Dhairya Chaudhary
2019035

Operating Systems - Assignment 1

TASK 2

THE SYSTEM

The shell runs an infinite loop that has three functions:

1. Takes a command from the user
2. Executes it
3. Frees memory

This loop is exited once the exit function is called (the loop has a variable called status which is initially 1 and set to zero when we call exit). The user can enter a new command for the shell each time it displays the shell prompt (“uwu’s shell>>>” in this case). Input is read using the built-in getline function. After being read the command is added to history. Arguments are then separated (using the space character) and stored in an array of strings (character pointers).

This array is sent to the execute function that can call all the commands I have implemented. This function always returns a value!=0 unless we call exit. Memory is freed using free(). There is one helper function present in the code.

Internal commands implemented: cd, echo, history, pwd, exit
External commands implemented: ls, cat, date, rm, mkdir (these are handled using fork(), execl() and wait(). We have specified the path to implement these.)

SHELL COMMANDS

Internal-

- **cd**

  OPTIONS
  > “-L”
  Signals to follow symbolic links. This is the default state of cd.
  > ”~”
  Takes user to home directory

ERRORS/ATTACKS TACKLED
> If more than one directory is specified
In this case, the directory is not changed and an appropriate error message is printed to stderr.
> If the directory mentioned does not exist
In this case, again, the directory is not changed and an appropriate error message is printed to stderr.

ASSUMPTIONS
> If no directory is mentioned with, directory is switched to the home directory.

- **history**

OPTIONS
> "-c"
Clears history.
> "-a"
Adds the history from the current running of the shell to the history file (implemented as a text file history.txt).

ERRORS/ATTACKS TACKLED
> If input provided is invalid
An appropriate error message is printed to stderr.
> If history file is not present or cannot be opened due to some error
This is for the -a function. An appropriate error message is printed to stderr and further execution of the command doesn't take place.

ASSUMPTIONS
> Size of history is taken to be 500.
> History is updated when the user enters a command.
> History file is implemented as a text file history.txt (present in folder).

- **echo**

OPTIONS
> "-n"
No newline added to the end.
> "*"
Prints contents of the current directory (similar to ls).

ERRORS/ATTACKS TACKLED
> If input provided is invalid
An appropriate error message is printed to stderr.
> In the implementation of *, we close the directory and take precautions to prevent errors or leaks.

ASSUMPTIONS
> If no argument follows echo, a newline is printed

- **pwd**

  OPTIONS
  > "-P"
  Displays physical directory without symlinks (This is assumed by Linux).
  > "-L"
  Does not resolve symlinks (This is assumed by Linux).

  ERRORS/ATTACKS TACKLED
  > If input provided is invalid
  An appropriate error message is printed to stderr.
  > Different lengths of command are considered.

  ASSUMPTIONS
  > The general configuration (for -L and -P are followed)

- **exit**

  OPTIONS
  N/A

  ERRORS/ATTACKS TACKLED
  N/A

  ASSUMPTIONS
  > Previous memory can be safely freed after exiting

Dhairya Chaudhary
2019035

External-

- **ls**

  OPTIONS
  > “-1”
  Displays each member of the directory in a separate line.
  > ”-a”
  Also prints the files and directories whose names start with a ‘.’.

  ERRORS/ATTACKS TACKLED
  > If the input is a file and not a directory
  An appropriate error message is printed to stderr and further operations aren’t performed.
  > If directory cannot be opened
  An appropriate error message is printed to stderr and further operations aren’t performed.
  > If correct input format is not followed
  An appropriate error message is printed to stderr and further operations aren’t performed.
  > Data leaks are prevented at every stage.

  ASSUMPTIONS
  > ls can handle multiple directories as input
  > ls can handle multiple flags

- **cat**

  OPTIONS
  > “-n”
  Numbers all file lines while printing
  > ”-E”
  Displays line ends (denoted with a hash).

  ERRORS/ATTACKS TACKLED
  > If file is not found
  An appropriate error message is printed to stderr and further operations aren’t performed.
  > If insufficient operands are provided

An appropriate error message is printed to stderr and further operations aren't performed.
> If correct input format is not followed
An appropriate error message is printed to stderr and further operations aren't performed.
> All data leaks are prevented and precautions are taken while opening and closing files.

ASSUMPTIONS
> cat can handle multiple directories as input
> cat can handle multiple flags
> File data is presented as it is, no new lines, spaces etc. modify the actual file content.

- **date**

OPTIONS
> "-u"/"--utc"/"--universal"
Prints UTC time.
> "--help"
Prints information regarding functionality.

ERRORS/ATTACKS TACKLED
> If correct input format is not followed
An appropriate error message is printed to stderr and further operations aren't performed.
> Best practices are followed and leaks are prevented using these.

ASSUMPTIONS
N/A

- **rm**

OPTIONS
> "-i"
Provides an interactive interface while deleting files.
> "-v"
Prints each deleted file after it is deleted.

ERRORS/ATTACKS TACKLED
> If directory is picked instead of file
An appropriate error message is printed to stderr and further operations aren't performed.
> If insufficient operands are provided
An appropriate error message is printed to stderr and further operations aren't performed.
> If correct input format is not followed
An appropriate error message is printed to stderr and further operations aren't performed.
> All data leaks are prevented and precautions are taken while opening and closing files.

ASSUMPTIONS
> rm can handle multiple directories as input
> rm can handle multiple flags

- **mkdir**
  OPTIONS
  > "-v"
  Prints a message for each directory created.
  > "-P"
  No error if present, creates parent as needed.

  ERRORS/ATTACKS TACKLED
  > If directory already exists
  An appropriate error message is printed to stderr and further operations aren't performed.
  > If insufficient operands are provided
  An appropriate error message is printed to stderr and further operations aren't performed.
  > If correct input format is not followed
  An appropriate error message is printed to stderr and further operations aren't performed.
  > All data leaks are prevented and precautions are taken while opening and closing files.

  ASSUMPTIONS
  > mkdir can handle multiple directories as input

Dhairya Chaudhary
2019035

> mkdir can handle multiple flags

TEST CASES

All commands below can be run on the shell after calling make and having it running-

NOTE: The files and directories mentioned in the test case (newdir, anotherdir, onemoredir, somefile.txt, someotherfile.txt, textfile.txt) are simply for example/demonstration purposes and are not present in the submitted folder.

```
>>> pwd
>>> ls -1 -a
>>> date -u
>>> cd ..
>>> mkdir newdir anotherdir onemoredir -v
>>> ls -a -1
>>> cd newdir
>>> history -a
>>> echo hello
>>> echo -n hello
>>> history
>>> pwd -P
>>> ls -a
>>> cd ..
>>> rm somefile.txt someotherfile.txt -i -v
>>> cat textfile.txt -E -n
>>> history -c
>>> exit
```