COP6611: Operating Systems **Name:** *Dr. Ankur Mali*
Assignment 1: Implement uniq and head command in user and kernel mode xv6
09/01/2023

Academic Honesty

All work submitted in this course must be your own. Cheating and plagiarism will not be tolerated. If you have any questions about a specific case, please ask me and your TA. We will be checking for this!

Assignment Notes

General Notes

- Please read the assignment instructions carefully, including what files to include/submit.
- Don't assume any limitations unless they are explicitly stated.
- Developing optimized solutions is good, but I won't deduct any points if your code is not optimized
- Please ensure the functionalities requested in these assignments are included; you can add extra but cannot remove any.
- When in doubt regarding what needs to be done, ask. Another plausible option is to test it in the real UNIX or Ubuntu operating system. Unix is your friend; use it wisely. Does your code behave the same way? If not, debug. Remember to use man "command_name" for reference.
- Test your solutions and make sure they work. Attach your screenshots showing your steps, your console, and other details.

Rubric

Here are some of the basic things our script will test, but these are not the only things we will test. Don't hard code any solution. Ensure you test your program completely and provide detailed information concerning steps followed/adopted to ensure your code is bug-free. This assignment will add 6.00 to your total grade point. To make it easy, I have allocated 600 points, which will be converted into 6.00 after your submission.

Total: 600 points = 6.00 Grade point

1 Implementing uniq on xv6 as user program and also as system call - 300 points

The uniq command in Linux is a command-line utility that reports or filters out the repeated lines in a file. In simple words, uniq is the tool that helps to detect the adjacent duplicate lines and also deletes the duplicate lines. Uniq filters out the adjacent matching lines from the input file(that is required as an argument) and writes the filtered data to the output file. For instance, look at the simple file OS611 example.txt. The cat command will print out all the contents from the file, and uniq removes duplicates from the adjacent line. For example, consider the file "example.txt" consisting of the following text. **Example** $cat OS611 example.txt

- I understand the Operating system.
- I understand the Operating system.
- I understand the Operating system.
- I love to work on OS.
- I love to work on OS.
- Thanks xv6.

Now, when I run the uniq command on the example.txt I get the following $uniq OS611 example.txt

- I understand the Operating system.
- I love to work on OS.
- Thanks xv6.

Task-1 - User space Similar to HW0 (hello.c), create uniq.c that performs the above task by replicating the functionality of uniq. This program should display/print the following message " Uniq command is getting executed in user mode, " followed by content. In the above case, you will get the following output $uniq OS611 example.txt

"Uniq command is getting executed in user mode."

- I understand the Operating system.
- I love to work on OS.
- Thanks xv6.

Task-2 - Kernel space Similar to HW0, create a system call named uniq. This should display/print the following message " Uniq command is getting executed in kernel mode", then followed by content.

1.1 Rubric for Part-1

We assume all commands are working in user and kernel mode.

- Program is working in user mode but not kernel mode. (-150)
- If uniq does not compile in any mode (user or kernel mode). (-200)
  - If nothing works, then based on logic, some points will be provided (range 0 -100).
- If any debug statements (printf) are getting printed alongside output. (-15)
- If the program does not terminate successfully (-50)
- cat "filename" — uniq does not work (-25)
- uniq -c "filename" does not work (-15)
- uniq -i "filename" does not work (-25)
- uniq -d "filename" does not work (-15)

2 Implementing head on xv6 as user program and also as system call - 300 points

It is complementary to the Tail command. The head command, as the name implies, prints the top N number of data of the given input. By default, it prints the first 14 lines of the specified files. If more than one file name is provided, then data from each file is preceded by its file name. If no filename is provided, the head should read from standard input. You should also be able to invoke it without a file and have it read from standard input. (Important:If the first argument does not start with -, the number of lines to be printed should default to 14)

Like uniq, you will also implement head command in user and kernel mode.

Task-1 - User space Similar to HW0 (hello.c), create head.c that performs the above task by replicating the functionality of the head. This program should display/print the following message: " Head command is getting executed in user mode, " followed by content.

Task-2 - Kernel space Similar to HW0, create a system call named head. This should display/print the following message: "Head command is getting executed in kernel mode", then followed by content.

2.1 Rubric for Part-2

- Program is working in user mode but not kernel mode. (-150)
- If the head program does not compile in any mode (user or kernel mode). (-200)
  - If nothing works, then based on logic, some points will be provided (range 0 -100).
- If any debug statements (printf) are getting printed alongside output. (-15)
- If the program does not terminate successfully (-50)
- head "filename" does not work. Assuming it works but doesn't respect the default settings = 14 lines. Then you will lose (-50 points) (-50)
- head -n "some number" "filename" does not work (-45)

Submission Instructions

The file should be submitted in zip format, with 2 folders (uniq and head)

Each folder should consist of the following documents:

- Screenshots showing successful execution of the command "make clean."
- Screenshot showing successful execution of the command "make qemu" or "make qemu-nox".
- Output of all conditions provided in your rubric.
- A detailed Readme file that explains your logic (code section), and steps followed to run your code (attach a screenshot and then explain steps). If README files are missing, then we will deduct 25 points for each command (25 for uniq and 25 points for head).
- All c scripts and modified Makefile.