Käyttöjärjestelmät ja systeemiohjelmointi

Lukas Honka

Project repository root link: https://github.com/LStackH/LUTCourse-SystemProgramming

All projects can be found from the root repository, each project has additional documentation in their respective README.md

Project 1: **Warmup to C and Unix programming**

This project implements a simple program, `reverse`, that reverses input lines. It supports three modes of operation:

1. Terminal Input to Terminal Output.

2. File Input to Terminal Output.

3. File Input to File Output.

The solution dynamically manages memory using a linked list and validates user input to prevent invalid operations. Input is added line by line to the linked list, and finally printed from the beginning of the list to the end, which ends up printing the original text in reverse order.

Additional Documentation: More details available in the repository README.

Usage examples:

Using ./reverse and exiting with CTRL+D:

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ ./reverse
Hello
World!
World!
Hello
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$
```

Using ./reverse input.txt, printing to terminal:

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ cat input.txt
Hello
World!
From a file!
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ ./reverse input.txt
From a file!
World!
Hello
```

Using ./reverse input.txt output.txt, printing to output.txt:

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ cat output.txt
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ ./reverse input.txt output.txt
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$ cat output.txt
From a file!
World!
Hello
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 1$
```

Error cases:

Cannot open file:

Too many arguments:

Project 1 repository: https://github.com/LStackH/LUTCourse-SystemProgramming/tree/main/Project%201

Project 2: **Implementing Unix Utilities**

This project implements four simple Unix-like utilities: my-cat, my-grep, my-zip, and my-unzip.

1. **my-cat**: Reads and outputs the content of one or more files to the terminal.
2. **my-grep**: Searches for lines containing a specified term in one or more files and outputs the matching lines.
3. **my-zip**: Compresses the contents of files using a basic Run-Length Encoding (RLE) algorithm.
4. **my-unzip**: Decompresses files that were compressed using my-zip.

The programs handle edge cases such as invalid arguments, missing files, and unsupported operations, printing appropriate error messages to stderr.

Additional Documentation: More details available in the repository README.

Usage example:

Using ./my-cat <file>, prints the contents of the file to the terminal

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 2$ ./my-cat tests/bar.txt
this line has foo in it

so does this foolish line; do you see where?

even this line, which has barfood in it, will be printed.

FOO BAR FOO BAR ho hoho or

orfoorfoorf totoo
```

Using ./my-grep <searchterm> <file>, outputs all lines in <file> containing <searchterm>

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 2$ ./my-grep line tests/bar.txt
this line has foo in it
so does this foolish line; do you see where?
even this line, which has barfood in it, will be printed.
```

Using ./my-zip file.txt > compressed_file.txt, compress contents of file.txt to compressed_file.txt. Using ./my-unzip compressed_file will uncompress and print the contents to terminal:

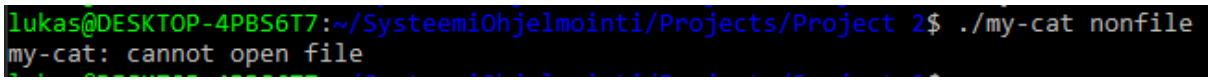
```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 2$ ./my-zip compress/file.txt > compress/compressed_file.txt
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 2$ cat compress/compressed_file.txt

a□b□
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 2$ ./my-unzip compress/compressed_file.txt
aaaaaaaaaabbbb
```

Error cases:

my-cat:

- Opening non-existent file



my-grep:

- Incorrect usage

my-zip:

- Can’t open input file

my-unzip:

- Can’t open file:

Project 2 repository: https://github.com/LStackH/LUTCourse-SystemProgramming/tree/main/Project%202

Project 3: **Implementing a Unix Shell**

This project implements a basic Unix shell called wish (short for Wisconsin Shell). The shell processes user input, executes commands, and supports common shell features like built-in commands, redirection, and parallel command execution.

The shell operates in two modes:

1. **Interactive Mode**: Accepts commands directly from the user.
2. **Batch Mode**: Reads commands from a file and executes them sequentially.

Key Features:

1. **Built-in Commands**:
    - exit: Terminates the shell.
    - cd: Changes the current working directory.
    - path: Modifies the search path for executable files.
2. **Command Execution**:
    - Executes external programs using the specified search path.
3. **Redirection**:
    - Redirects the output of commands to a specified file using the > symbol.
4. **Parallel Execution**:
    - Executes multiple commands simultaneously, separated by &.
5. **Different types of error messages:**
    - Easy use of defining new error messages and printing them to stderr at appropriate places.

Additional Documentation: More details available in the repository README.

Usage Examples:

Interactive Mode:

Entering Interactive Mode, with ./wish

Using non-built-in commands, while wish is running, with ./ls and /cat <file>

```
wish> ls
README.md  batch_test.txt  input.txt  ls.txt  output.txt  output2.txt  testfolder  wish  wish.c
wish> cat ls.txt
README.md
batch_test.txt
input.txt
ls.txt
output.txt
output2.txt
testfolder
wish
wish.c
wish>
```

Using built-in commands, clearing path and then exiting with built-in command “exit.

```
wish> cd testfolder
wish> ls
inside
wish> path
wish> ls
Error: command not found
wish> exit
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 3V2$
```

Using redirection with > to redirect output, pwd > pwdoutput.txt:

```
wish> pwd > pwdoutput.txt
wish> cat pwdoutput.txt
/home/lukas/SysteemiOhjelmointi/Projects/Project 3V2
wish>
```

Using parallel execution with &, waits until all commands have been executed before resuming shell, in this case, ends up waiting for the “sleep 5” to finish, while doing the two other commands in parallel:

```
wish> sleep 5 & cat ls.txt & echo "doing in parallel"
README.md
batch_test.txt
input.txt
ls.txt
output.txt
output2.txt
testfolder
wish
wish.c
"doing in parallel"
wish>
```

Batch mode:

Entering batch mode requires ./wish <file>. Each line in the file will either be recognized as a command or an error message will be printed. Here, it will also wait for the “sleep 5”, before continuing execution.

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 3V2$ cat batch_test.txt
not a command
ls
pwd
pwd > output.txt
echo "here" & sleep 5
cat output.txt > input.txt
cat input.txt
exit
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 3V2$ ./wish batch_test.txt
Error: command not found
README.md  batch_test.txt  input.txt  ls.txt  output.txt  output2.txt  pwdoutput.txt  testfolder  wish  wish.c
/home/lukas/SysteemiOhjelmointi/Projects/Project 3V2
"here"
/home/lukas/SysteemiOhjelmointi/Projects/Project 3V2
```

Error cases:

Entering ./wish with too many arguments:

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 3V2$ ./wish arg1 arg2
Error: Too many arguments, usage wish <file>
```

Fail to open the batch file:

```
lukas@DESKTOP-4PBS6T7:~/SysteemiOhjelmointi/Projects/Project 3V2$ ./wish batchfile
Error: Couldn't open file
```

Clearing path, can’t access non-built-in commands:

```
wish> path
wish> ls
Error: command not found
wish> path /bin
wish> ls
README.md  batch_test.txt  input.txt  ls.txt  output.txt  output2.txt  pwdoutput.txt  testfolder  wish  wish.c
wish>
```

Using redirection wrong:

```
wish> ls > > lsoutput.txt
Error: invalid redirection syntax
wish>
```

Extra arguments to cd:

```
wish> cd there here
Error: cd requires exactly one argument
wish>
```

Extra arguments to exit:

```
wish> exit 1
Error: exit takes no arguments
wish>
```

Project repository: https://github.com/LStackH/LUTCourse-SystemProgramming/tree/main/Project%203V2