SHELL IN C

Description:---

This straightforward shell can execute the three internal commands "cd," "echo," and "pwd,". The shell would deal with these commands directly. Additionally, this shell is able to process the following five external commands: ls, cat, date, rm, and mkdir.
Each external command has a different program, while internal commands are written inside the main class. There are one or two options available for each external command.

Some critical information/pointers and assumptions-

- '~' expands my home folder. For example, /home/hitesh in my case.
- The size of the paths to a file cannot exceed more than 100 characters in my shell.
- Commands like rm and mkdir can take multiple file names as arguments.
- .. (dot dot) represents the parent directory, and .(dot) represents the current directory.
- User can only choose one option at a time.

_____COMMANDS_____

1)Internal Commands->

- CD-> Goes into the directory which we provided.
  CD ~= goes into the home directory
  CD ..=help int the previous directory

- ECHO->echo hello It will print the rest of the string as it is echo It will print a new line echo -n It will not end it with a new line
  Echo -n= do not output the trailing newline
  Echo *= ls lists all the files in the current directory.

- PWD-> Show the current directory
  Pwd -L=use PWD from environment, even if it contains symlinks
  Pwd -P=avoid all symlinks

2)External Commands->

- Cat ->it will print the file on the shell window
  cat -n: number all output lines
  cat -E: display $ at the end of each line

- MKDIR- mkdir filename this will create a new directory and give an error if not possible.
  mkdir -v filename this will create a new file directory and give a messege on completion.

  mkdir -m mode filename this will create a file directory with the particular mode.

- DATE -> date Display the current time in the given FORMAT, or set the system date.
  date -I output date/time in ISO 8601 format.

  Example:2006-08-14T02:34:56-0600

date -R output date/time in ISO 8601 format

Example:2006-08-14T02:34:56-0600

- MR -> rm filename this will delete the file with that name if it exists else gives an error.
  rm -i filename this will promt a question before deleting the file.

  rm -v filename this will delete the file the give a missege if it is deleted or not.

- LS-> ls lists all the files in the current directory
  ls -a: do not ignore entries starting with . also it ignores .(dot) and ..(dot dot) files
  In the current directory.
  Ls ~ : show all the files present in the home directory

System Calls Used:

1)Fork(): Used for creating a new process, known as the child process and it concurrently runs with the parent process.
2)readdir(): This function returns a pointer to a different struct variable pointing to the next directory
3)chdir(): It changes the current directory of the calling process to the directory specified in the path.
4) execl(): Like all exec functions, execl **replaces the calling process image with a new process image**. This has the effect of running a new program with the process ID of the calling process. Note that a new process is not started; the new process image overlaps the original process image.

5)opendir(): This function opens a directory corresponding to the directory name given and returns a pointer to that directory
6)wait(): It blocks the calling process until one of the child processes exits. After child process
exits or ends, the parent continues its execution after the wait call instruction.
7)remove(): used to delete a file or directory.
8)mkdir(): creates a new directory of the name passed in it as an argument.
9)getcwd(): used to get the current working directory.

The thread-based execution will be performed if the command is followed by the characters “&t.” The rest of the functionalities should remain the same. Also we used the set of external command programs which could be used with either versions of the shell, be it the one that uses fork()/execl() or the one that uses pthread create()/system().

Also, pthread_create() and pthread_join() have been implemented in the function. Which further uses system() in stdlib.h to work as a terminal.