desktop/test.txt → error (since -f is not supported by shell)

Error Handling
If rm is passed without an argument, shell prints an error message. If any option other than the ones specified above are passed with the rm argument, the shell prints an error message. If file does not exist, shell prints an error message saying file does not exist.

5. **mkdir** - Create new directory in the current working directory (if argument specified is relative path), or at the specified path if absolute path is passed as argument

    mkdir -v
    “Verbose” i.e. lists the names of the directories as they are created
    mkdir -p
    Takes the path of the new directory as argument, and creates any intermediate parent directories, if needed, as well
    Test Cases
    - mkdir /Users/akshatgupta/Desktop/newdir → (creates newdir in the Desktop directory)
    - mkdir -v /Users/akshatgupta/Desktop/newdir → (creates newdir in the Desktop directory and prints “newdir” to stdout)
    - mkdir -p /Users/akshatgupta/Desktop/new/dir/newdir → (creates “new” at Desktop directory, “dir” in “new” directory, and “newdir” in “dir” directory recursively)
    - mkdir -m newdir → error (since -m is not supported by shell)

    Error Handling
    If options other than the ones specified above are passed to mkdir, shell throws an error. If directory name passed as argument is already present at the specified path, shell prints an error message for the same. If no argument is passed to mkdir, shell prints an error message for the same.

External Commands (using Threads):

- If commands are followed by “&t” (space-separated), the process is called through thread creation, while otherwise the process is called through fork(), exec() and wait() system calls.
- The shell checks for the command being followed by “&t”. If affirmative, then the shell calls the thread_execute(char **args) function which in turn does the process of thread creation and subsequently calls the system() function through the arbitrarily defined function sys_call() to execute the specified command.
- pthread_create() → sys_call() → system() → pthread_exit() → pthread_join()

Test Cases

```
sh456 >>> echo "hello"
"hello"
sh456 >>> echo -n "hello"
"hello" sh456 >>> echo -E "hello \c world"
-E "hello \c world"
sh456 >>> cd ..
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS
sh456 >>> cd os_assignments
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments
sh456 >>> cd ~
sh456 >>> pwd
/Users/akshatgupta
sh456 >>> cd Desktop/IIITD/SEM_3/OS/os_assignments
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments
sh456 >>> pwd -L
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments
sh456 >>> pwd -P
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments
sh456 >>> date
Wed Oct 26 17:08:58 IST 2022
sh456 >>> date -n
Wed Oct 26 17:09:00 IST 2022
sh456 >>> date -R
Wed, 26 Oct 2022 17:09:02 +0530
sh456 >>> cat makefile
sh456:
    gcc sh456.c -o sh456
    gcc ls.c -o ls
    gcc cat.c -o cat
    gcc date.c -o date
    gcc rm.c -o rm
    gcc mkdir.c -o mkdir


sh456 >>> cat -n makefile
1  sh456:
2      gcc sh456.c -o sh456
3      gcc ls.c -o ls
4      gcc cat.c -o cat
5      gcc date.c -o date
6      gcc rm.c -o rm
7      gcc mkdir.c -o mkdir
8
9
```

```
sh456 >>> cat -b makefile
1  sh456:
2       gcc sh456.c -o sh456
3       gcc ls.c -o ls
4       gcc cat.c -o cat
5       gcc date.c -o date
6       gcc rm.c -o rm
7       gcc mkdir.c -o mkdir


sh456 >>> mkdir -v newdir
newdir
sh456 >>> ls
cat  ls.c  makefile  date  rm.c  cat.c  newdir  mkdir.c  OS_.pdf  mkdir  date.c  sh456.c  ls  rm  sh456
sh456 >>> cd newdir
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments/newdir
sh456 >>> ls

sh456 >>> ls -a
.  ..
sh456 >>> mkdir -p n1/n2/n3
sh456 >>> ls -a
.  ..  n1
sh456 >>> cd n1/n2
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments/newdir/n1/n2
sh456 >>> cd ../..
sh456 >>> pwd
/Users/akshatgupta/Desktop/IIITD/SEM_3/OS/os_assignments/newdir
sh456 >>> cd ..
sh456 >>> rm -i newdir
Do you wish to delete newdir? (y/n) y
no such file or directory: Directory not empty
sh456 >>> cd ./newdir/n1/n2
sh456 >>> rm n3
sh456 >>> cd ..
sh456 >>> rm n2
sh456 >>> cd ..
sh456 >>> rm n1
sh456 >>> ls

sh456 >>> cd ..
sh456 >>> rm newdir
sh456 >>> ls
cat  ls.c  makefile  date  rm.c  cat.c  mkdir.c  OS_.pdf  mkdir  date.c  sh456.c  ls  rm  sh456
sh456 >>> exit
```