**LINUX:DAY1**

whoami # my login

hostname # name of the computer

echo "hello world" #print characters in the screen

echo $home #print environment variable

echo my login is $(whoami) #replace $(xx) with program o/p

date #current date and time

cal #month of a calender

shazone # bad command

A screenshot of a computer

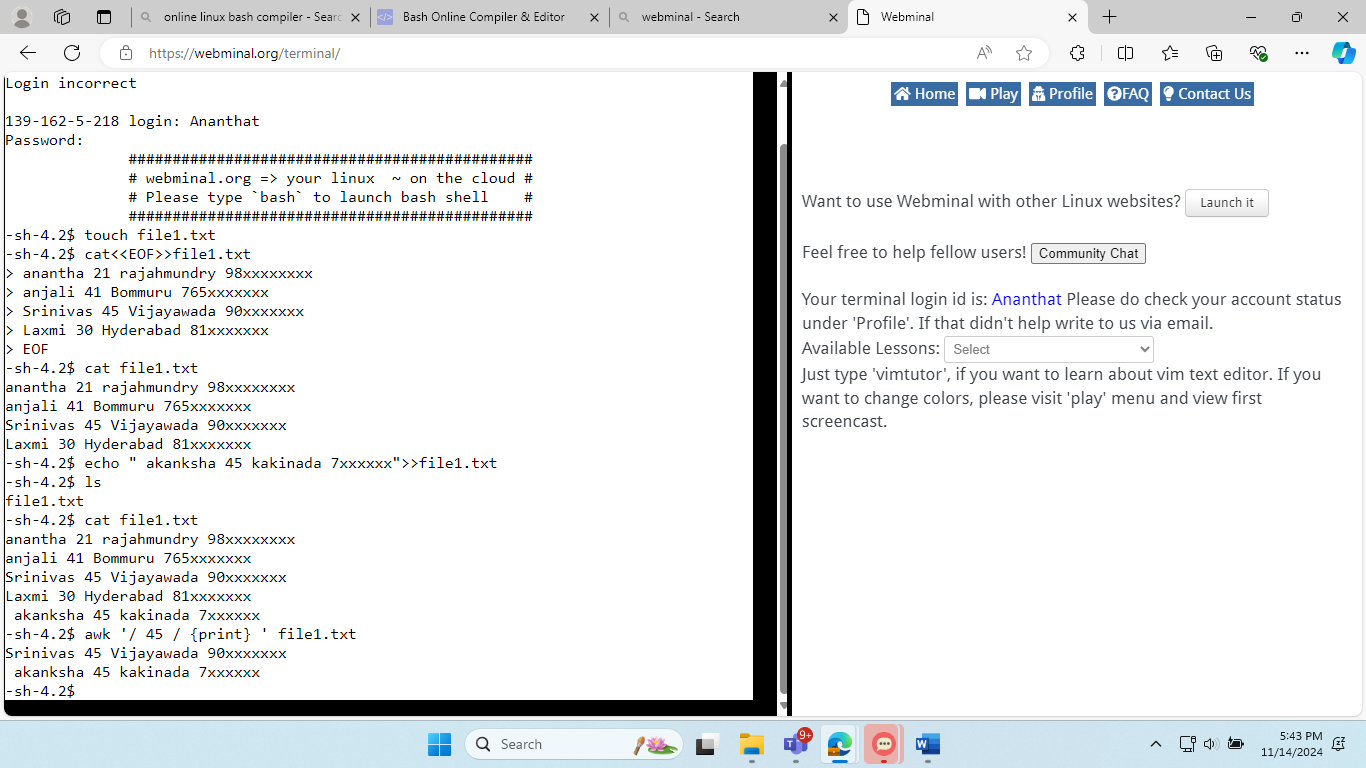
Description automatically generated

**Commands: Selected text processing utilities-**

* **awk** – **(Aho, Weinberger, and Kernighan)**

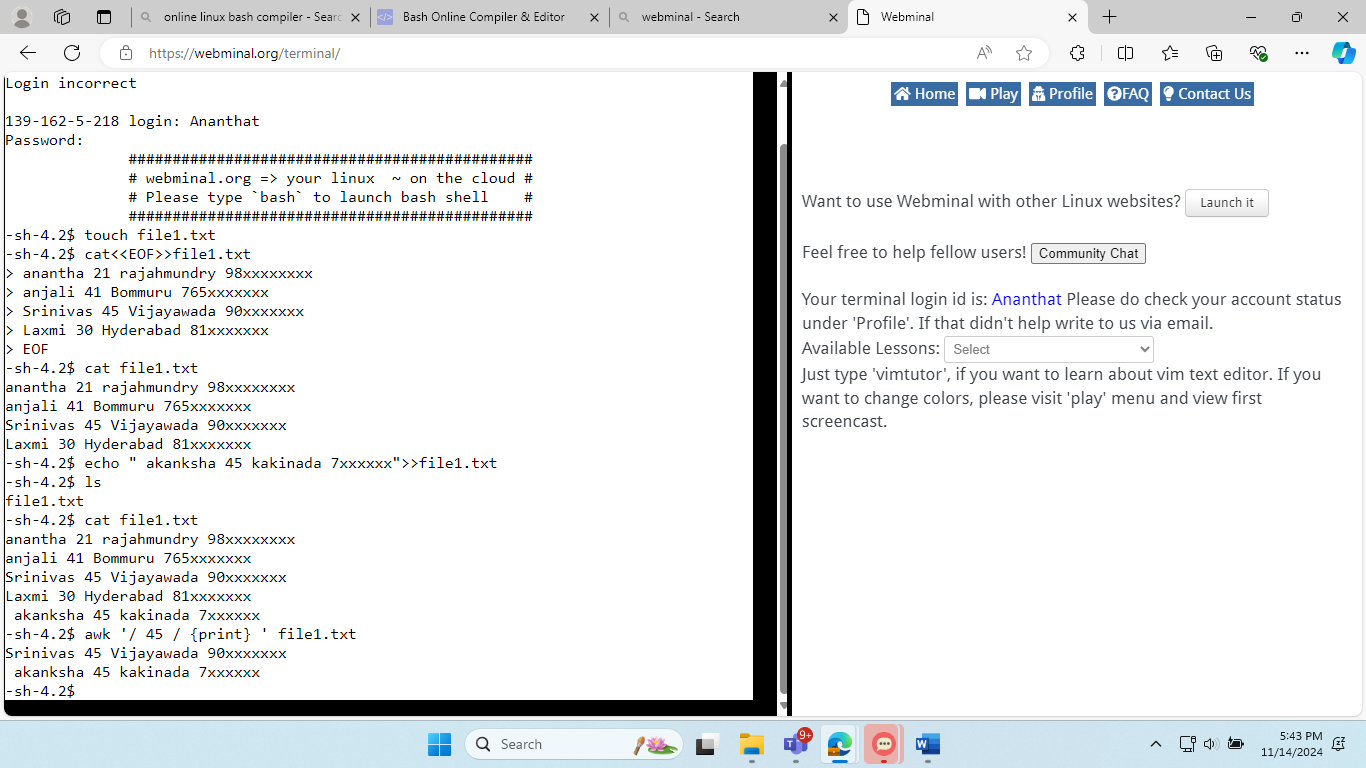
It is a versatilke text processing tool for extracting and manipulating data. It processes text files line by line, allowing you to perform calculations, format output, and generate reports.

Syntax : awk options 'selection \_criteria {action }' input-file > output-file



* **cat**-**("concatenate")**

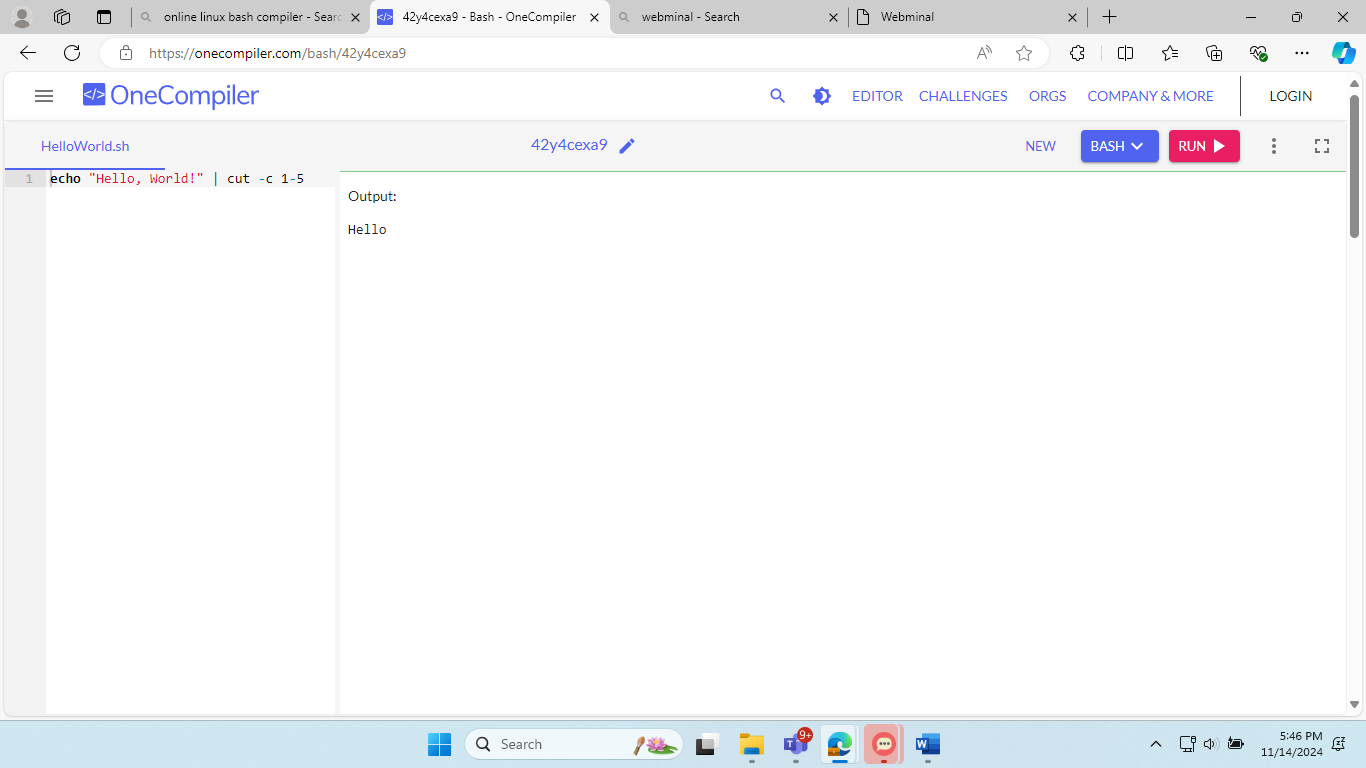
cat (short for "concatenate") is a simple but powerful command in Linux that is primarily used to display the content of text files in the terminal. It's often used for quick inspection of file contents.



* **cut** -

cut is used for extracting sections from each line of files. The cut command in linux is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by byte position, character, and field.

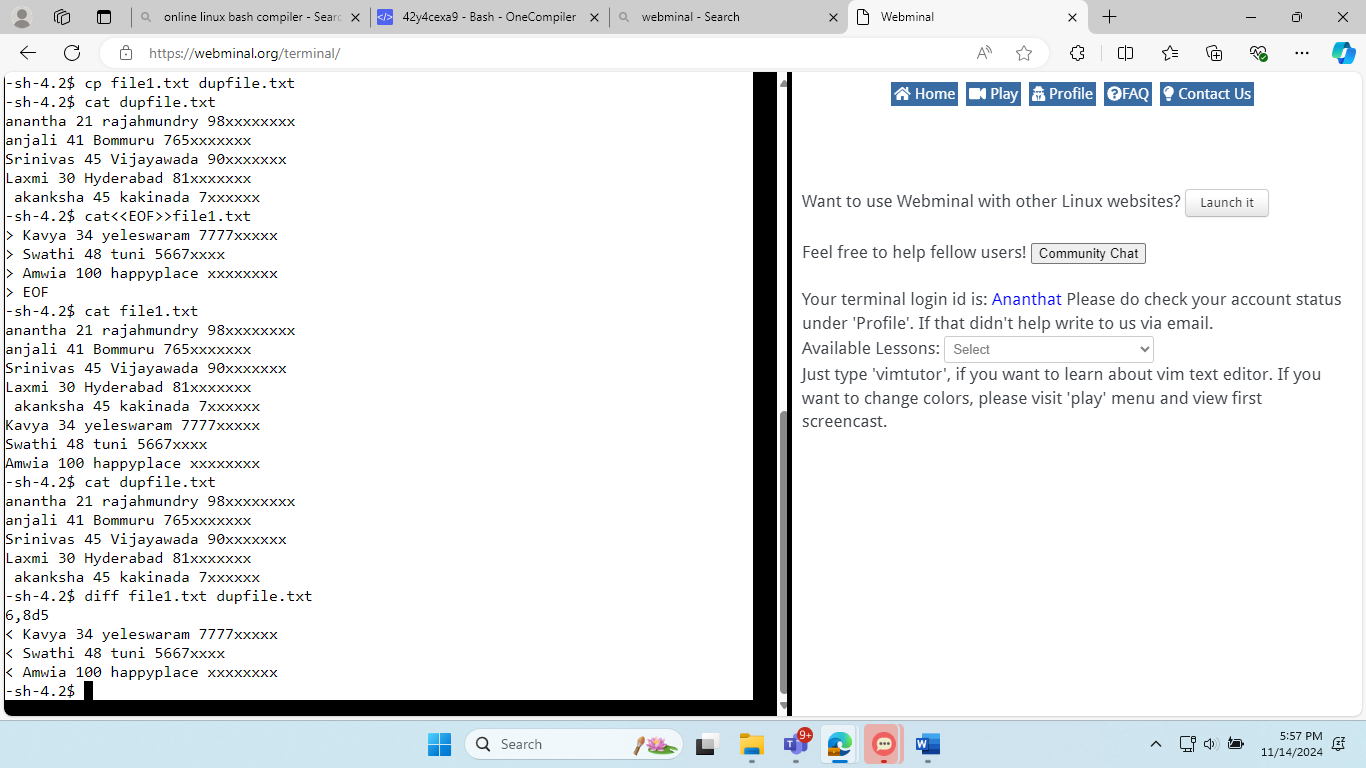
Syntax: cut OPTION... [FILE]...



* **diff-**

diff stands for **difference** .The diff command is a versatile utility that is pre-installed on most Linux distributions. Its primary purpose is to compare the contents of two files and display the differences between them.

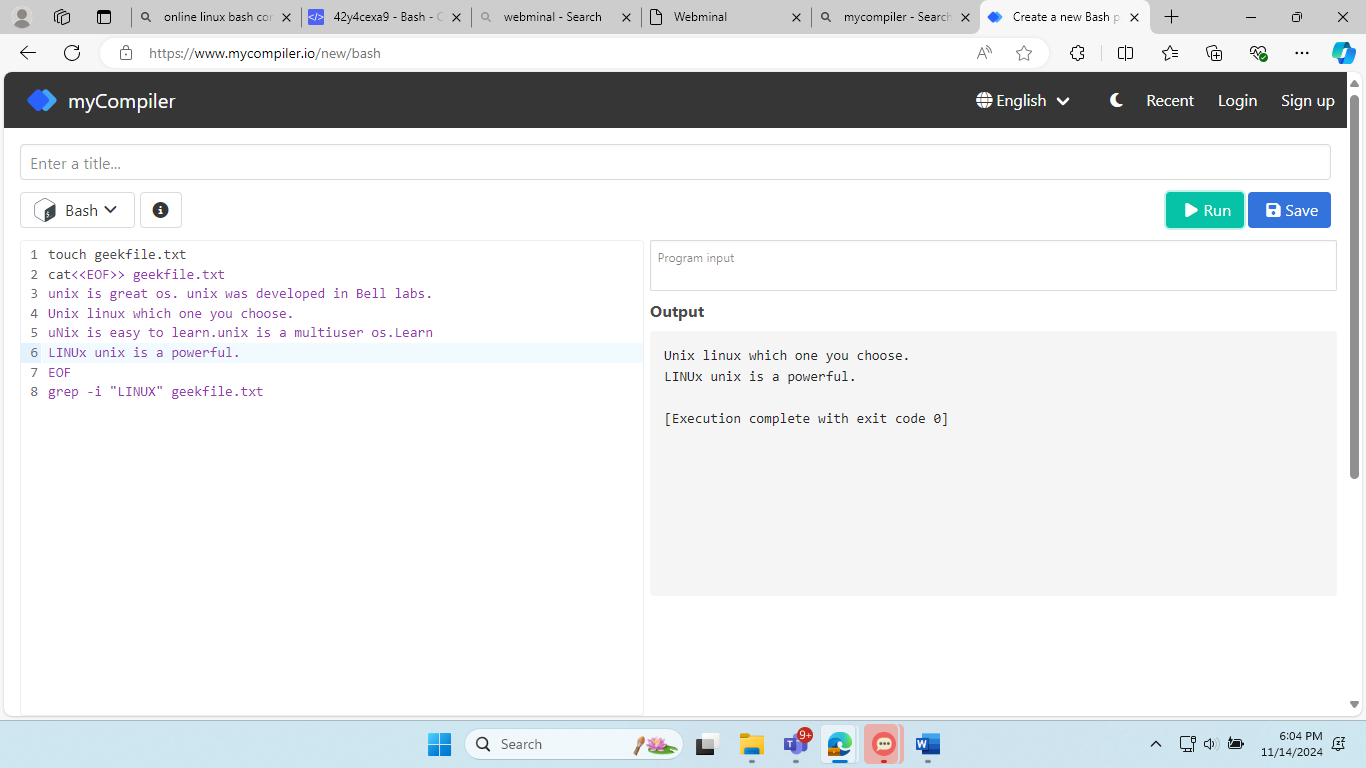
Basic syntax of the diff command is as follows:

 diff [OPTION]... FILE1 FILE2

* **grep -(Global Regular Expression Print)**

grep is used for searching text using patterns.Regular expressions provide a powerful way to specify complex search patterns, including character sequences, wildcards, and repetition rules

Syntax:grep [options] pattern [files]



* **head-**

The head command displays the first *n* lines of a text file. The default number of lines in *10* if *n* is not specified.

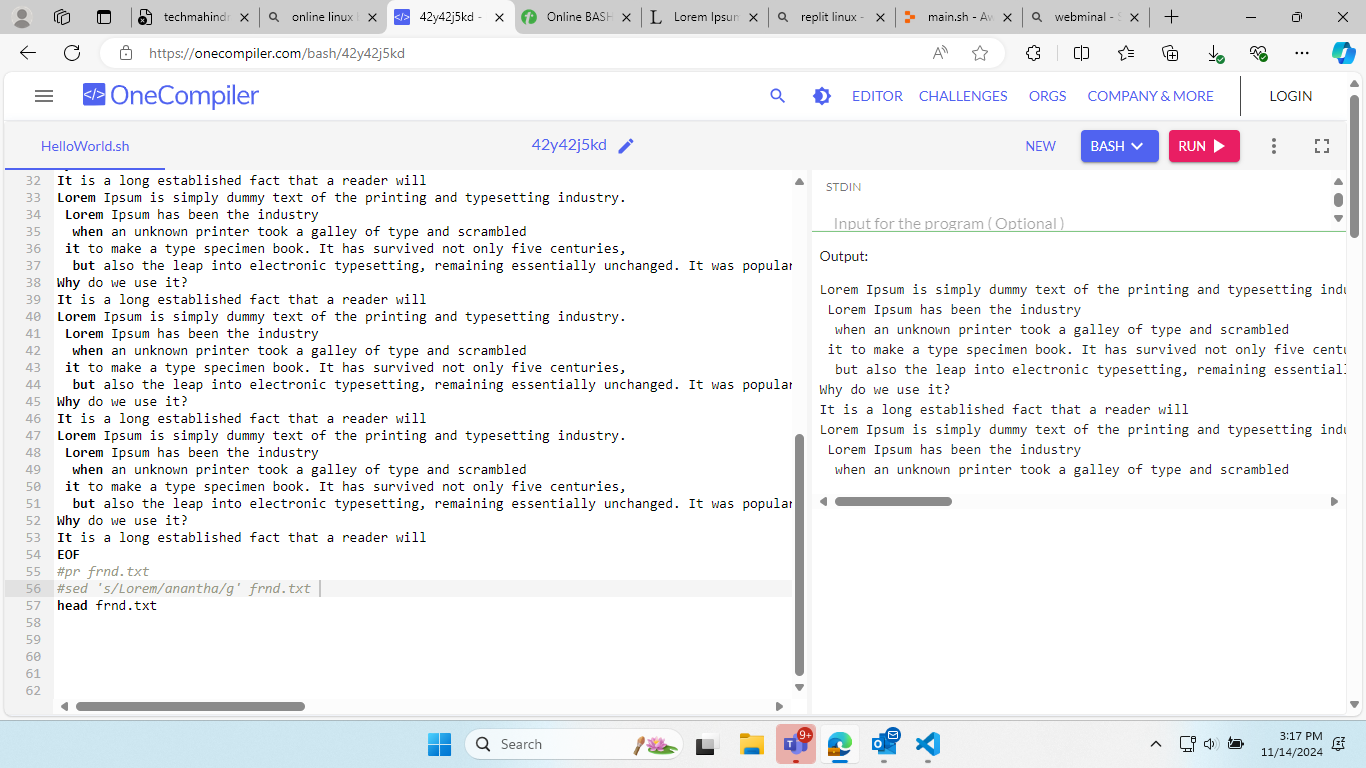
Syntax:

To display the first *10* lines we write

head filename.txt

To display the first n lines we write,

head -n filename.txt



* **less-**

The ‘less’ command in Linux is an indispensable utility for browsing the contents of text files interactively. Unlike traditional text editors, ‘less’ allows you to view text files page by page without loading the entire file into memory.

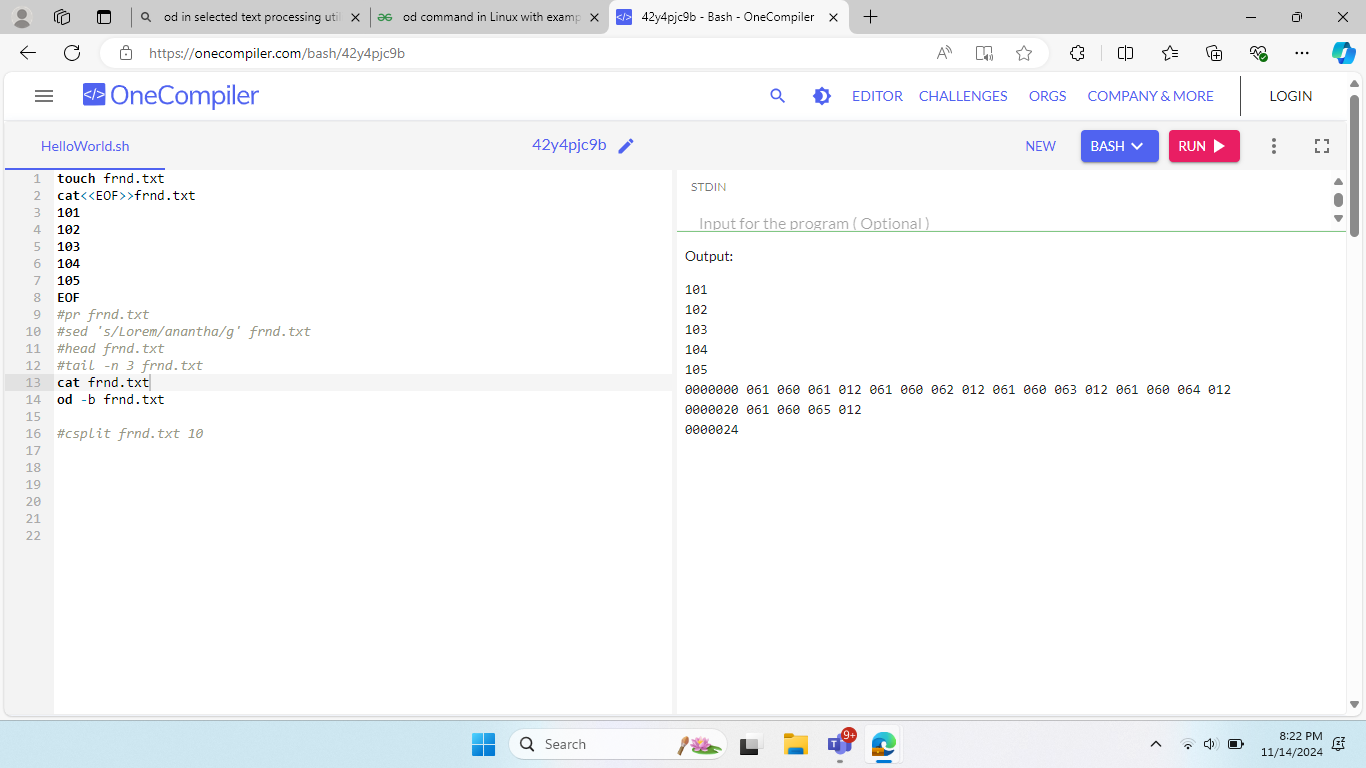
Syntax :less [options] filename

* **od-**

*od* stands for octal dump, this command is used to display contents of a file in different formats such as *octal*, *hexadecimal*, *ASCII* characters, *decimals*.

To display a file in *octal* format we write,

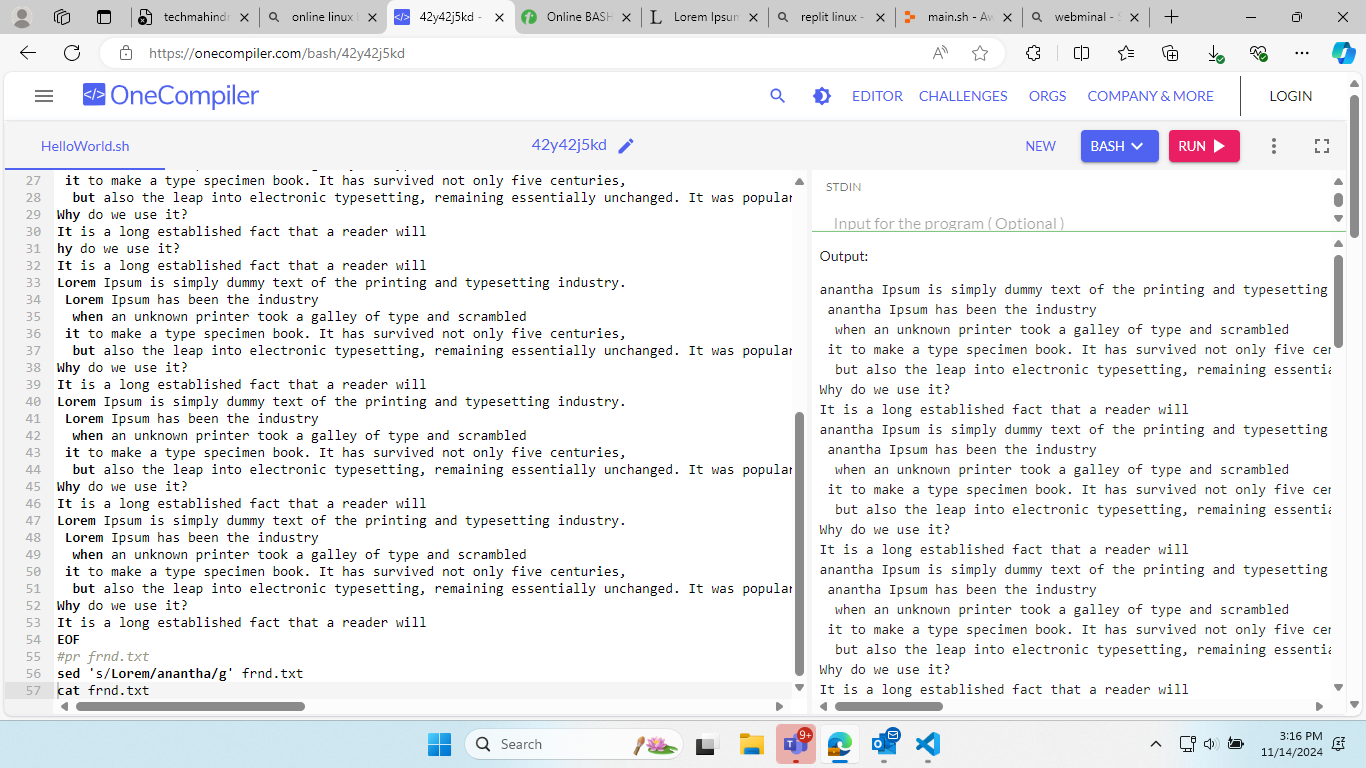
Syntax :od -b test.txt



* **sed – (Stream Editor)**

sed is a stream editor that enables non-interactive text manipulation. It allows you to modify, replace, or delete text patterns in files. It works by processing text line by line, applying a set of editing commands specified in a script or directly on the command line.

Syntax:sed [OPTIONS] 'SCRIPT' [INPUTFILE...]



* **Sort-**

We can also sort output alphabetically using the sort command.

Syntax:

To sort the file test.txt we write,

sort test.text

Given a file of numbers we can sort it numerically using the *-n* option,

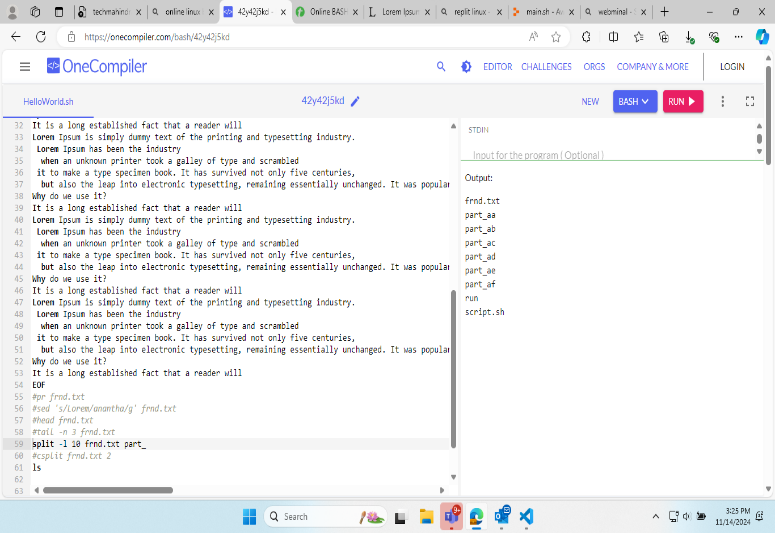
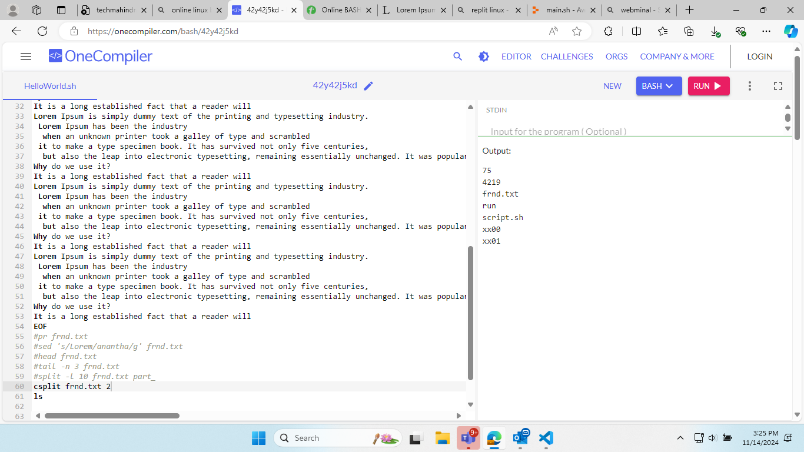
A screenshot of a computer

Description automatically generated sort -n numericalFile.txt

* **Split-**

to split a text file based on a regular expression with the help of a few popular text-processing utilities in Linux.The standard use case for *csplit* utility is for text files containing headers or titles and using them as a delimiter. Let’s split our input file using the regex for the header

Syntax: $ csplit tables.txt

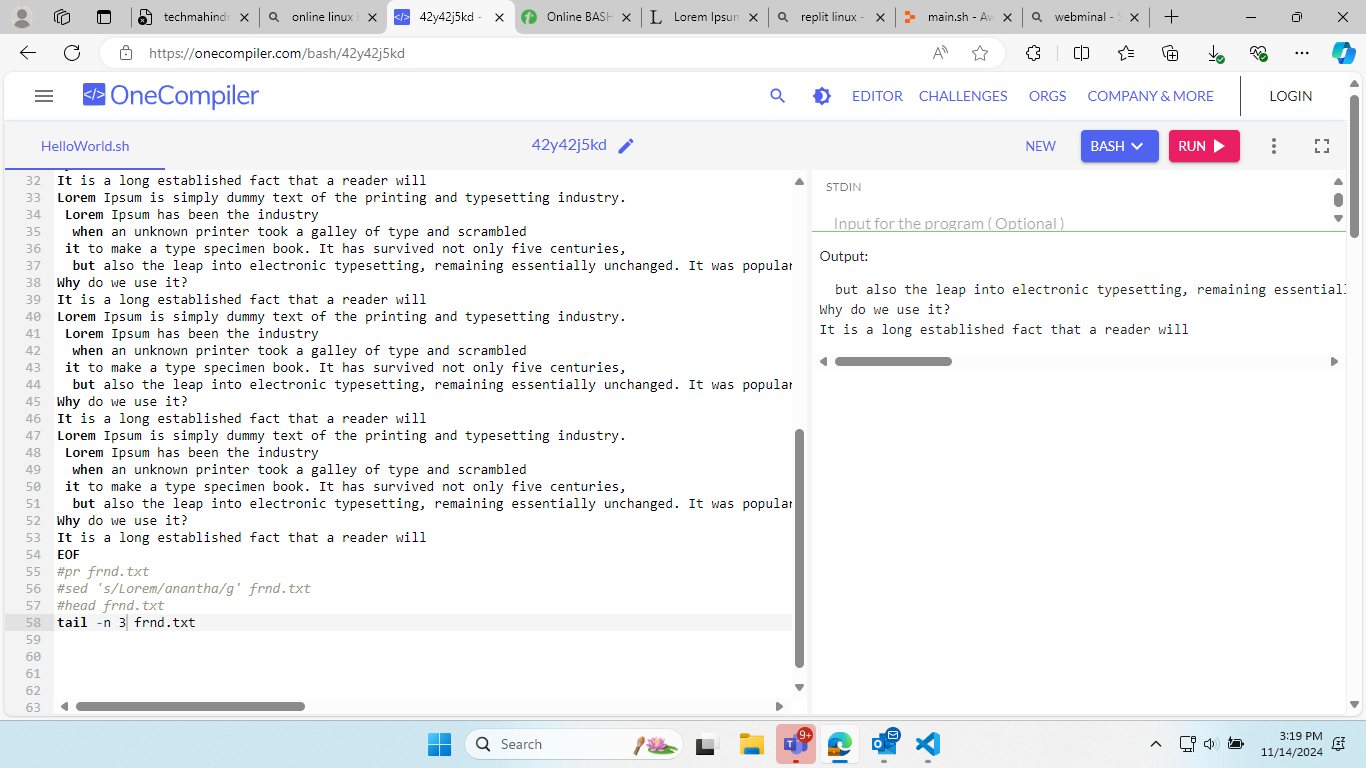


* **Tail-**

The tail command works the same as the head command but in reverse, that is it displays lines from the bottom of a text file.

To display the last 10 lines(default), we write,

Syntax: tail test.txt



* **Tr-**

 It supports a range of transformations including uppercase to lowercase, squeezing repeating characters, deleting specific characters, and basic find and replace. It can be used with UNIX pipes to support more complex translation. **tr stands for translate.**

Syntax: $ tr [OPTION] SET1 [SET2]

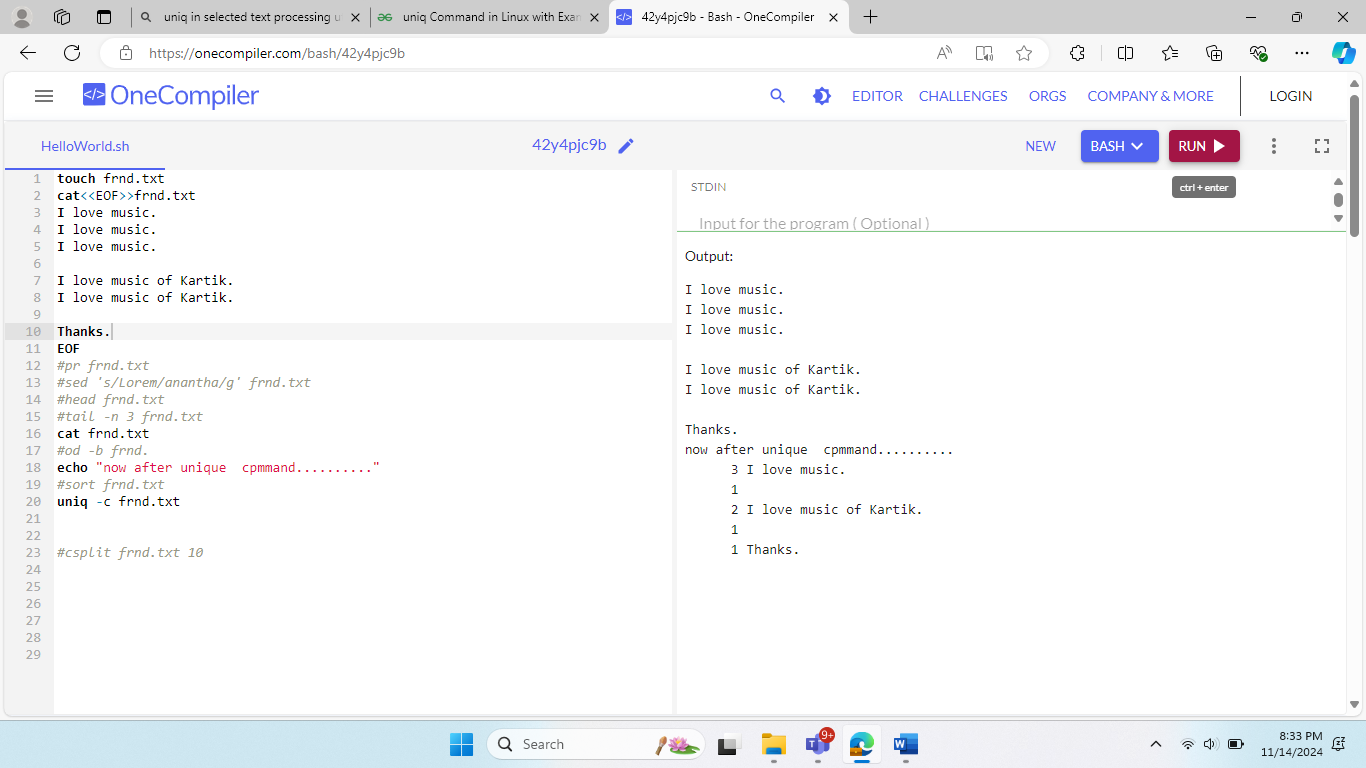
A screenshot of a computer

Description automatically generated

* **Uniq-**

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.

Syntax: uniq [OPTIONS] [INPUT\_FILE [OUTPUT\_FILE]]



* **Wc-**

*wc* stands for word count, this command counts the words, characters and lines in a file.

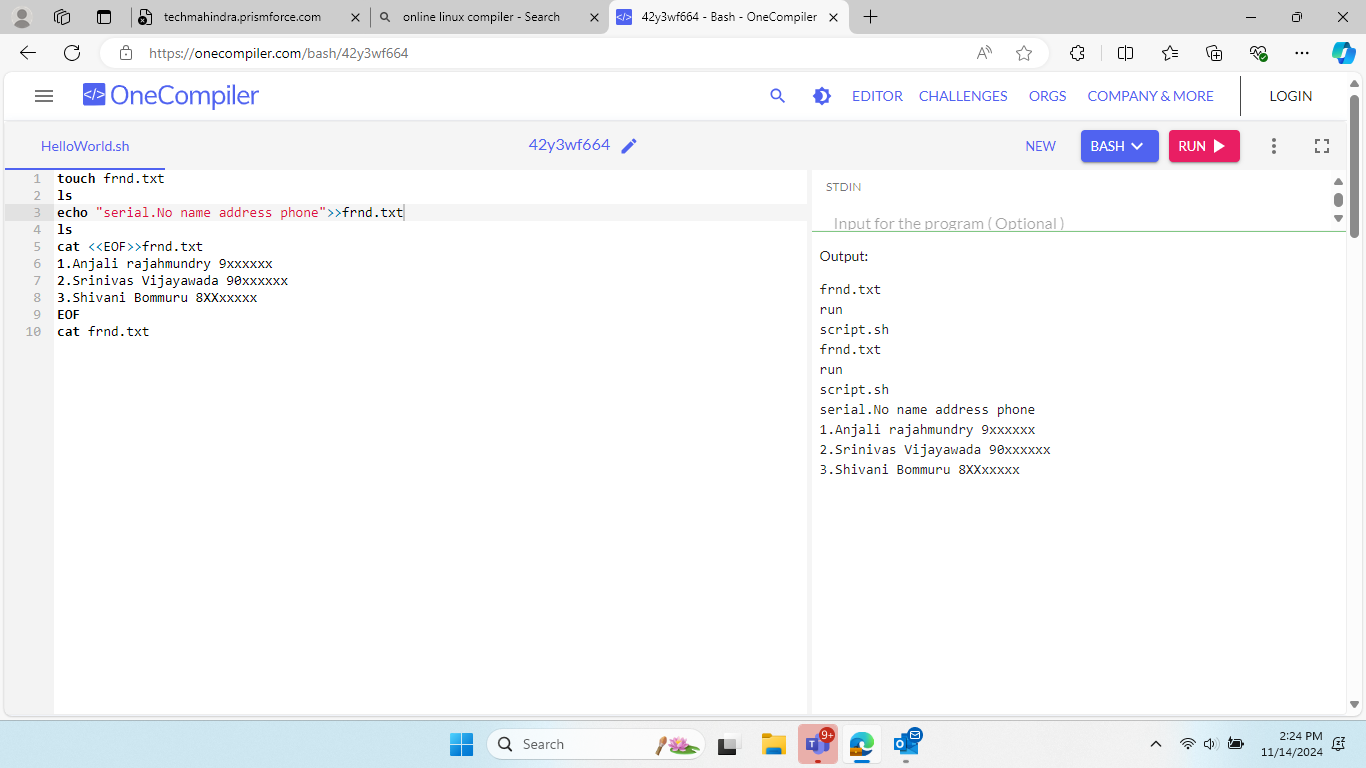
Syntax : wc test.txt

By default this command will give all three counts, that is words, characters and lines however we can use options to control this.

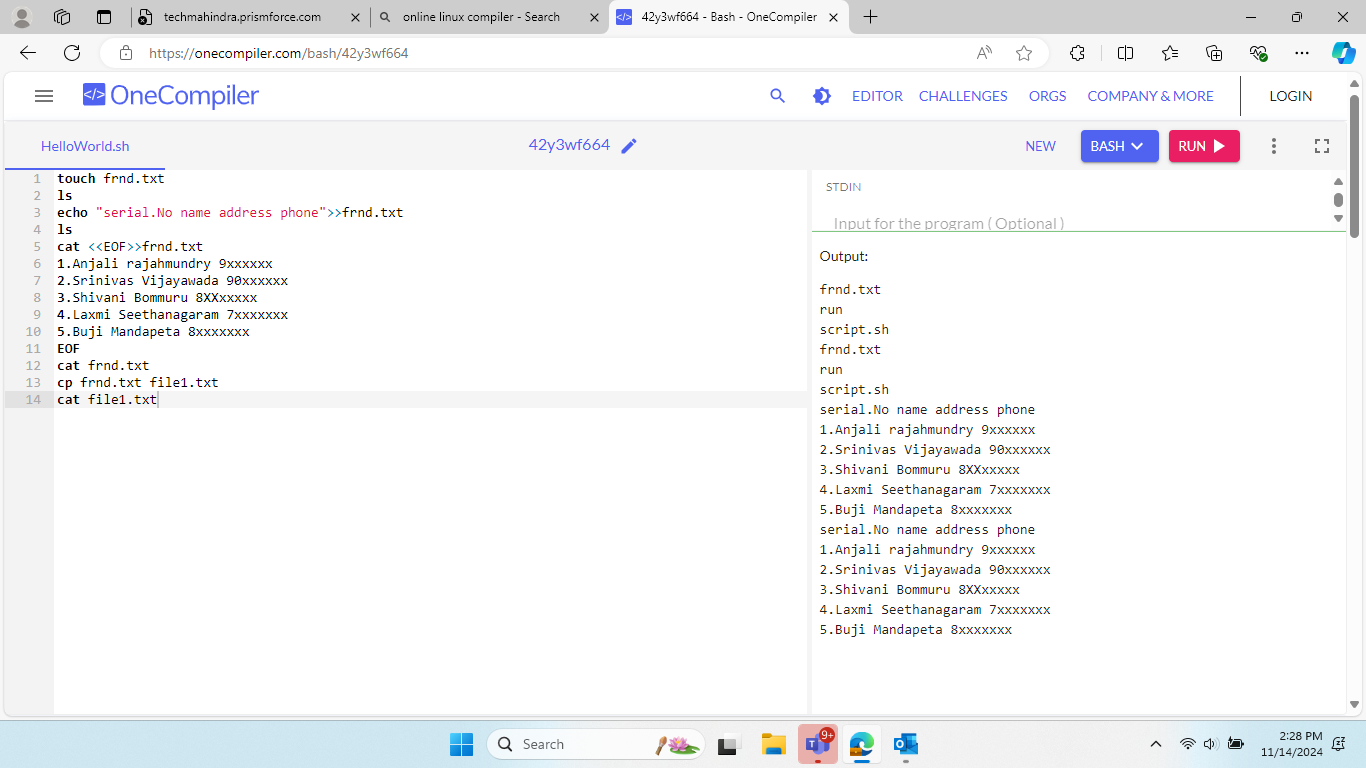
A screenshot of a computer

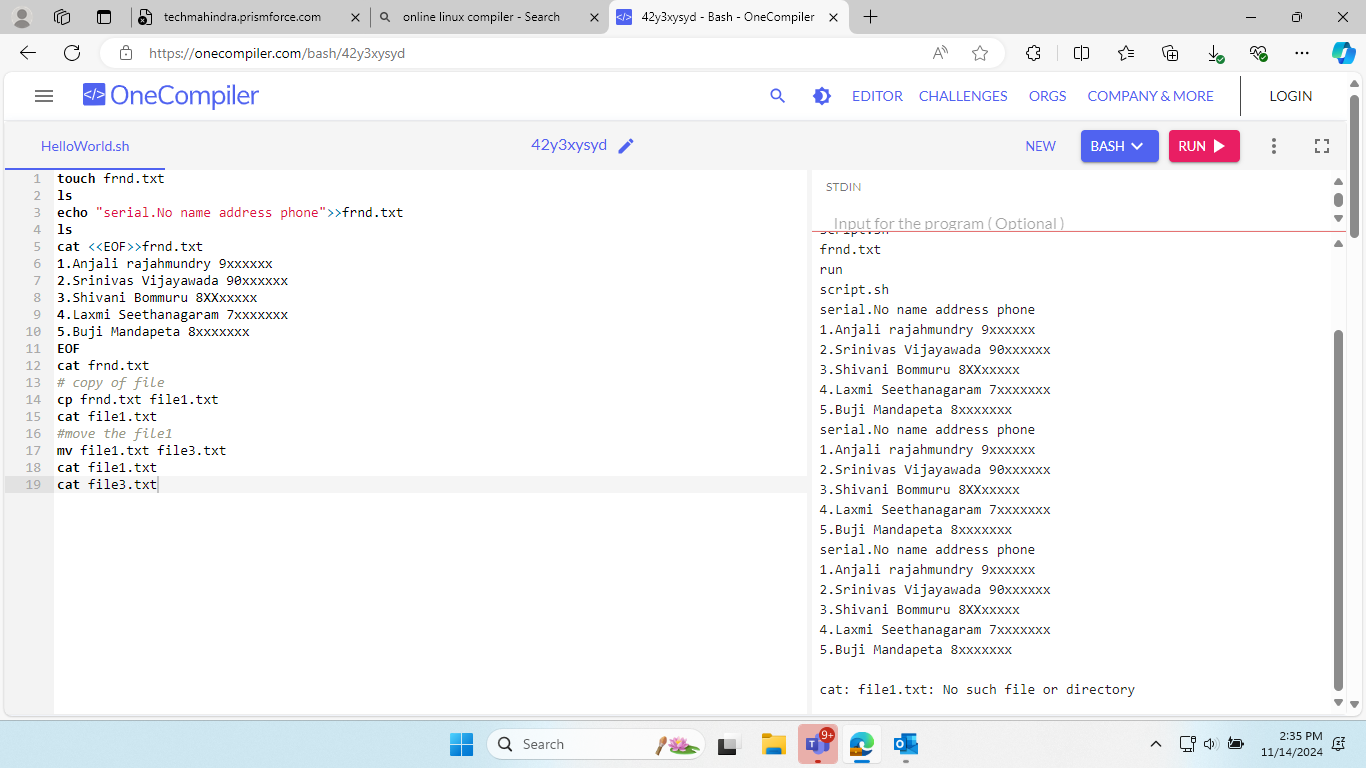
Description automatically generated Syntax: wc -l test.txt

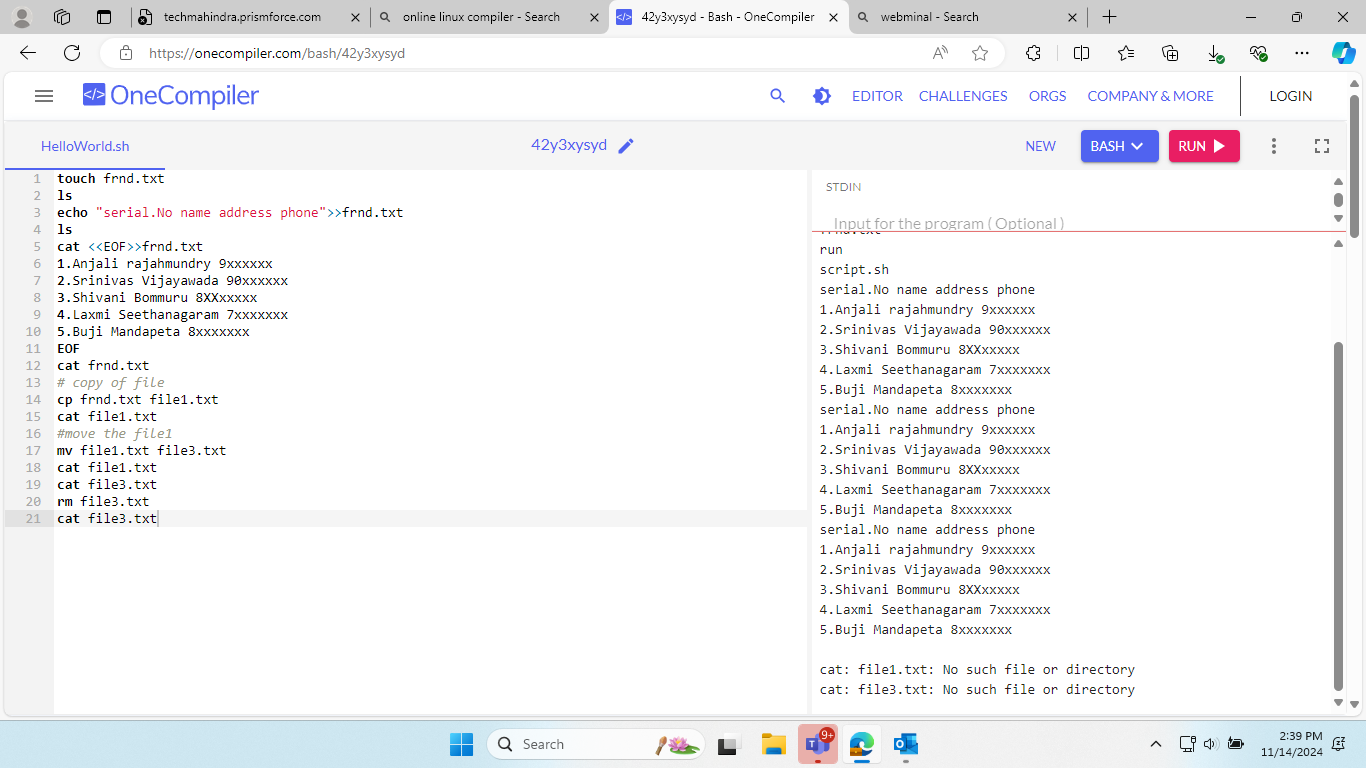
**Practice Sessions:**

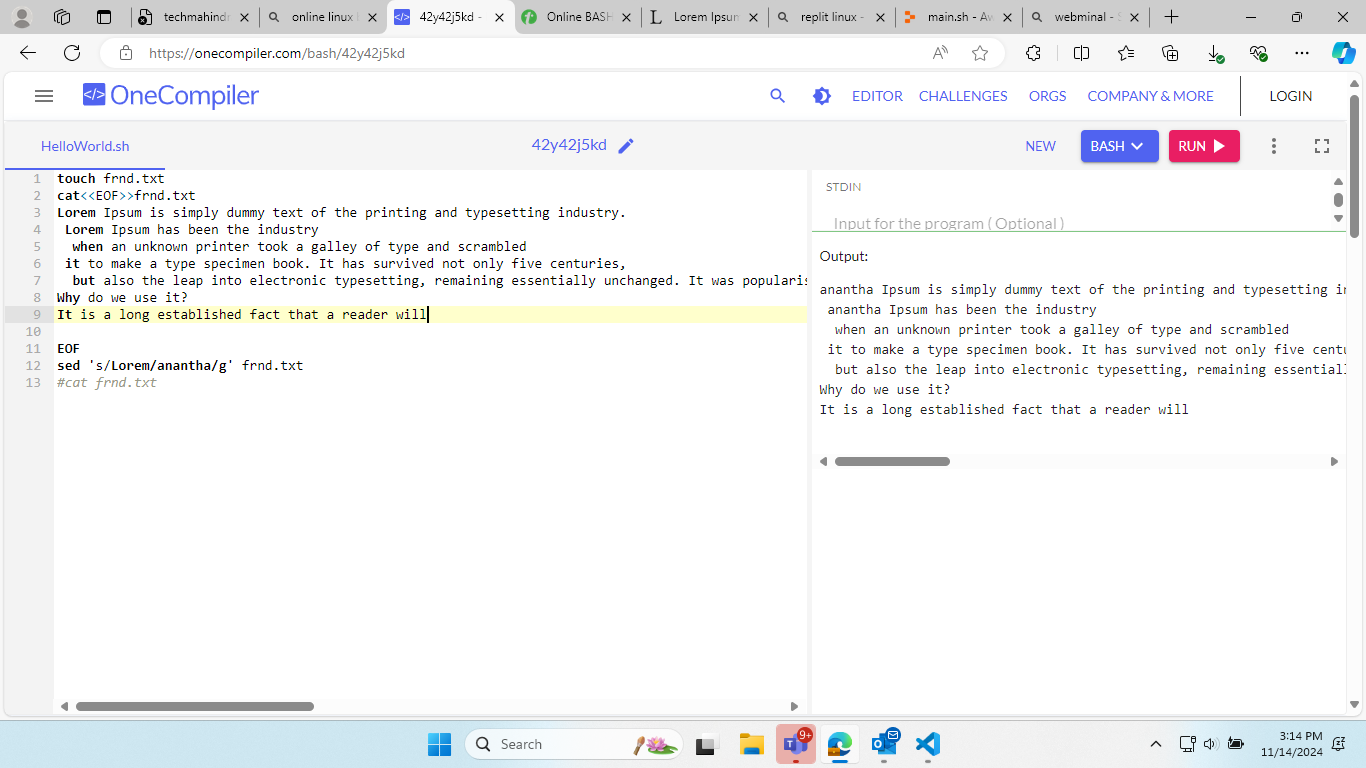
1.Create a file and enter the data in it

2. Copy the created file to the new file

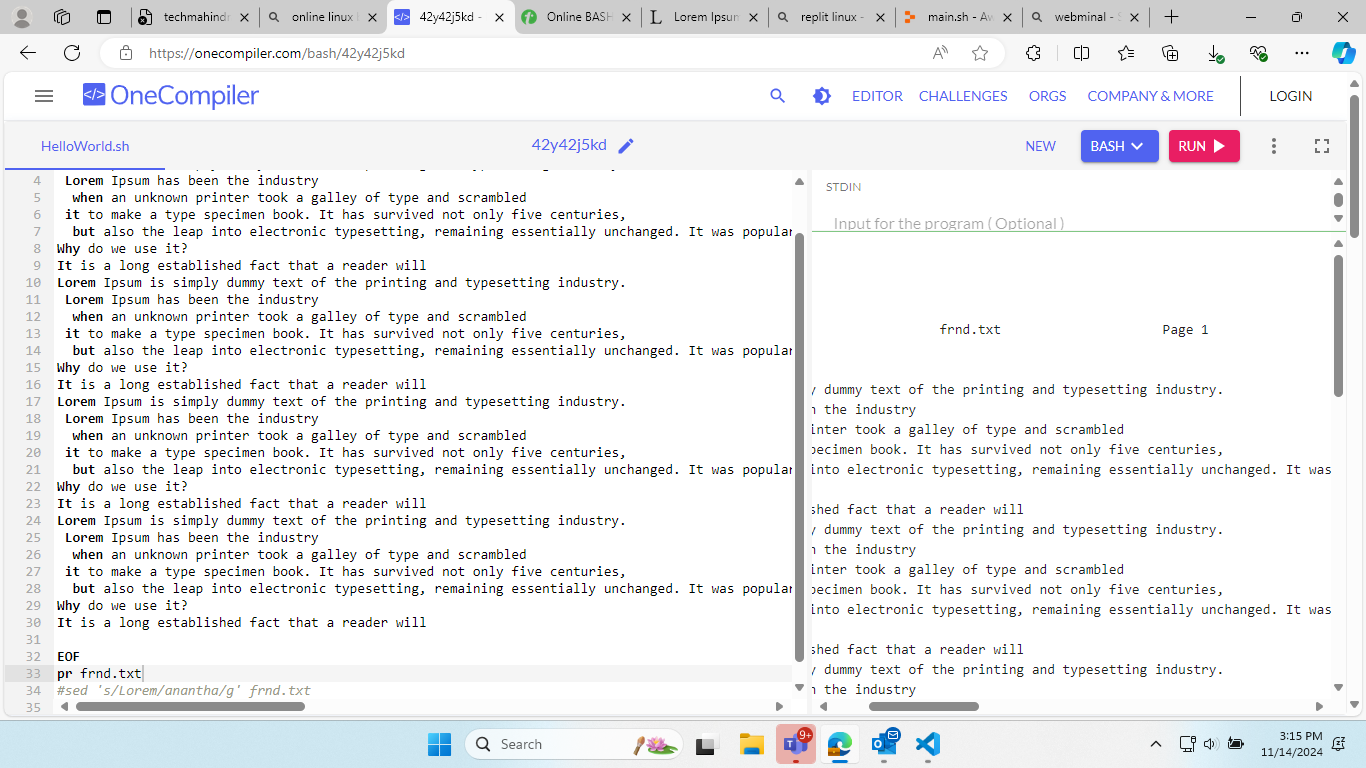


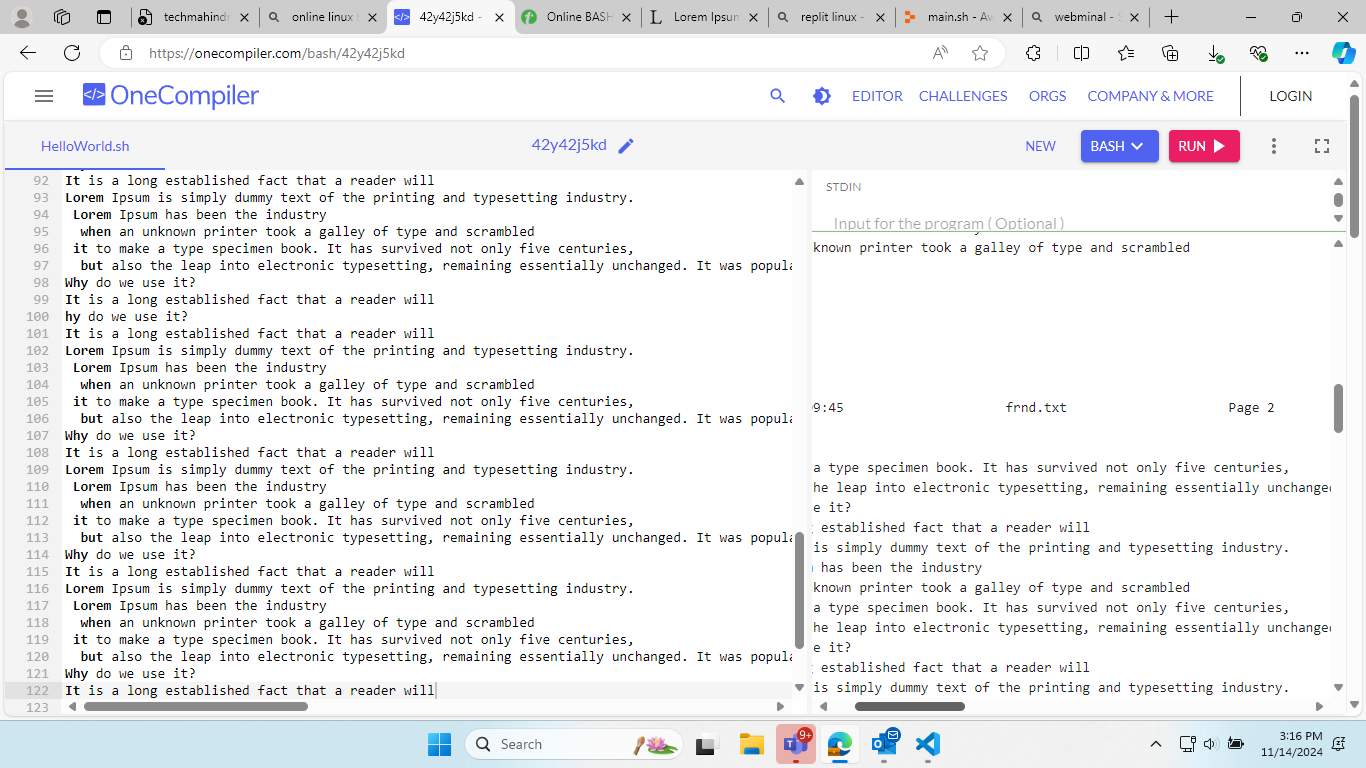
3.Move the file to new file

4.remove the file

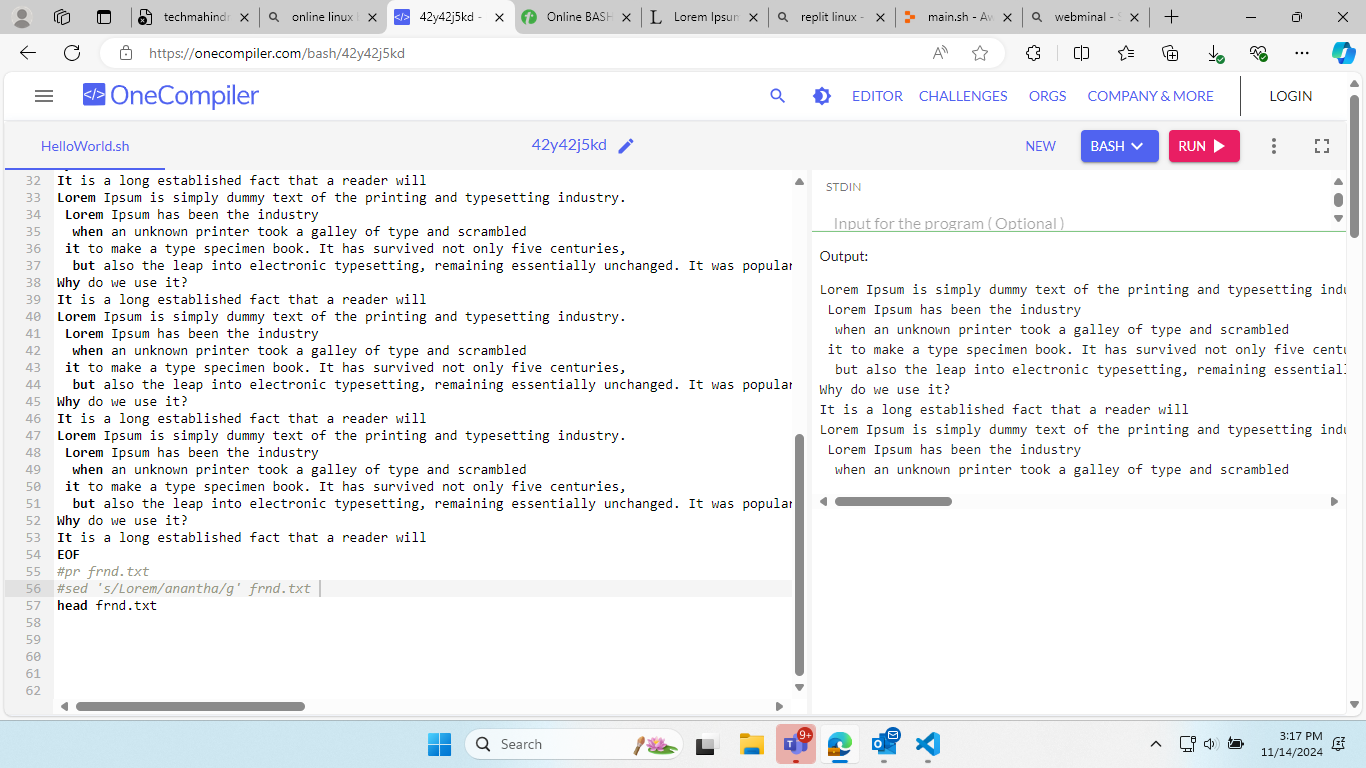
5.replace a word with new another word in the file

6.Create pages in the given file

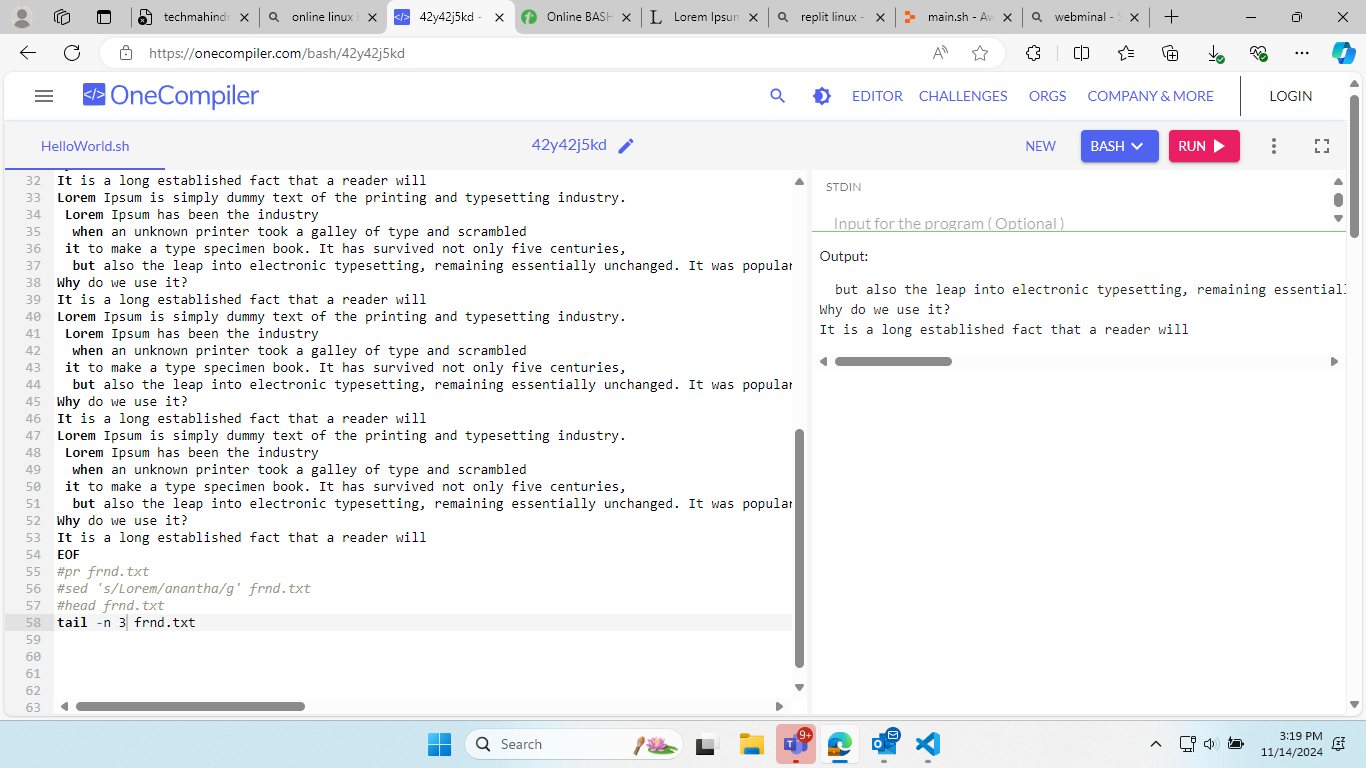




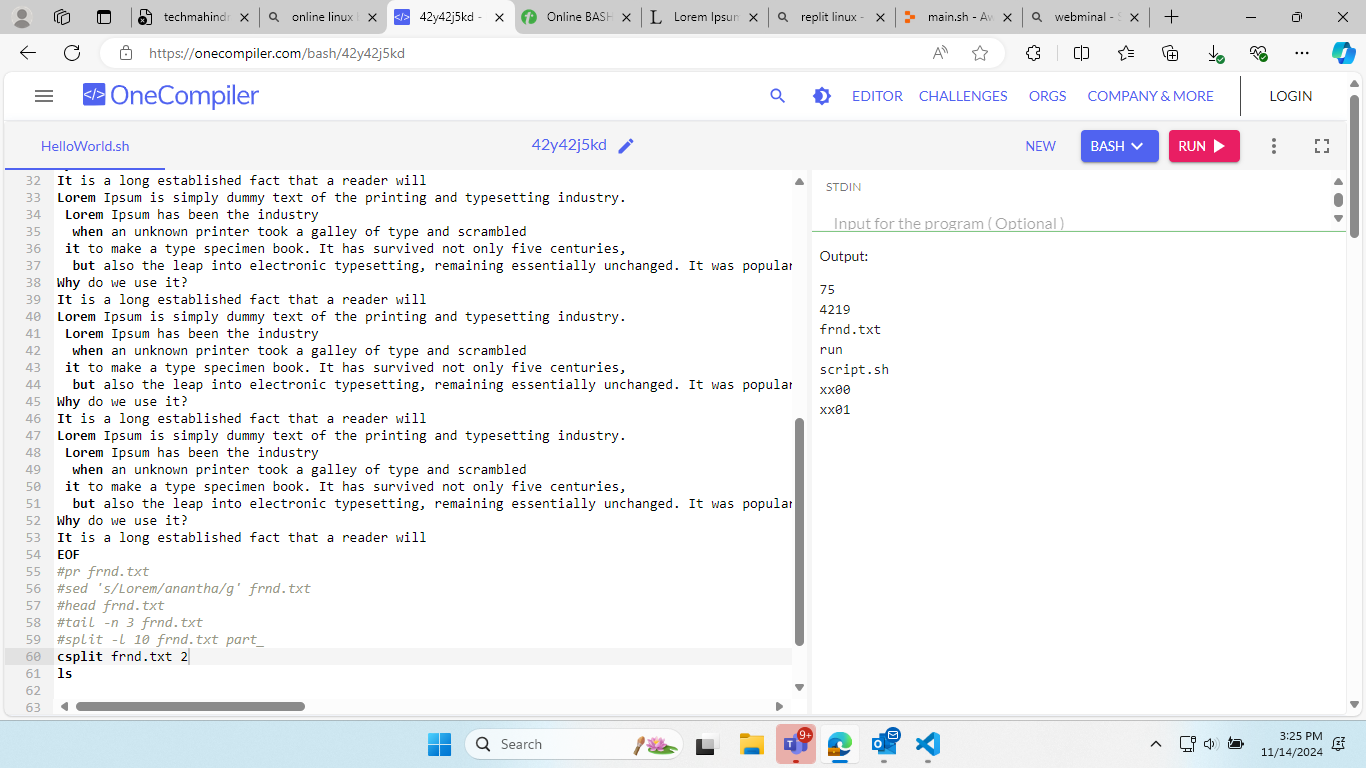
7.Print starting few lines of content in the file



8. print last few lines of content in the file



9.Split the file



10. Do the word count of the content in the file

