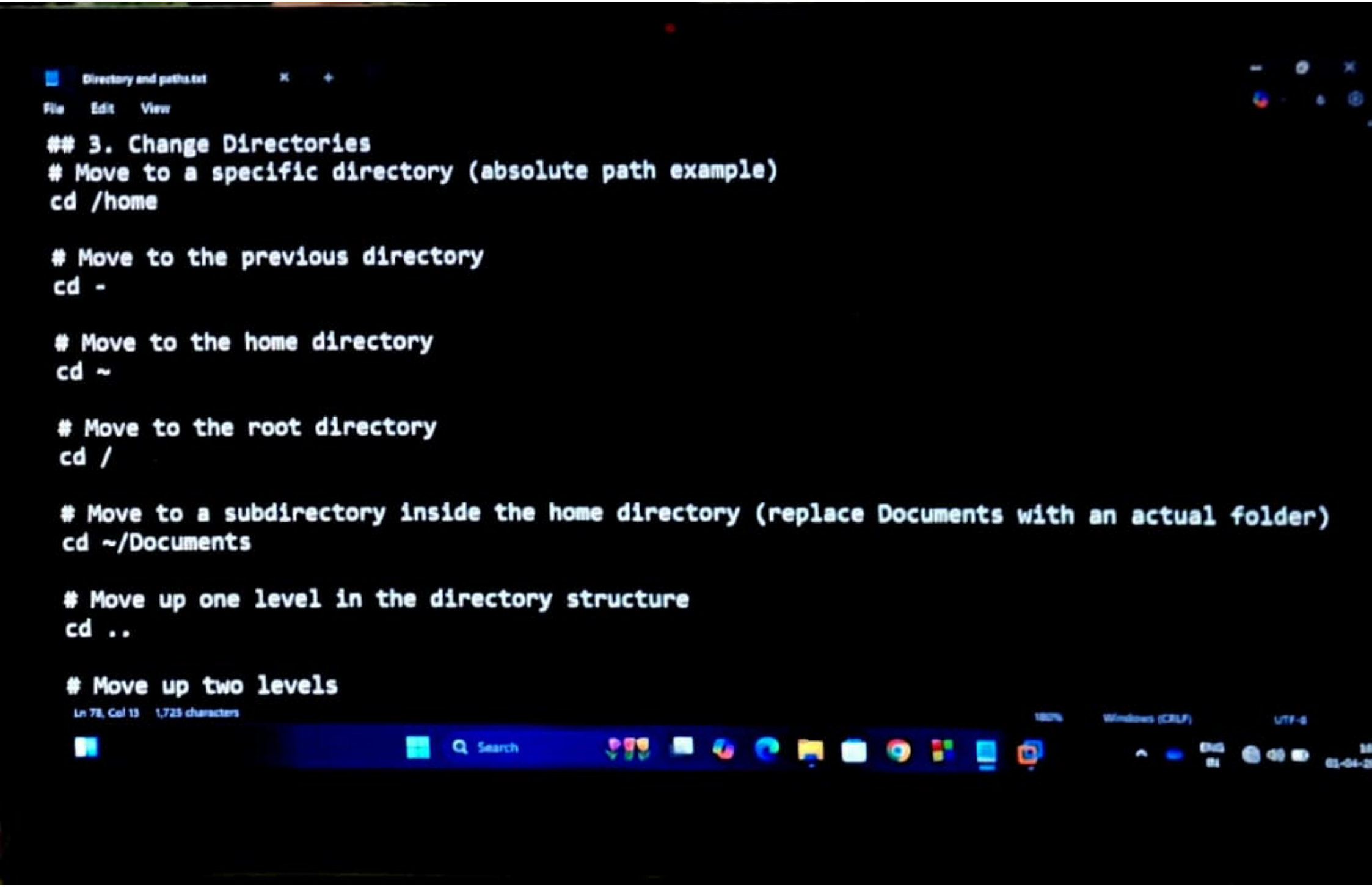
Directory and paths.txt

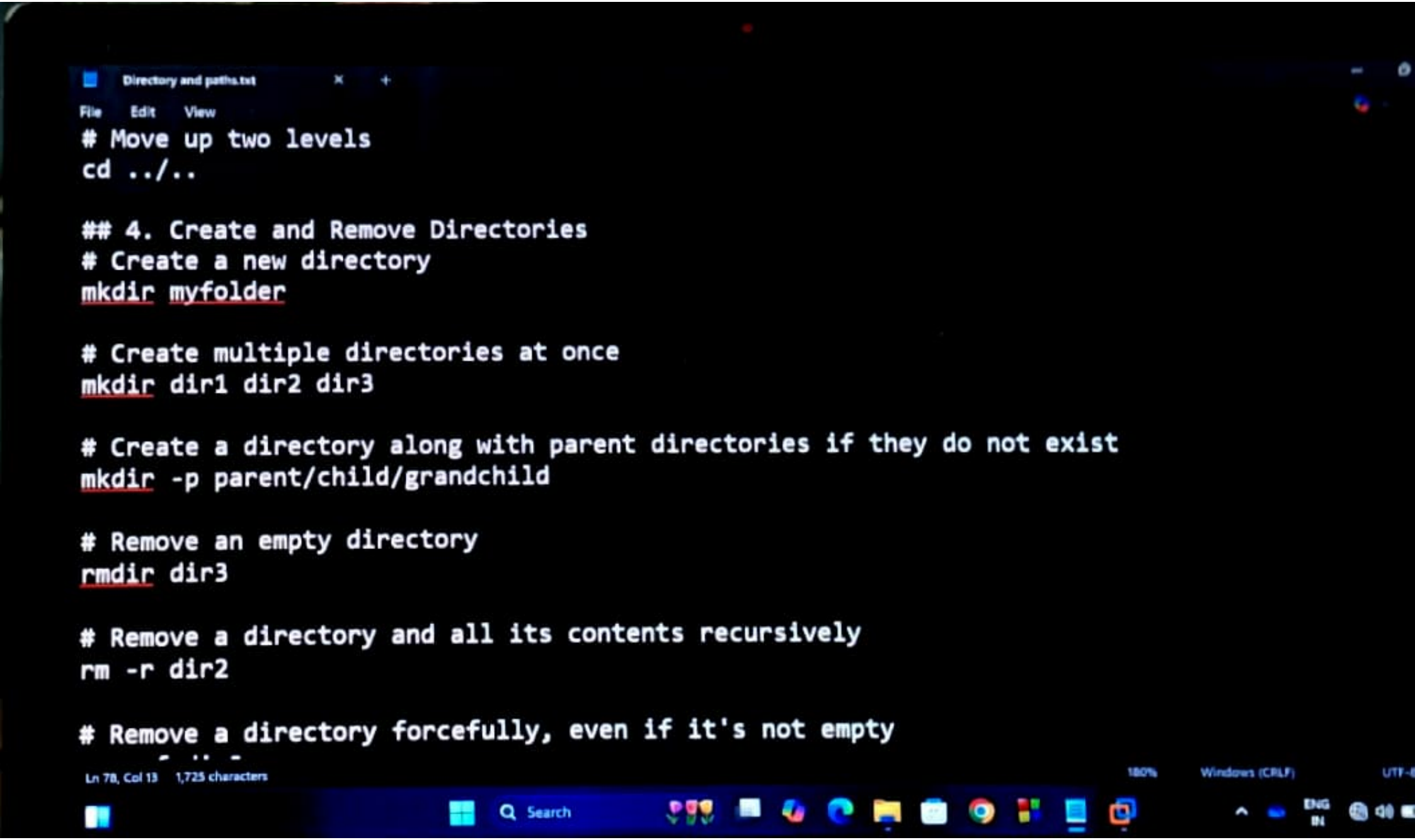
File Edit View

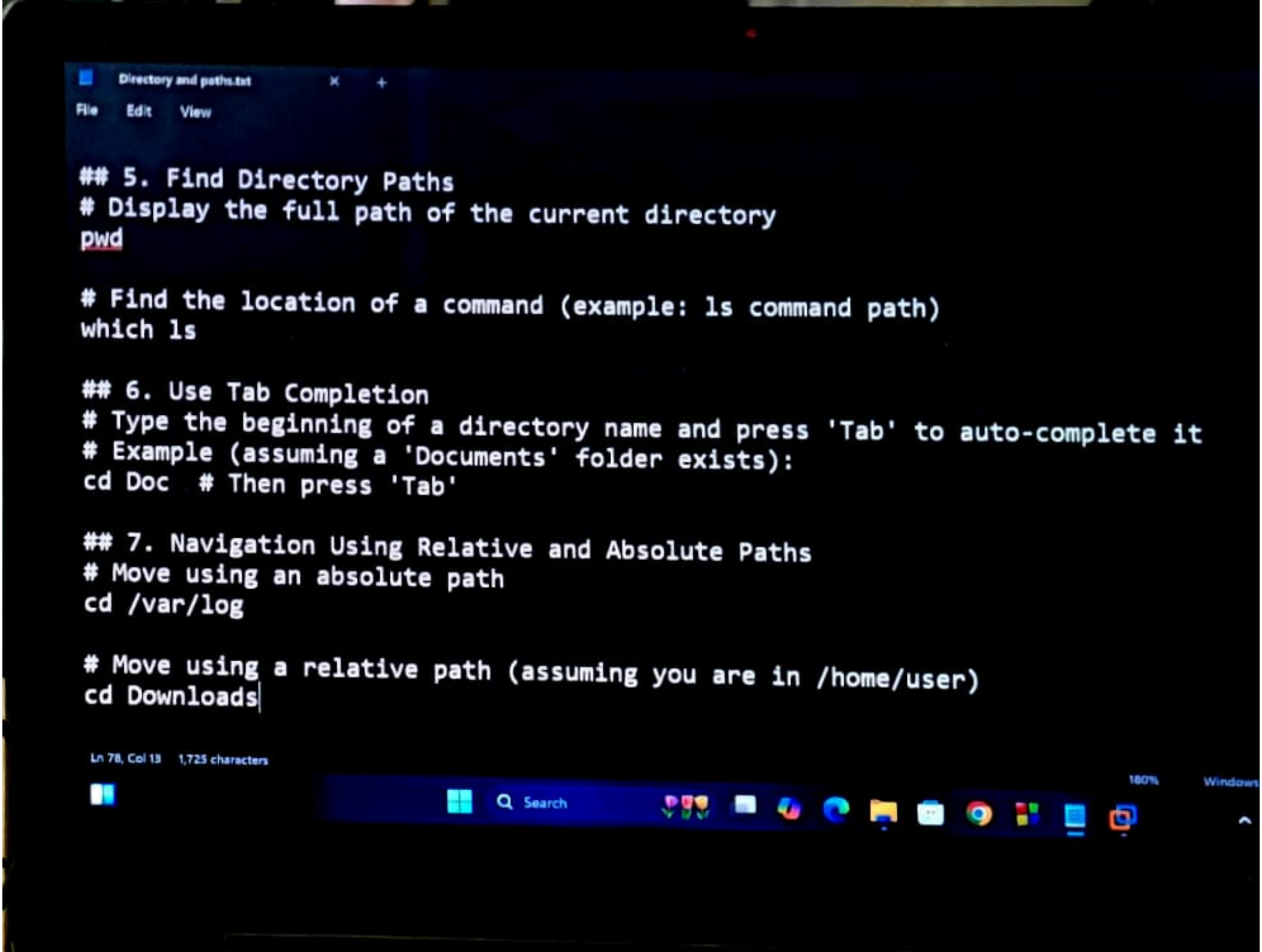
```
# Linux Directory Navigation Commands

## 1. Check Current Directory
# Display the full path of the current working directory
pwd

## 2. List Directory Contents
# List files and directories in the current location
ls

# List files with detailed information (permissions, owner, size, timestamp)
ls -l

# List all files, including hidden ones (starting with .)
ls -a

# List files with human-readable sizes
ls -lh

## 3. Change Directories
```

Ln 78, Col 13 1,725 characters 180% Windows (CRL

Search


Directory and paths.txt
File Edit View
3. Change Directories
Move to a specific directory (absolute path example)
cd /home
Move to the previous directory
cd -
Move to the home directory
cd ~
Move to the root directory
cd /
Move to a subdirectory inside the home directory (replace Documents with an actual folder)
cd ~/Documents
Move up one level in the directory structure
cd ..
Move up two levels
Ln 78, Col 13 1,725 characters
100%
Windows (CRLF)
UTF-8
Search


Directory and paths.txt
File Edit View
Move up two levels
cd ../..
4. Create and Remove Directories
Create a new directory
mkdir myfolder
Create multiple directories at once
mkdir dir1 dir2 dir3
Create a directory along with parent directories if they do not exist
mkdir -p parent/child/grandchild
Remove an empty directory
rmdir dir3
Remove a directory and all its contents recursively
rm -r dir2
Remove a directory forcefully, even if it's not empty
Ln 78, Col 13 1,725 characters
180%
Windows (CRLF)
Search


Directory and paths.txt
File Edit View
5. Find Directory Paths
Display the full path of the current directory
pwd
Find the location of a command (example: ls command path)
which ls
6. Use Tab Completion
Type the beginning of a directory name and press 'Tab' to auto-complete it
Example (assuming a 'Documents' folder exists):
cd Doc # Then press 'Tab'
7. Navigation Using Relative and Absolute Paths
Move using an absolute path
cd /var/log
Move using a relative path (assuming you are in /home/user)
cd Downloads
Ln 78, Col 13 1,725 characters
Search
180%
Windows

Directory and paths.txt

File Edit View

```
# Remove a directory forcefully, even if it's not empty
rm -rf dir2

## 5. Find Directory Paths
# Display the full path of the current directory
pwd

# Find the location of a command (example: ls command path)
which ls

## 6. Use Tab Completion
# Type the beginning of a directory name and press 'Tab' to auto-complete it
# Example (assuming a 'Documents' folder exists):
cd Doc  # Then press 'Tab'

## 7. Navigation Using Relative and Absolute Paths
# Move using an absolute path
cd /var/log
```

Ln 78, Col 13 1,725 characters 180% Windows (CRLF)

Directory and paths.txt

File Edit View

```
## 3. Change Directories
# Move to a specific directory (absolute path example)
cd /home

# Move to the previous directory
cd -

# Move to the home directory
cd ~

# Move to the root directory
cd /

# Move to a subdirectory inside the home directory (replace Documents with an actu
folder)
cd ~/Documents

# Move up one level in the directory structure
cd ..
```

Ln 50, Col 15 | 1,725 characters | 190% | Windows (CRLF)

Search

ENG
IN

Directory and paths.txt

File Edit View

```
# Linux Directory Navigation Commands

## 1. Check Current Directory
# Display the full path of the current working directory
pwd

## 2. List Directory Contents
# List files and directories in the current location
ls

# List files with detailed information (permissions, owner, size, timestamp)
ls -l

# List all files, including hidden ones (starting with .)
ls -a

# List files with human-readable sizes
ls -lh
```

Ln 50, Col 15 1,725 characters 190% Windows (CRLF)

Search

Directory and paths.txt

File Edit View

```
# Move up two levels
cd ../..

## 4. Create and Remove Directories
# Create a new directory
mkdir myfolder

# Create multiple directories at once
mkdir dir1 dir2 dir3

# Create a directory along with parent directories if they do not exist
mkdir -p parent/child/grandchild

# Remove an empty directory
rmdir dir3

# Remove a directory and all its contents recursively
rm -r dir2
```

Ln 50, Col 15 | 1,725 characters | 190% | Windows (CRLF)

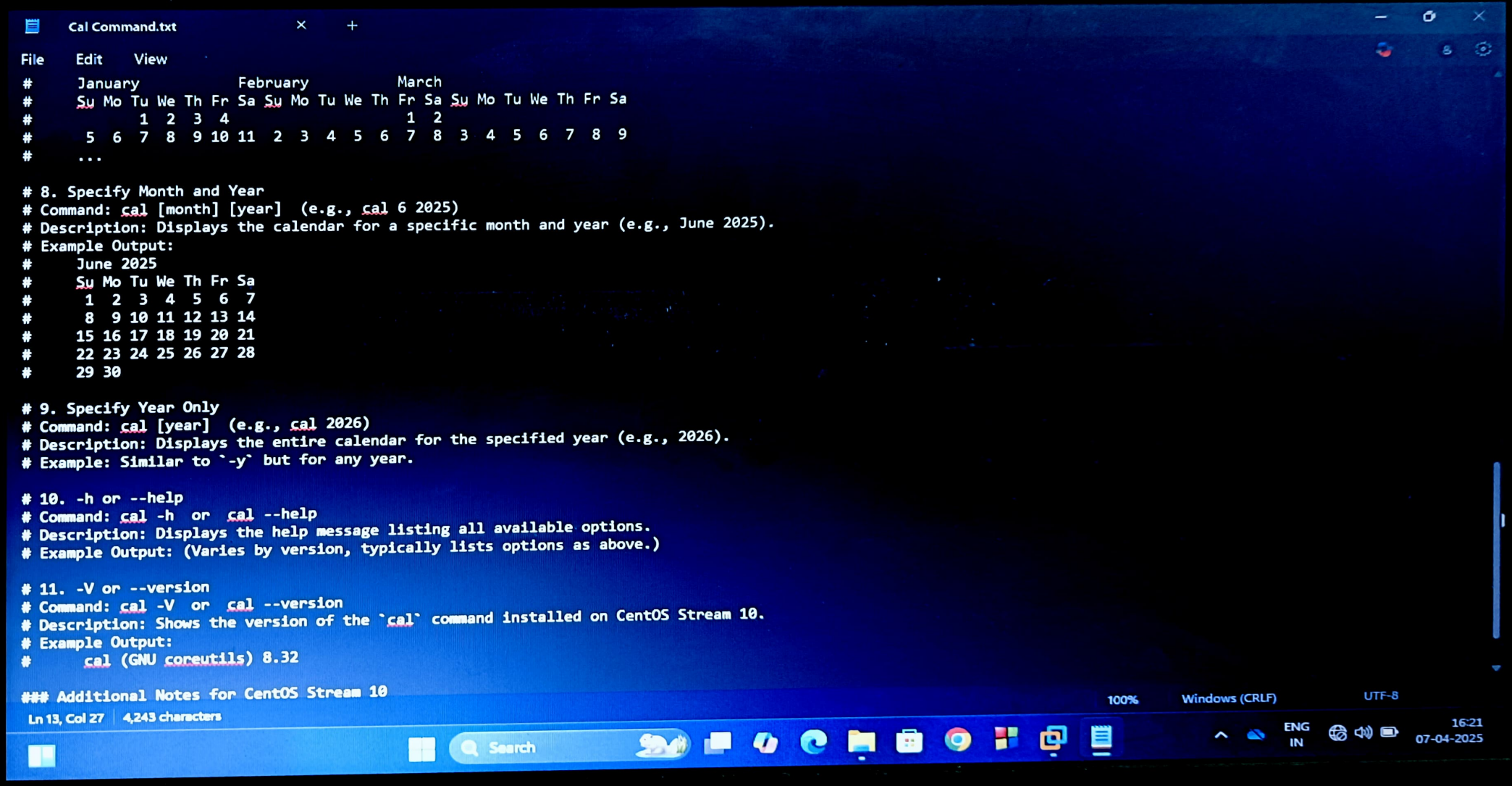
Search

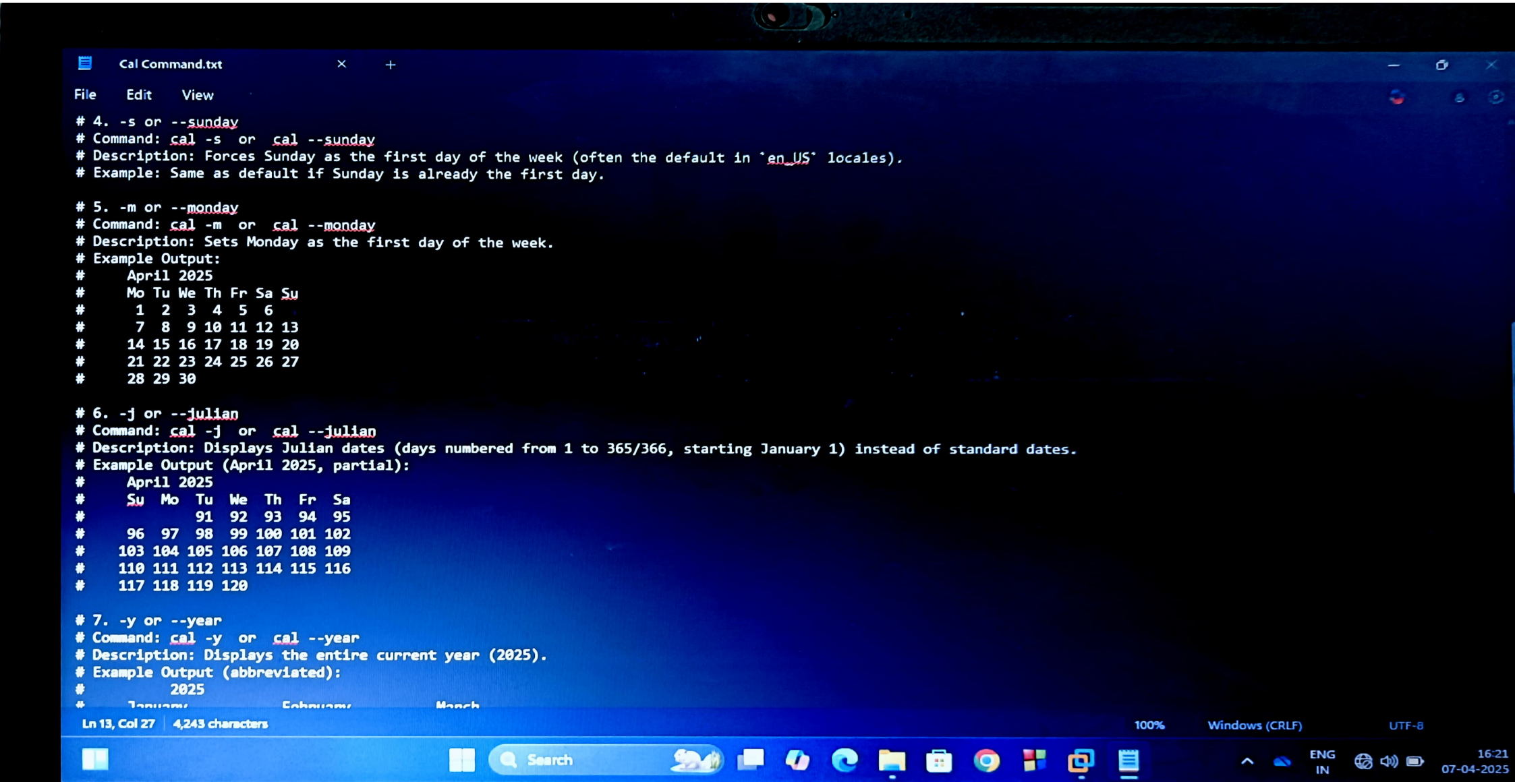
ENG
IN

```
#      June 2025
#      Su Mo Tu We Th Fr Sa
#       1  2  3  4  5  6  7
#       8  9 10 11 12 13 14
#      15 16 17 18 19 20 21
#      22 23 24 25 26 27 28
#      29 30

# 9. Specify Year Only
# Command: cal [year]  (e.g., cal 2026)
# Description: Displays the entire calendar for the specified year (e.g., 2026).
# Example: Similar to `-y` but for any year.

# 10. -h or --help
# Command: cal -h  or  cal --help
# Description: Displays the help message listing all available options.
# Example Output: (Varies by version, typically lists options as above.)

# 11. -V or --version
# Command: cal -V  or  cal --version
# Description: Shows the version of the `cal` command installed on CentOS Stream 10.
# Example Output:
#      cal (GNU coreutils) 8.32

## Additional Notes for CentOS Stream 10
# - Installation Check: Ensure `cal` is available with `which cal` (typically `/usr/bin/cal`). If missing, install via `sudo dnf install util-linux`.
# - Locale Impact: The first day of the week depends on your locale (e.g., `en_US` uses Sunday, `en_GB` uses Monday). Modify with `export LC_TIME="en_GB.UTF-8"` if
needed.
# - Historical Calendars: For dates like September 1752 (`cal 9 1752`), `cal` reflects the Gregorian switch, skipping 11 days.
```

Ln 13, Col 27 | 4,243 characters | 100% | Windows (CRLF) | UTF-8

Search | ENG IN | 16:21 07-04-2025


Cal Command.txt
File Edit View
January February March
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
1 2 3 4 1 2
5 6 7 8 9 10 11 2 3 4 5 6 7 8 3 4 5 6 7 8 9
...
8. Specify Month and Year
Command: cal [month] [year] (e.g., cal 6 2025)
Description: Displays the calendar for a specific month and year (e.g., June 2025).
Example Output:
June 2025
Su Mo Tu We Th Fr Sa
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30
9. Specify Year Only
Command: cal [year] (e.g., cal 2026)
Description: Displays the entire calendar for the specified year (e.g., 2026).
Example: Similar to `-y` but for any year.
10. -h or --help
Command: cal -h or cal --help
Description: Displays the help message listing all available options.
Example Output: (Varies by version, typically lists options as above.)
11. -V or --version
Command: cal -V or cal --version
Description: Shows the version of the `cal` command installed on CentOS Stream 10.
Example Output:
cal (GNU coreutils) 8.32
Additional Notes for CentOS Stream 10
Ln 13, Col 27 4,243 characters
100% Windows (CRLF) UTF-8
Search
ENG IN
16:21
07-04-2025


Cal Command.txt
File Edit View
4. -s or --sunday
Command: cal -s or cal --sunday
Description: Forces Sunday as the first day of the week (often the default in `en_US` locales).
Example: Same as default if Sunday is already the first day.
5. -m or --monday
Command: cal -m or cal --monday
Description: Sets Monday as the first day of the week.
Example Output:
April 2025
Mo Tu We Th Fr Sa Su
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
6. -j or --julian
Command: cal -j or cal --julian
Description: Displays Julian dates (days numbered from 1 to 365/366, starting January 1) instead of standard dates.
Example Output (April 2025, partial):
April 2025
Su Mo Tu We Th Fr Sa
91 92 93 94 95
96 97 98 99 100 101 102
103 104 105 106 107 108 109
110 111 112 113 114 115 116
117 118 119 120
7. -y or --year
Command: cal -y or cal --year
Description: Displays the entire current year (2025).
Example Output (abbreviated):
2025
Ln 13, Col 27 | 4,243 characters
100%
Windows (CRLF)
UTF-8
Search
ENG
IN
16:21
07-04-2025

Cal Command.txt

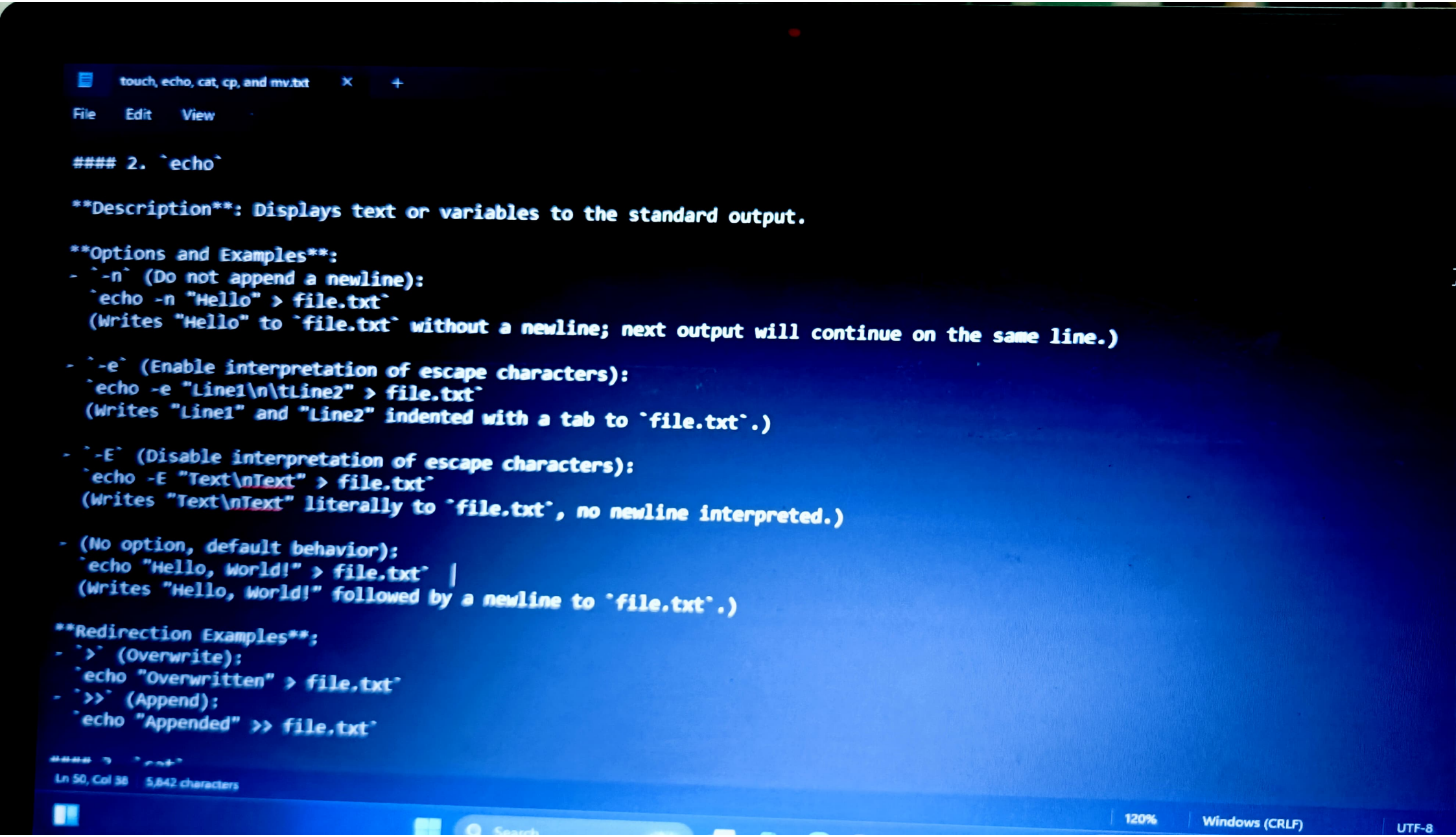
File Edit View

```
### `cal` Command Overview
# The `cal` command displays a Gregorian calendar in the terminal. By default, it shows the current month (April 2025, given the date).
# Syntax: cal [options] [[[day] month] year]
# Part of the `util-linux` package in CentOS Stream 10. Install with: sudo dnf install util-linux

### Options, Variants, and Descriptions

# 1. No Options (Default)
# Command: cal
# Description: Displays the calendar for the current month (April 2025) with the current day (1) highlighted.
# Example Output:
#     April 2025
#     Su Mo Tu We Th Fr Sa
#           1  2  3  4  5
#      6  7  8  9 10 11 12
#     13 14 15 16 17 18 19
#     20 21 22 23 24 25 26
#     27 28 29 30

# 2. -1 or --one
# Command: cal -1  or  cal --one
# Description: Explicitly displays a single month (same as default behavior).
# Example: Same as above.

# 3. -3 or --three
# Command: cal -3  or  cal --three
# Description: Displays the previous, current, and next months side by side (March, April, May 2025).
# Example Output:
#       March 2025         April 2025          May 2025
#     Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
#                           1  2  3  4  5              1  2  3
#      2  3  4  5  6  7  8  6  7  8  9 10 11 12  4  5  6  7  8  9 10
#      9 10 11 12 13 14 15 13 14 15 16 17 18 19 11 12 13 14 15 16 17
#     16 17 18 19 20 21 22 20 21 22 23 24 25 26 18 19 20 21 22 23 24
```

Ln 13, Col 27 | 4,243 characters | 100% | Windows (CRLF) | UTF-8

Search

ENG IN


touch, echo, cat, cp, and mv.txt
File Edit View
2. `echo`
Description: Displays text or variables to the standard output.
Options and Examples:
- `-n` (Do not append a newline):
`echo -n "Hello" > file.txt`
(Writes "Hello" to `file.txt` without a newline; next output will continue on the same line.)
- `-e` (Enable interpretation of escape characters):
`echo -e "Line1\n\tLine2" > file.txt`
(Writes "Line1" and "Line2" indented with a tab to `file.txt`.)
- `-E` (Disable interpretation of escape characters):
`echo -E "Text\nText" > file.txt`
(Writes "Text\nText" literally to `file.txt`, no newline interpreted.)
- (No option, default behavior):
`echo "Hello, World!" > file.txt`
(Writes "Hello, World!" followed by a newline to `file.txt`.)
Redirection Examples:
- `>` (Overwrite):
`echo "Overwritten" > file.txt`
- `>>` (Append):
`echo "Appended" >> file.txt`
Ln 50, Col 38 5,842 characters
120% Windows (CRLF) UTF-8

touch, echo, cat, cp, and mv.txt

File Edit View

1. `touch`

Description: Creates an empty file or updates the timestamp of an existing file.

Options and Examples:
- `-a` (Change only the access time):
  `touch -a file.txt`
  (Updates the last access time of `file.txt` without changing modification time.)

- `-m` (Change only the modification time):
  `touch -m file.txt`
  (Updates the last modification time of `file.txt` without changing access time.)

- `-r <file>` (Use the timestamp of the specified file):
  `touch -r reference.txt file.txt`
  (Sets `file.txt`'s timestamp to match `reference.txt`.)

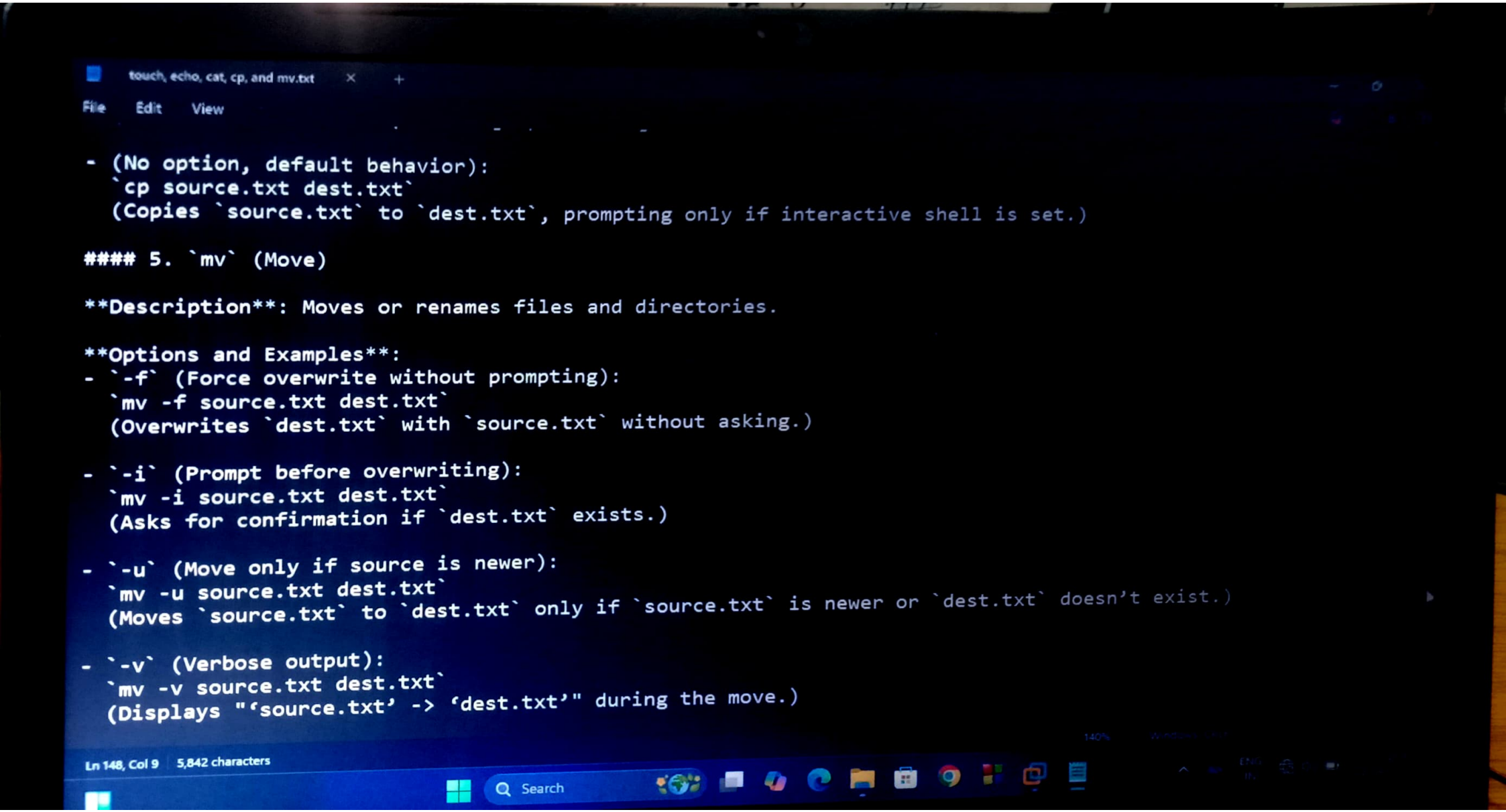
- `-t <timestamp>` (Set a specific timestamp, format: `[[CC]YY]MMDDhhmm[.ss]`):
  `touch -t 202301151030.45 file.txt`
  (Sets `file.txt`'s timestamp to Jan 15, 2023, 10:30:45 AM.)

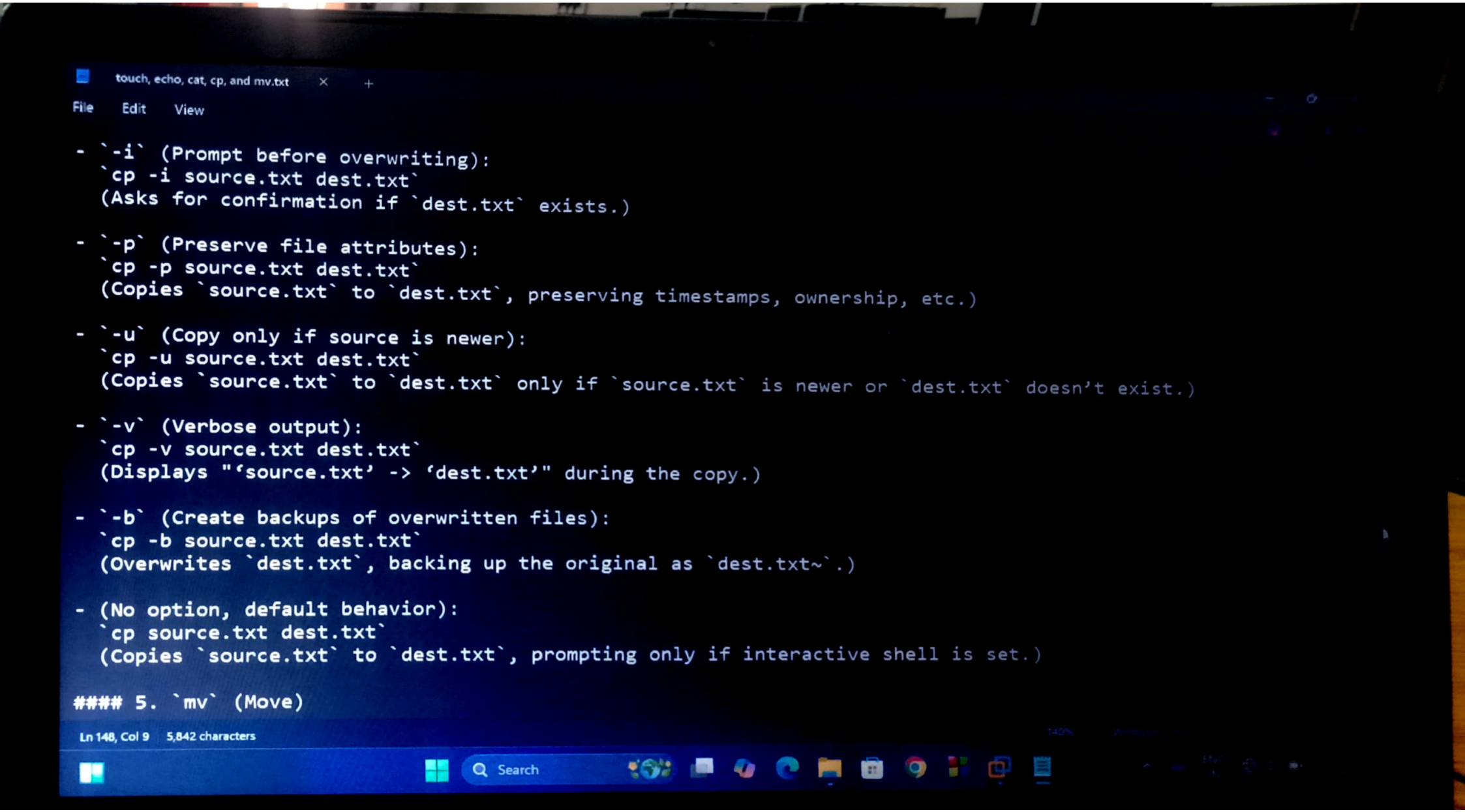
- `--no-create` (Do not create the file if it doesn't exist):
  `touch --no-create missing.txt`
  (Does nothing if `missing.txt` doesn't exist; no file is created.)

- (No option, default behavior):
  `touch newfile.txt`
  (Creates `newfile.txt` if it doesn't exist or updates its timestamp.)

Ln 50, Col 38 5,842 characters

120% Windows (CRLF)


touch, echo, cat, cp, and mv.txt
File Edit View
- (No option, default behavior):
`cp source.txt dest.txt`
(Copies `source.txt` to `dest.txt`, prompting only if interactive shell is set.)
5. `mv` (Move)
Description: Moves or renames files and directories.
Options and Examples:
- `-f` (Force overwrite without prompting):
`mv -f source.txt dest.txt`
(Overwrites `dest.txt` with `source.txt` without asking.)
- `-i` (Prompt before overwriting):
`mv -i source.txt dest.txt`
(Asks for confirmation if `dest.txt` exists.)
- `-u` (Move only if source is newer):
`mv -u source.txt dest.txt`
(Moves `source.txt` to `dest.txt` only if `source.txt` is newer or `dest.txt` doesn't exist.)
- `-v` (Verbose output):
`mv -v source.txt dest.txt`
(Displays "'source.txt' -> 'dest.txt'" during the move.)
Ln 148, Col 9 5,842 characters
Search


touch, echo, cat, cp, and mv.txt
File
Edit
View
- `-i` (Prompt before overwriting):
`cp -i source.txt dest.txt`
(Asks for confirmation if `dest.txt` exists.)
- `-p` (Preserve file attributes):
`cp -p source.txt dest.txt`
(Copies `source.txt` to `dest.txt`, preserving timestamps, ownership, etc.)
- `-u` (Copy only if source is newer):
`cp -u source.txt dest.txt`
(Copies `source.txt` to `dest.txt` only if `source.txt` is newer or `dest.txt` doesn't exist.)
- `-v` (Verbose output):
`cp -v source.txt dest.txt`
(Displays "'source.txt' -> 'dest.txt'" during the copy.)
- `-b` (Create backups of overwritten files):
`cp -b source.txt dest.txt`
(Overwrites `dest.txt`, backing up the original as `dest.txt~`.)
- (No option, default behavior):
`cp source.txt dest.txt`
(Copies `source.txt` to `dest.txt`, prompting only if interactive shell is set.)
5. `mv` (Move)
Ln 148, Col 9
5,842 characters
Search

touch, echo, cat, cp, and mv.txt

File Edit View

```
**Redirection Examples**:
- `>` (Create/overwrite with input):
  `cat > newfile.txt`
  `Type some text here.`
  `[Press Ctrl+D]`

- `>>` (Append to file):
  `cat file.txt >> existing.txt`

#### 4. `cp` (Copy)

**Description**: Copies files or directories from one location to another.

**Options and Examples**:
- `-r` or `-R` (Copy directories recursively):
  `cp -r source_dir/ dest_dir/`
  (Copies `source_dir` and all its contents to `dest_dir`.)

- `-f` (Force overwrite without prompting):
  `cp -f source.txt dest.txt`
  (Overwrites `dest.txt` with `source.txt` without asking.)

- `-i` (Prompt before overwriting):
  `cp -i source.txt dest.txt`
  (Asks for confirmation if `dest.txt` exists.)
```

Ln 148, Col 9 | 5,842 characters

140%

Search

touch, echo, cat, cp, and mv.txt

File Edit View

3. `cat`

Description: Concatenates and displays file contents, or creates files with input redirection.

Options and Examples:
- `-n` (Number all output lines):
  `cat -n file.txt > numbered.txt`
  (Writes contents of `file.txt` to `numbered.txt` with line numbers.)

- `-b` (Number non-blank output lines):
  `cat -b file.txt > numbered.txt`
  (Numbers only non-empty lines from `file.txt` in `numbered.txt`.)

- `-s` (Squeeze multiple blank lines into one):
  `cat -s file_with_blanks.txt > squeezed.txt`
  (Reduces multiple consecutive blank lines in `file_with_blanks.txt` to one in `squeezed.txt`.)

- `-A` (Show all non-printing characters):
  `cat -A file.txt > visible.txt`
  (Writes `file.txt` contents to `visible.txt`, showing tabs as `^I`, line ends as `$`, etc.)

- (No option, default behavior):
  `cat file1.txt file2.txt > combined.txt`
  (Concatenates `file1.txt` and `file2.txt` into `combined.txt`.)

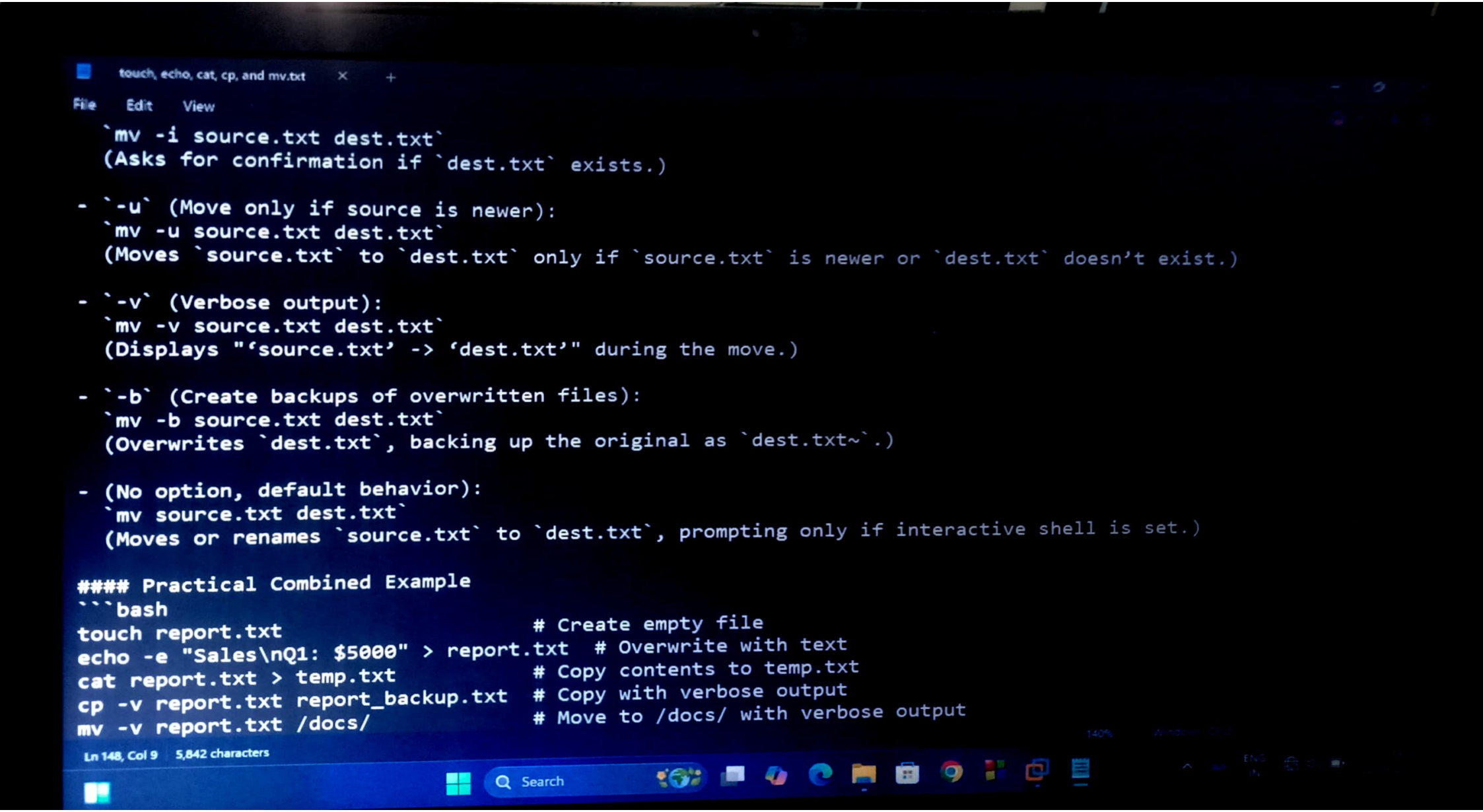
Ln 148, Col 9 | 5,842 characters

140% Windows (CRLF) UTF-8

Search

ENG IN

16:44 22-04-2025


touch, echo, cat, cp, and mv.txt
File Edit View
`mv -i source.txt dest.txt`
(Asks for confirmation if `dest.txt` exists.)
- `-u` (Move only if source is newer):
`mv -u source.txt dest.txt`
(Moves `source.txt` to `dest.txt` only if `source.txt` is newer or `dest.txt` doesn't exist.)
- `-v` (Verbose output):
`mv -v source.txt dest.txt`
(Displays "'source.txt' -> 'dest.txt'" during the move.)
- `-b` (Create backups of overwritten files):
`mv -b source.txt dest.txt`
(Overwrites `dest.txt`, backing up the original as `dest.txt~`.)
- (No option, default behavior):
`mv source.txt dest.txt`
(Moves or renames `source.txt` to `dest.txt`, prompting only if interactive shell is set.)
Practical Combined Example
```bash
touch report.txt # Create empty file
echo -e "Sales\nQ1: $5000" > report.txt # Overwrite with text
cat report.txt > temp.txt # Copy contents to temp.txt
cp -v report.txt report_backup.txt # Copy with verbose output
mv -v report.txt /docs/ # Move to /docs/ with verbose output
Ln 148, Col 9 5,842 characters
Search
```

less and more.txt

File Edit View

1. less Command
What it does: Displays a file one screen at a time, allowing scrolling up/down, searching, and customization. More powerful than more.

Common Options and Examples:
- -E (Exit at end): less -E test.txt
  (Press Space to scroll down; exits after "Last line of the file." Press q to quit early.)

- -F (Exit if fits on screen): less -F test.txt
  (If 10 lines fit your screen, exits immediately; otherwise, press Space to scroll, q to quit.)

- -g (Highlight last search match): less -g test.txt
  (Type /Tab, press Enter, only last "Tab" highlights. Press n for next match, q to quit.)

- -G (No highlighting): less -G test.txt
  (Type /Tab, press Enter, cursor moves to "Tab" but no highlight. Press q to quit.)

- -i (Case-insensitive search): less -i test.txt
  (Type /line, press Enter, matches "line" and "LINE". Press q to quit.)

- -I (Strong case-insensitive search): less -I test.txt
  (Type /LINE, press Enter, matches "line" and "LINE". Press q to quit.)

- -m (Verbose prompt): less -m test.txt
  (Bottom shows "test.txt 50%" when halfway. Press q to quit.)

- -M (More verbose prompt): less -M test.txt
  (Bottom shows "lines 1-5/10 50%". Press q to quit.)

Ln 64, Col 71 30 of 4,523 characters 130% Windows (CRLF) UTF-8

Search ENG IN 18:42 23-04-2025

less and more.txt

File Edit View

1. less Command
What it does: Displays a file one screen at a time, allowing scrolling up/down, searching, and customization. More
more.

Common Options and Examples:
- -E (Exit at end): less -E test.txt
  (Press Space to scroll down; exits after "Last line of the file." Press q to quit early.)

- -F (Exit if fits on screen): less -F test.txt
  (If 10 lines fit your screen, exits immediately; otherwise, press Space to scroll, q to quit.)

- -g (Highlight last search match): less -g test.txt
  (Type /Tab, press Enter, only last "Tab" highlights. Press n for next match, q to quit.)

- -G (No highlighting): less -G test.txt
  (Type /Tab, press Enter, cursor moves to "Tab" but no highlight. Press q to quit.)

- -i (Case-insensitive search): less -i test.txt
  (Type /line, press Enter, matches "line" and "LINE". Press q to quit.)

- -I (Strong case-insensitive search): less -I test.txt
  (Type /LINE, press Enter, matches "line" and "LINE". Press q to quit.)

- -m (Verbose prompt): less -m test.txt
  (Bottom shows "test.txt 50%" when halfway. Press q to quit.)

- -M (More verbose prompt): less -M test.txt
  (Bottom shows "lines 1-5/10 50%". Press q to quit.)

Ln 64, Col 71 93 of 4,513 characters 100% Windows (CRLF)

Search

less and more.txt

File Edit View

- -n (No line numbers): less -n test.txt
  (No numbers even if -N set elsewhere. Press q to quit.)

- -N (Show line numbers): less -N test.txt
  (Shows "1 This is line 1...", "2 Here is LINE 2...". Press q to quit.)

- -s (Squeeze blank lines): less -s test.txt
  (Three blank lines become one. Press q to quit.)

- -S (Chop long lines): less -S test.txt
  ("Lorem ipsum..." truncates; use right/left arrows to scroll horizontally, q to quit.)

- -x 4 (Set tab stops to 4 spaces): less -x4 test.txt
  (Tabs in "Tab1 Tab2" are 4 spaces. Press q to quit.)

- -z 10 (Set window size to 10 lines): less -z10 test.txt
  (Press Space to scroll 10 lines, q to quit.)

- +G (Start at end): less +G test.txt
  (Starts at "Last line...". Press b to scroll up, q to quit.)

- (Default): less test.txt
  (Press Space to scroll down, b to scroll up, /text to search, q to quit.)

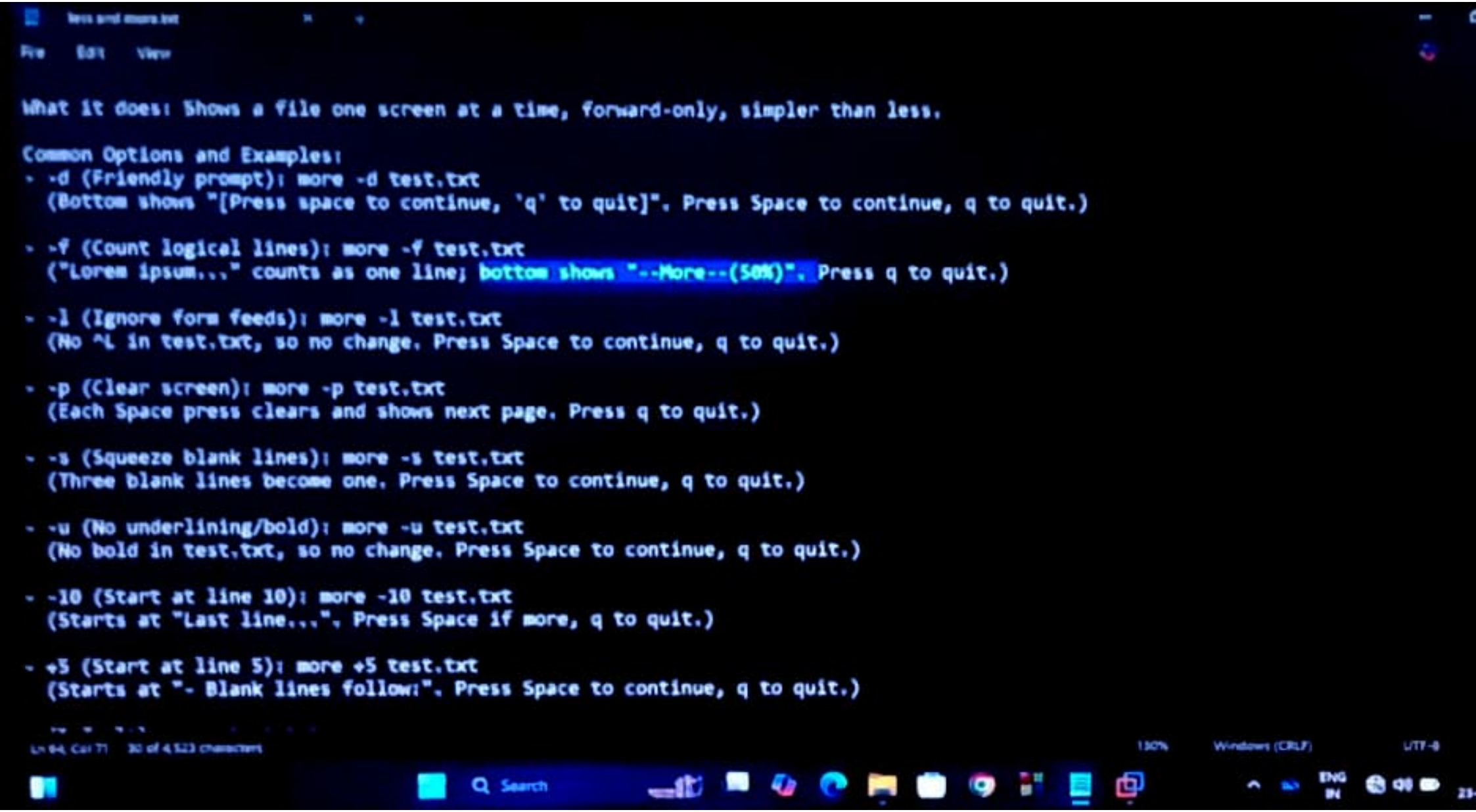
Practical Use: View large logs, search text, navigate files flexibly.

2. more Command

Ln 64, Col 71 30 of 4,523 characters 130% Windows (CRLF) UTF-8

Search

ENG IN 18:42 23-04-2025


File Edit View
What it does: Shows a file one screen at a time, forward-only, simpler than less.
Common Options and Examples:
- -d (Friendly prompt): more -d test.txt
(Bottom shows "[Press space to continue, 'q' to quit]". Press Space to continue, q to quit.)
- -f (Count logical lines): more -f test.txt
("Lorem ipsum..." counts as one line; bottom shows "--More--(50%)". Press q to quit.)
- -l (Ignore form feeds): more -l test.txt
(No ^L in test.txt, so no change. Press Space to continue, q to quit.)
- -p (Clear screen): more -p test.txt
(Each Space press clears and shows next page. Press q to quit.)
- -s (Squeeze blank lines): more -s test.txt
(Three blank lines become one. Press Space to continue, q to quit.)
- -u (No underlining/bold): more -u test.txt
(No bold in test.txt, so no change. Press Space to continue, q to quit.)
- -10 (Start at line 10): more -10 test.txt
(Starts at "Last line...". Press Space if more, q to quit.)
- +5 (Start at line 5): more +5 test.txt
(Starts at "- Blank lines follow:". Press Space to continue, q to quit.)
Search

```
File  Edit  View
  (Starts at "- Blank lines follow:". Press Space to continue, q to quit.)

- (Default): more test.txt
  (Press Space to move forward, see "--More--", press q to quit.)


Practical Use: Quick view of short files when scrolling back isn't needed.

#### Practical Combined Example Explained
echo -e "Line1\n\nLine2\nLong line here that wraps around\nLine4" > sample.txt
less -N -s sample.txt    # View with line numbers and squeezed blank lines
more -s -3 sample.txt    # View from line 3 with squeezed blank lines

Breakdown:
1. echo -e "Line1\n\nLine2\nLong line here that wraps around\nLine4" > sample.txt
   (Creates sample.txt with 5 lines: "Line1", blank, "Line2", long line, "Line4".)
2. less -N -s sample.txt
   (-N: Adds line numbers. -s: Squeezes two blank lines into one. Press Space to scroll, q to quit.)
3. more -s -3 sample.txt
   (-s: Squeezes blank lines. -3: Starts at "Line2". Press Space to continue, q to quit.)

With test.txt:
less -N -s test.txt    # Shows test.txt with line numbers, squeezed blanks (3 blanks -> 1). Press q to quit.
more -s -3 test.txt    # Starts at "- Tabs here: Tab1 Tab2", squeezed blanks. Press Space, then q to quit.

#### Key Differences
- less: Scroll up/down (Space, b), search (/text), quit (q). More flexible.
- more: Only forward (Space), no backtracking, simpler, quit (q).
```

less and more.txt touch, echo, cat, cp, and mv.txt

File Edit View

Comprehensive Examples of Linux Commands in CentOS: touch, echo, cat, cp, mv

1. `touch`

Description: Creates an empty file or updates the timestamp of an existing file.

Options and Examples:
- `-a` (Change only the access time):
  `touch -a file.txt`
  (Updates the last access time of `file.txt` without changing modification time.)

- `-m` (Change only the modification time):
  `touch -m file.txt`
  (Updates the last modification time of `file.txt` without changing access time.)

- `-r <file>` (Use the timestamp of the specified file):
  `touch -r reference.txt file.txt`
  (Sets `file.txt`'s timestamp to match `reference.txt`.)

- `-t <timestamp>` (Set a specific timestamp, format: `[[CC]YY]MMDDhhmm[.ss]`):
  `touch -t 202301151030.45 file.txt`
  (Sets `file.txt`'s timestamp to Jan 15, 2023, 10:30:45 AM.)

- `--no-create` (Do not create the file if it doesn't exist):
  `touch --no-create missing.txt`
  (Does nothing if `missing.txt` doesn't exist; no file is created.)

- (No option, default behavior):
  `touch newfile.txt`
  (Creates `newfile.txt` if it doesn't exist or updates its timestamp.)

2. `echo`

Description: Displays text or variables to the standard output.

Ln 1, Col 1 5,842 characters 100% Windows (CRLF)

Search ENG

less and more.txt touch, echo, cat, cp, and mv.txt

File Edit View

2. `echo`

Description: Displays text or variables to the standard output.

Options and Examples:
- `-n` (Do not append a newline):
  `echo -n "Hello" > file.txt`
  (Writes "Hello" to `file.txt` without a newline; next output will continue on the same line.)

- `-e` (Enable interpretation of escape characters):
  `echo -e "Line1\n\tLine2" > file.txt`
  (Writes "Line1" and "Line2" indented with a tab to `file.txt`.)

- `-E` (Disable interpretation of escape characters):
  `echo -E "Text\nText" > file.txt`
  (Writes "Text\nText" literally to `file.txt`, no newline interpreted.)

- (No option, default behavior):
  `echo "Hello, World!" > file.txt`
  (Writes "Hello, World!" followed by a newline to `file.txt`.)

Redirection Examples:
- `>` (Overwrite):
  `echo "Overwritten" > file.txt`
- `>>` (Append):
  `echo "Appended" >> file.txt`

3. `cat`

Description: Concatenates and displays file contents, or creates files with input redirection.

Options and Examples:
- `-n` (Number all output lines):
  `cat -n file.txt > numbered.txt`

Ln 1, Col 1 3,842 characters 100% Windows (CRLF)

less and more.txt touch, echo, cat, cp, and mv.txt

File Edit View

4. `cp` (Copy)

Description: Copies files or directories from one location to another.

Options and Examples:
- `-r` or `-R` (Copy directories recursively):
  `cp -r source_dir/ dest_dir/`
  (Copies `source_dir` and all its contents to `dest_dir`.)

- `-f` (Force overwrite without prompting):
  `cp -f source.txt dest.txt`
  (Overwrites `dest.txt` with `source.txt` without asking.)

- `-i` (Prompt before overwriting):
  `cp -i source.txt dest.txt`
  (Asks for confirmation if `dest.txt` exists.)

- `-p` (Preserve file attributes):
  `cp -p source.txt dest.txt`
  (Copies `source.txt` to `dest.txt`, preserving timestamps, ownership, etc.)

- `-u` (Copy only if source is newer):
  `cp -u source.txt dest.txt`
  (Copies `source.txt` to `dest.txt` only if `source.txt` is newer or `dest.txt` doesn't exist.)

- `-v` (Verbose output):
  `cp -v source.txt dest.txt`
  (Displays "'source.txt' -> 'dest.txt'" during the copy.)

- `-b` (Create backups of overwritten files):
  `cp -b source.txt dest.txt`
  (Overwrites `dest.txt`, backing up the original as `dest.txt~`.)

- (No option, default behavior):

Ln 1, Col 1 3,842 characters

100% Windows (CRL

Search

less and more.txt touch, echo, cat, cp, and mv.txt

File Edit View

3. `cat`

Description: Concatenates and displays file contents, or creates files with input redirection.

Options and Examples:
- `-n` (Number all output lines):
  `cat -n file.txt > numbered.txt`
  (Writes contents of `file.txt` to `numbered.txt` with line numbers.)

- `-b` (Number non-blank output lines):
  `cat -b file.txt > numbered.txt`
  (Numbers only non-empty lines from `file.txt` in `numbered.txt`.)

- `-s` (Squeeze multiple blank lines into one):
  `cat -s file_with_blanks.txt > squeezed.txt`
  (Reduces multiple consecutive blank lines in `file_with_blanks.txt` to one in `squeezed.txt`.)

- `-A` (Show all non-printing characters):
  `cat -A file.txt > visible.txt`
  (Writes `file.txt` contents to `visible.txt`, showing tabs as `^I`, line ends as `$`, etc.)

- (No option, default behavior):
  `cat file1.txt file2.txt > combined.txt`
  (Concatenates `file1.txt` and `file2.txt` into `combined.txt`.)

Redirection Examples:
- `>` (Create/overwrite with input):
  `cat > newfile.txt`
  `Type some text here.`
  `[Press Ctrl+D]`

- `>>` (Append to file):
  `cat file.txt >> existing.txt`

Ln 1, Col 1 5,842 characters 100% Windows (CRLF)

less and more.txt touch, echo, cat, cp, and mv.txt

File Edit View

- (No option, default behavior):
  `cp source.txt dest.txt`
  (Copies `source.txt` to `dest.txt`, prompting only if interactive shell is set.)

5. `mv` (Move)

Description: Moves or renames files and directories.

Options and Examples:
- `-f` (Force overwrite without prompting):
  `mv -f source.txt dest.txt`
  (Overwrites `dest.txt` with `source.txt` without asking.)

- `-i` (Prompt before overwriting):
  `mv -i source.txt dest.txt`
  (Asks for confirmation if `dest.txt` exists.)

- `-u` (Move only if source is newer):
  `mv -u source.txt dest.txt`
  (Moves `source.txt` to `dest.txt` only if `source.txt` is newer or `dest.txt` doesn't exist.)

- `-v` (Verbose output):
  `mv -v source.txt dest.txt`
  (Displays "'source.txt' -> 'dest.txt'" during the move.)

- `-b` (Create backups of overwritten files):
  `mv -b source.txt dest.txt`
  (Overwrites `dest.txt`, backing up the original as `dest.txt~`.)

- (No option, default behavior):
  `mv source.txt dest.txt`
  (Moves or renames `source.txt` to `dest.txt`, prompting only if interactive shell is set.)

Practical Combined Example

Ln 1, Col 1 3,842 characters 100% Windows (CRLF)

Search

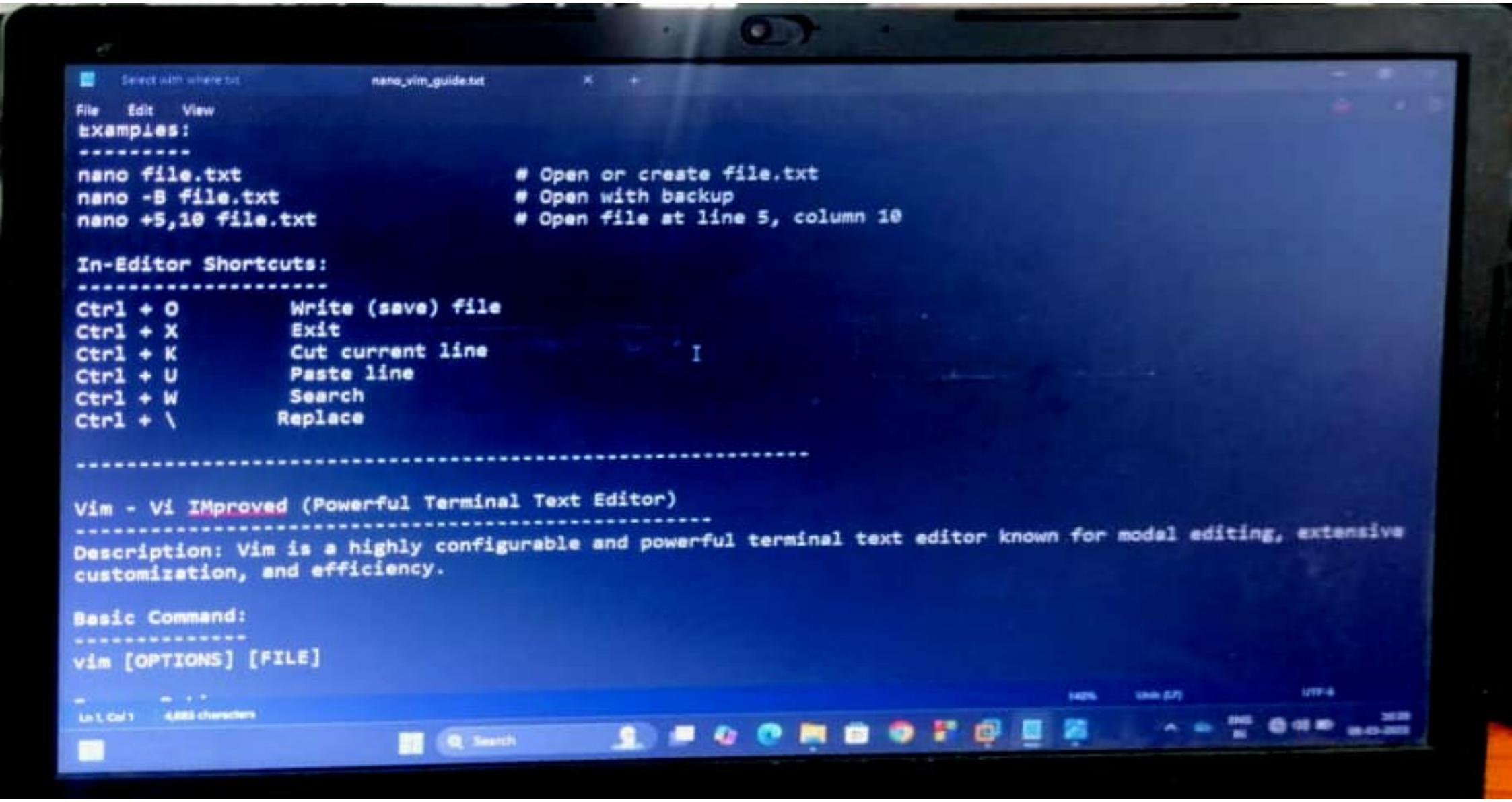
less and more.txt touch, echo, cat, cp, and mv.txt

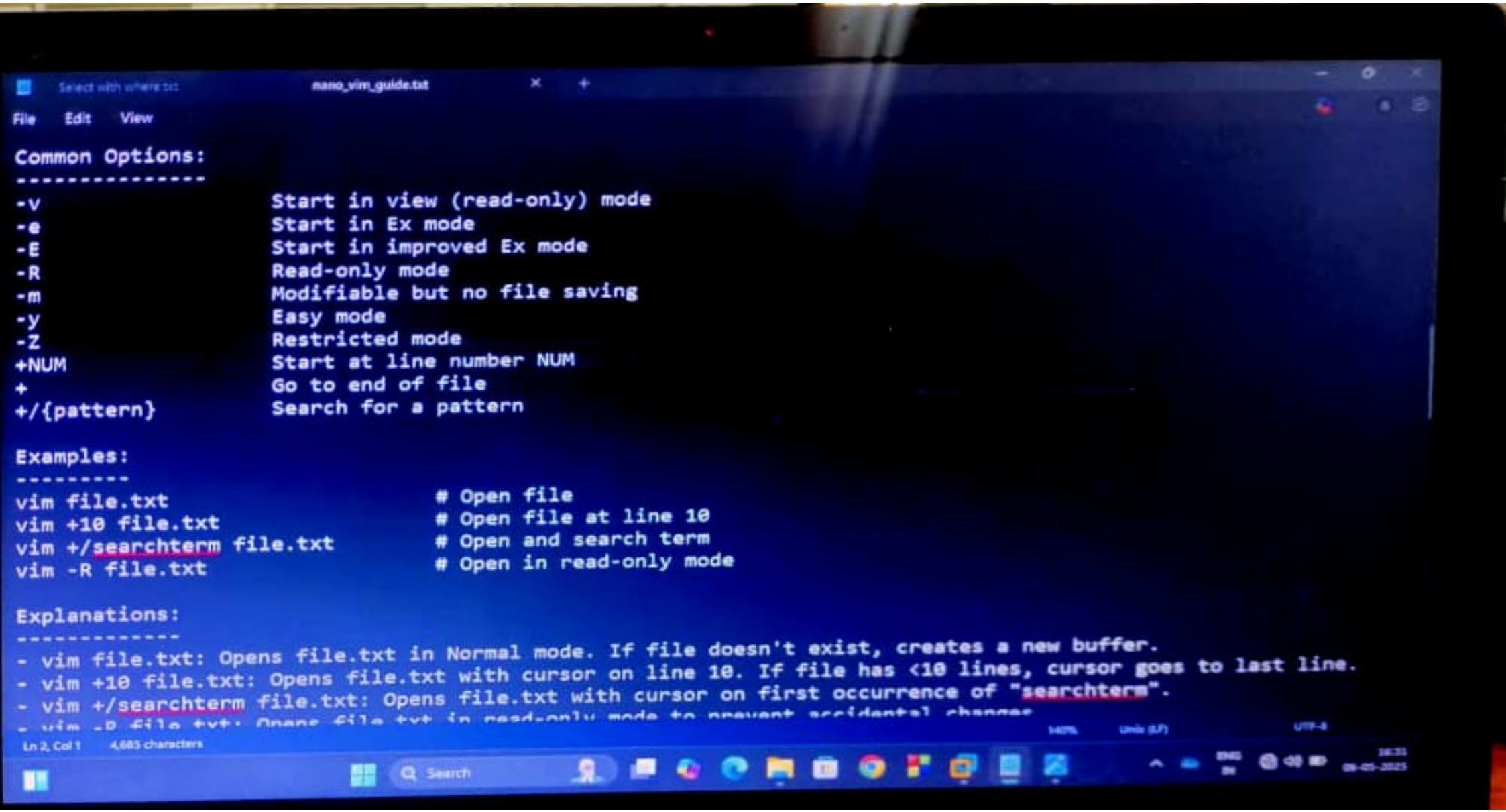
File Edit View

```
#### 3. `cat`

**Description**: Concatenates and displays file contents, or creates files with input redirection.

**Options and Examples**:
- `-n` (Number all output lines):
  `cat -n file.txt > numbered.txt`
  (Writes contents of `file.txt` to `numbered.txt` with line numbers.)

- `-b` (Number non-blank output lines):
  `cat -b file.txt > numbered.txt`
  (Numbers only non-empty lines from `file.txt` in `numbered.txt`.)

- `-s` (Squeeze multiple blank lines into one):
  `cat -s file_with_blanks.txt > squeezed.txt`
  (Reduces multiple consecutive blank lines in `file_with_blanks.txt` to one in `squeezed.txt`.)

- `-A` (Show all non-printing characters):
  `cat -A file.txt > visible.txt`
  (Writes `file.txt` contents to `visible.txt`, showing tabs as `^I`, line ends as `$`, etc.)

- (No option, default behavior):
  `cat file1.txt file2.txt > combined.txt`
  (Concatenates `file1.txt` and `file2.txt` into `combined.txt`.)

**Redirection Examples**:
- `>` (Create/overwrite with input):
  `cat > newfile.txt`
  `Type some text here.`
  `[Press Ctrl+D]`

- `>>` (Append to file):
  `cat file.txt >> existing.txt`
```

Ln 1, Col 1 5,842 characters 100% Windows (CRLF)

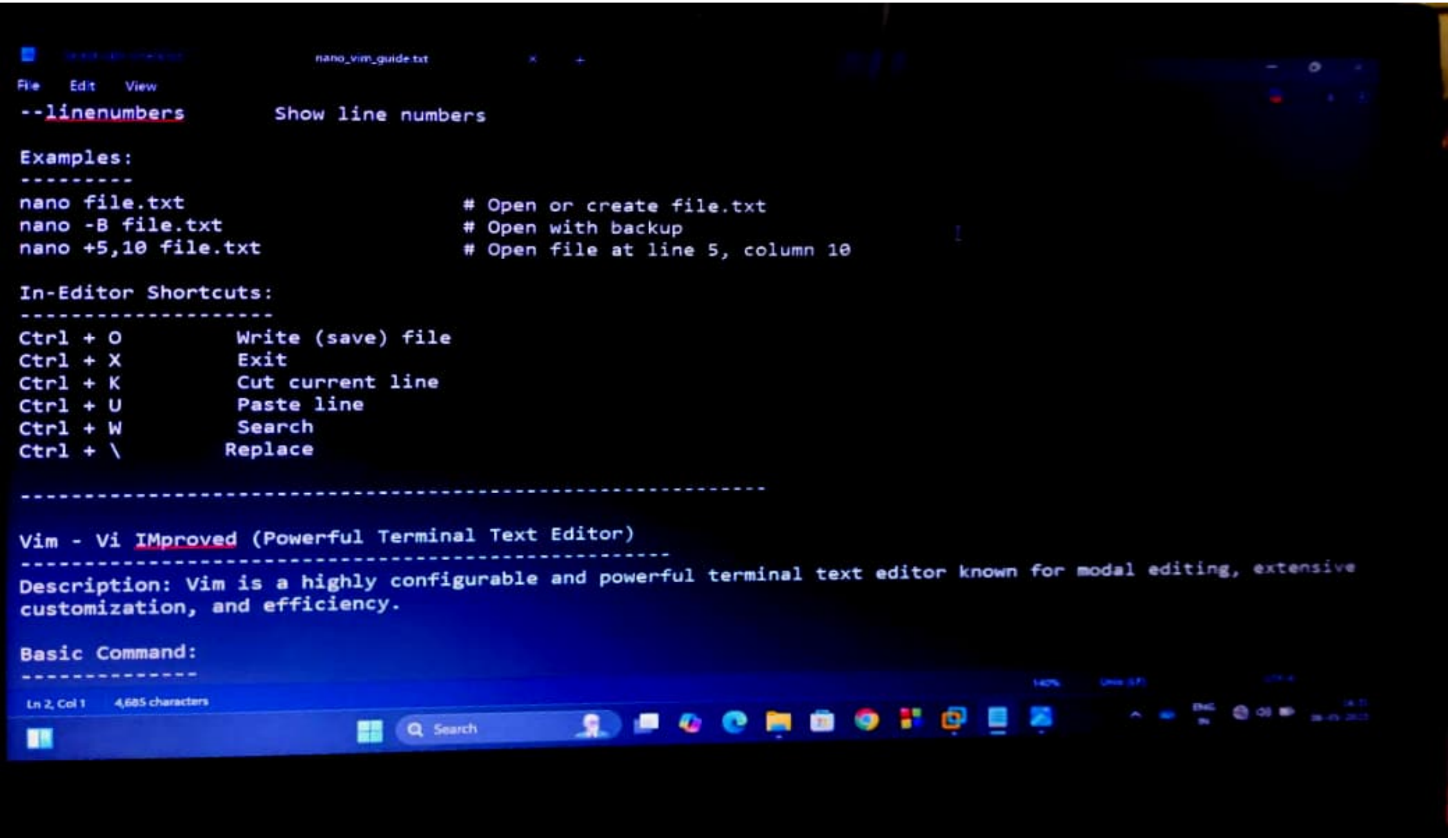
Search ENG IN

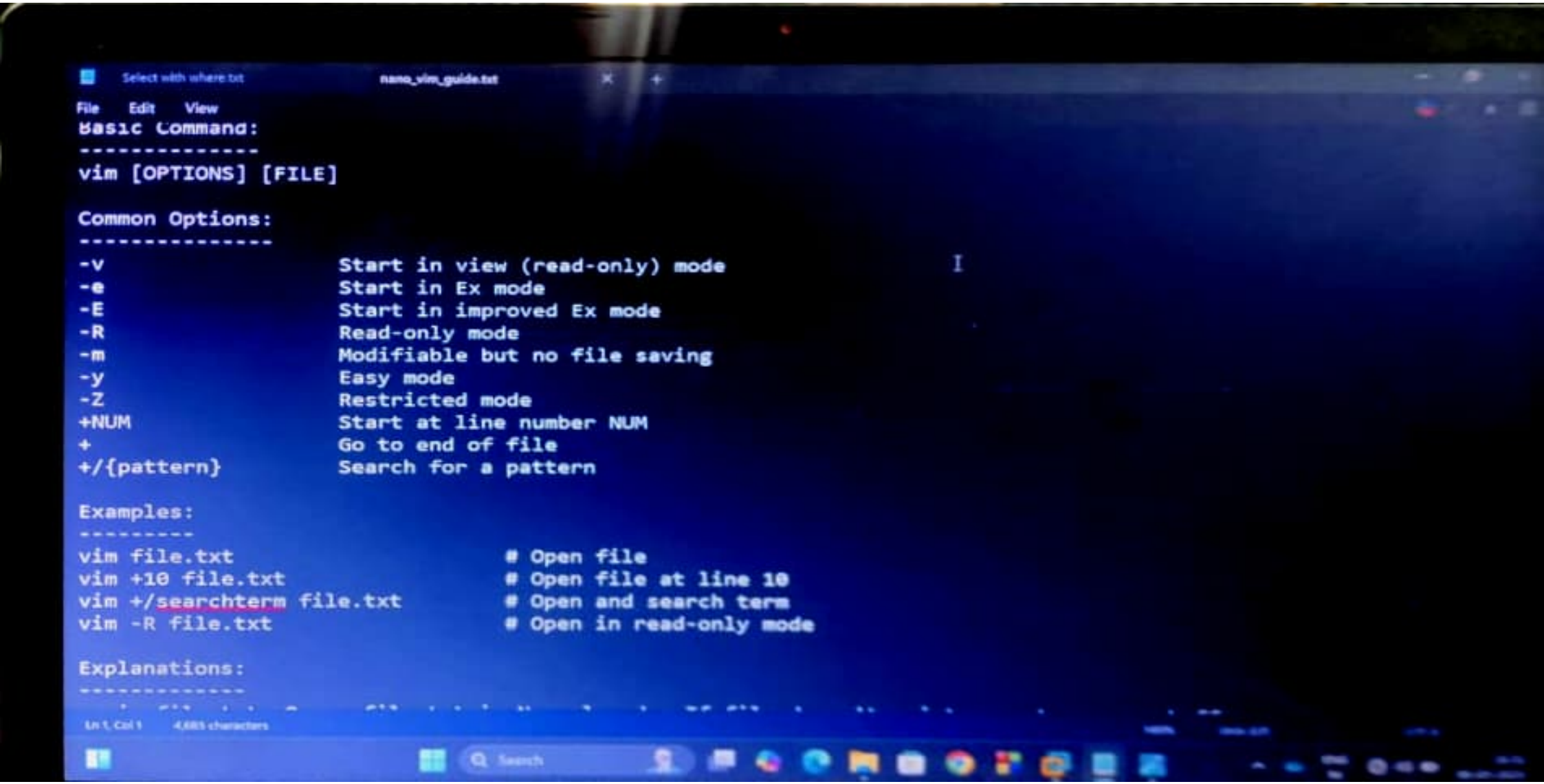

nano_vim_guide.txt
File Edit View
Examples:
nano file.txt # Open or create file.txt
nano -B file.txt # Open with backup
nano +5,10 file.txt # Open file at line 5, column 10
In-Editor Shortcuts:
Ctrl + O Write (save) file
Ctrl + X Exit
Ctrl + K Cut current line
Ctrl + U Paste line
Ctrl + W Search
Ctrl + \ Replace
Vim - Vi IMproved (Powerful Terminal Text Editor)
Description: Vim is a highly configurable and powerful terminal text editor known for modal editing, extensive customization, and efficiency.
Basic Command:
vim [OPTIONS] [FILE]

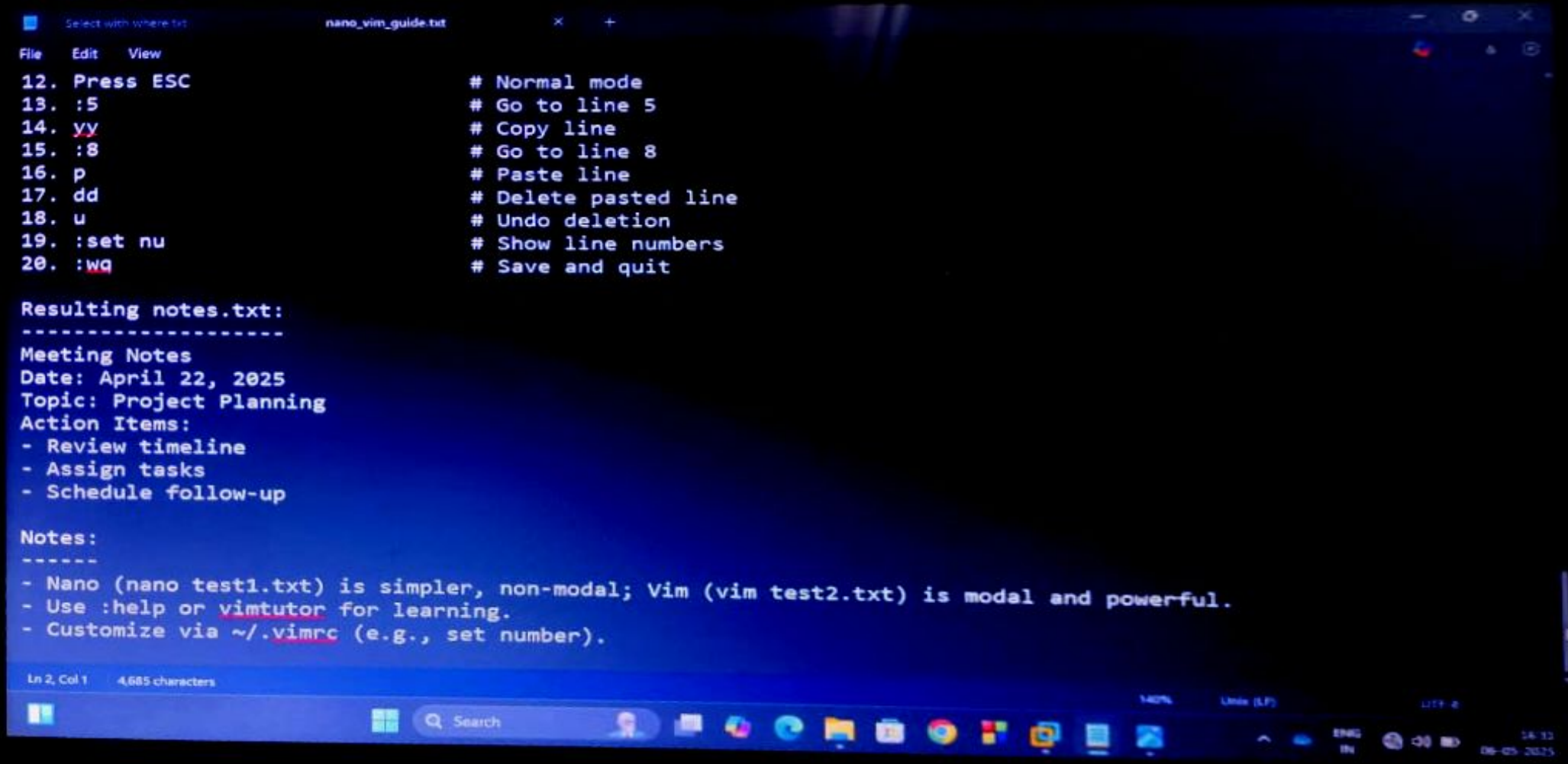

nano_vim_guide.txt
File Edit View
Common Options:

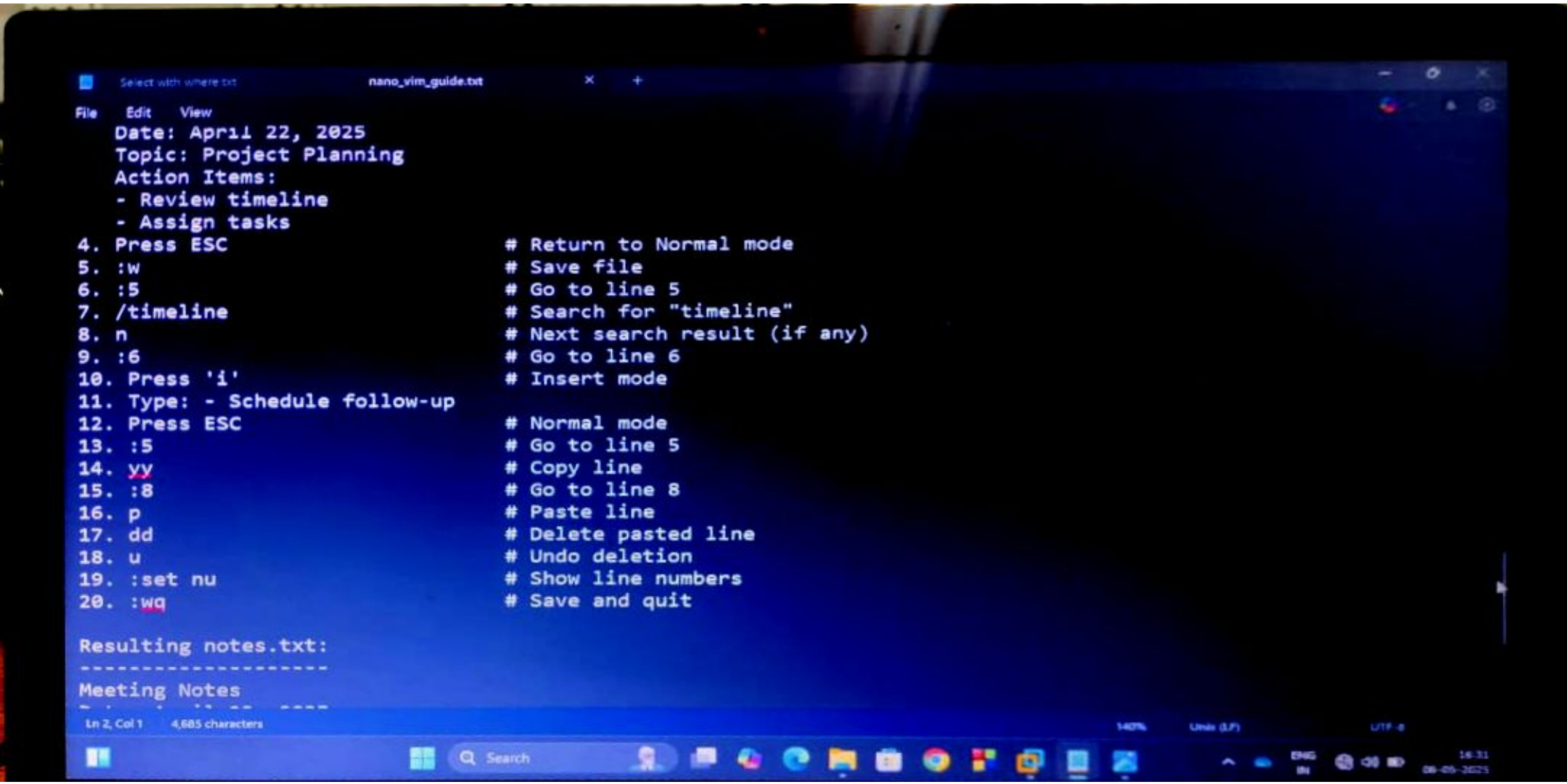
-v Start in view (read-only) mode
-e Start in Ex mode
-E Start in improved Ex mode
-R Read-only mode
-m Modifiable but no file saving
-y Easy mode
-Z Restricted mode
+NUM Start at line number NUM
+ Go to end of file
+/{pattern} Search for a pattern
Examples:
---
vim file.txt # Open file
vim +10 file.txt # Open file at line 10
vim +/searchterm file.txt # Open and search term
vim -R file.txt # Open in read-only mode
Explanations:

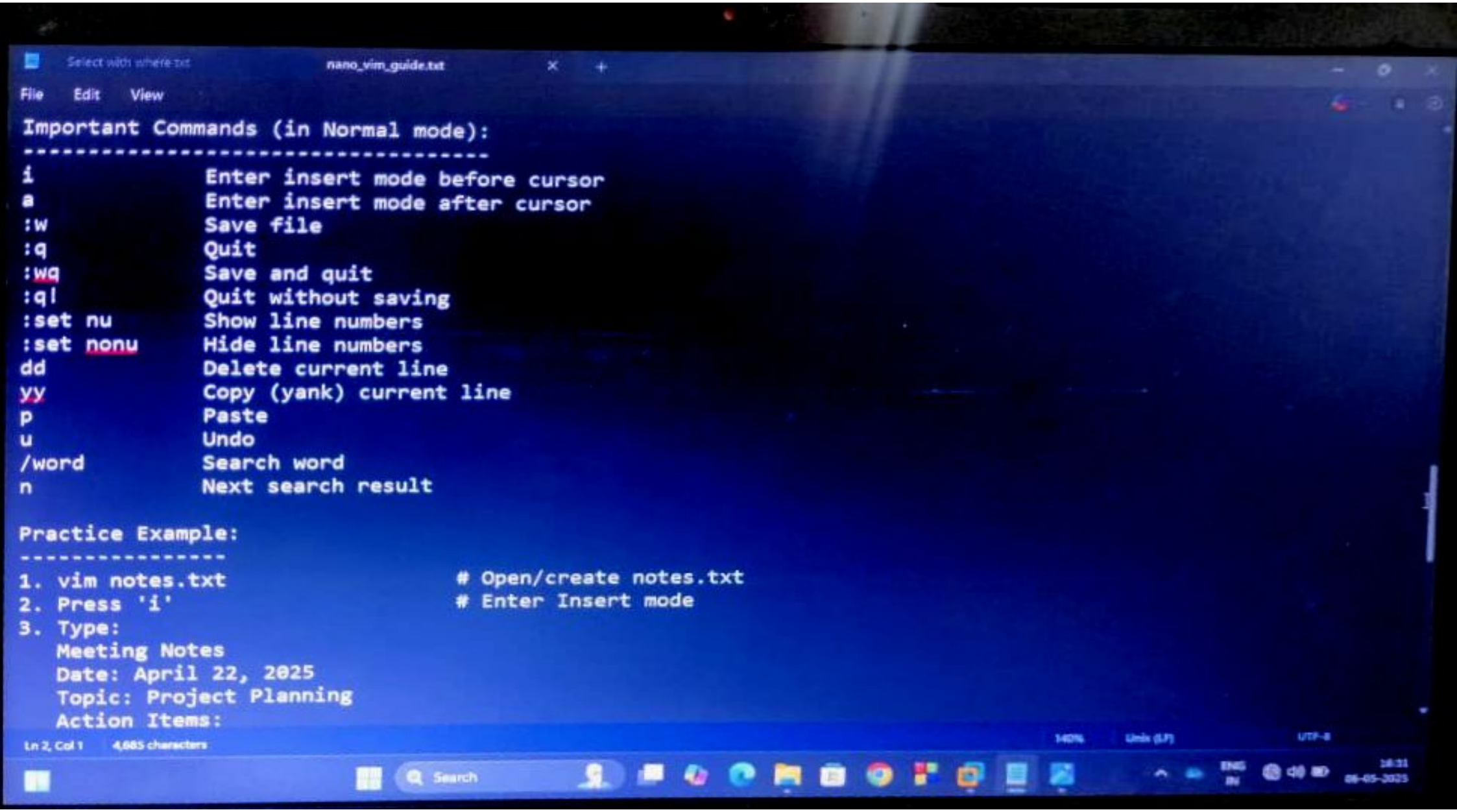
- vim file.txt: Opens file.txt in Normal mode. If file doesn't exist, creates a new buffer.
- vim +10 file.txt: Opens file.txt with cursor on line 10. If file has <10 lines, cursor goes to last line.
- vim +/searchterm file.txt: Opens file.txt with cursor on first occurrence of "searchterm".
Ln 2, Col 1
4,685 characters
Search

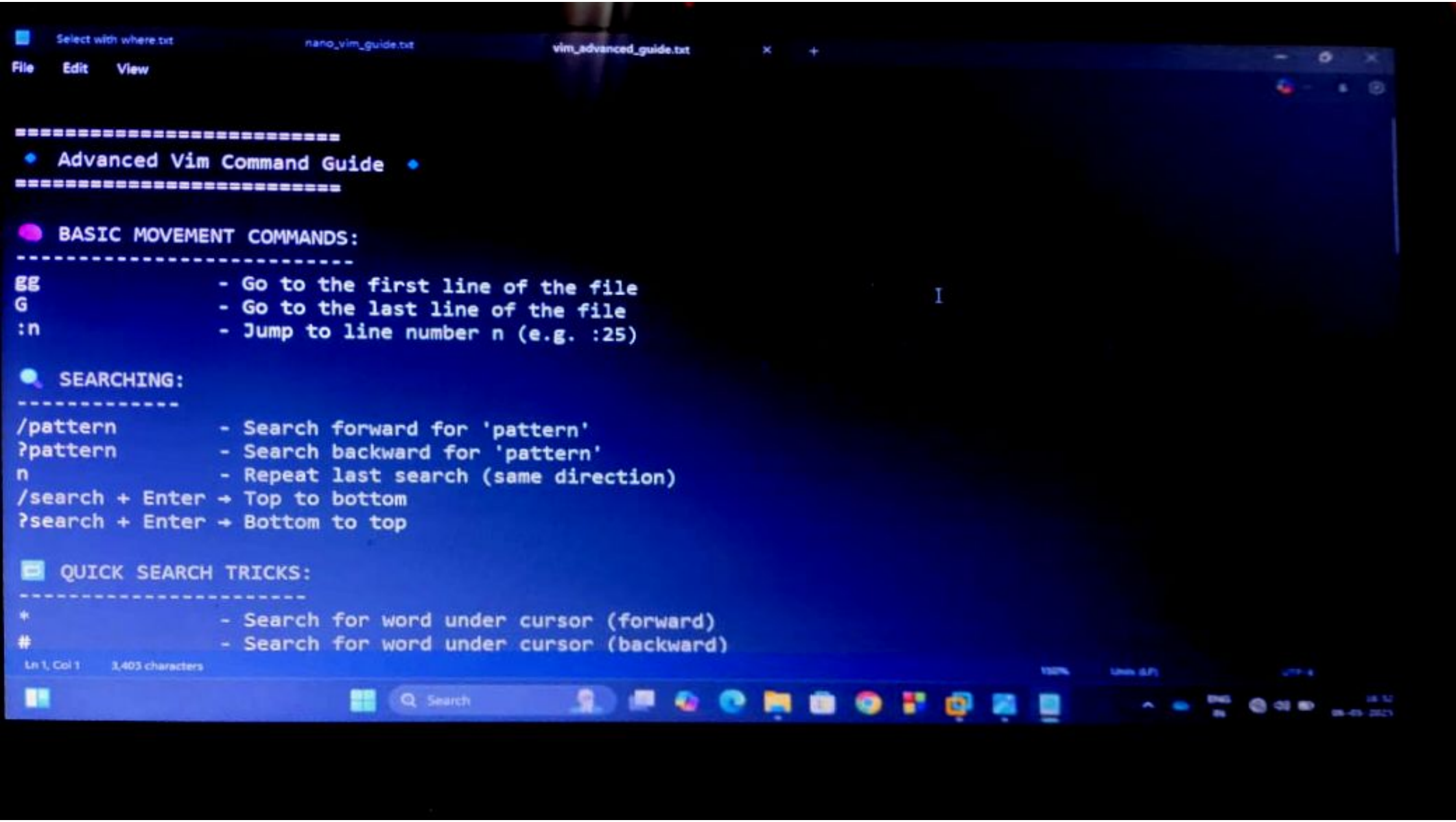

nano_vim_guide.txt
File Edit View
--linenumbers Show line numbers
Examples:
nano file.txt # Open or create file.txt
nano -B file.txt # Open with backup
nano +5,10 file.txt # Open file at line 5, column 10
In-Editor Shortcuts:
Ctrl + O Write (save) file
Ctrl + X Exit
Ctrl + K Cut current line
Ctrl + U Paste line
Ctrl + W Search
Ctrl + \ Replace
Vim - Vi IMproved (Powerful Terminal Text Editor)
Description: Vim is a highly configurable and powerful terminal text editor known for modal editing, extensive customization, and efficiency.
Basic Command:
Ln 2, Col 1 4,685 characters
Search

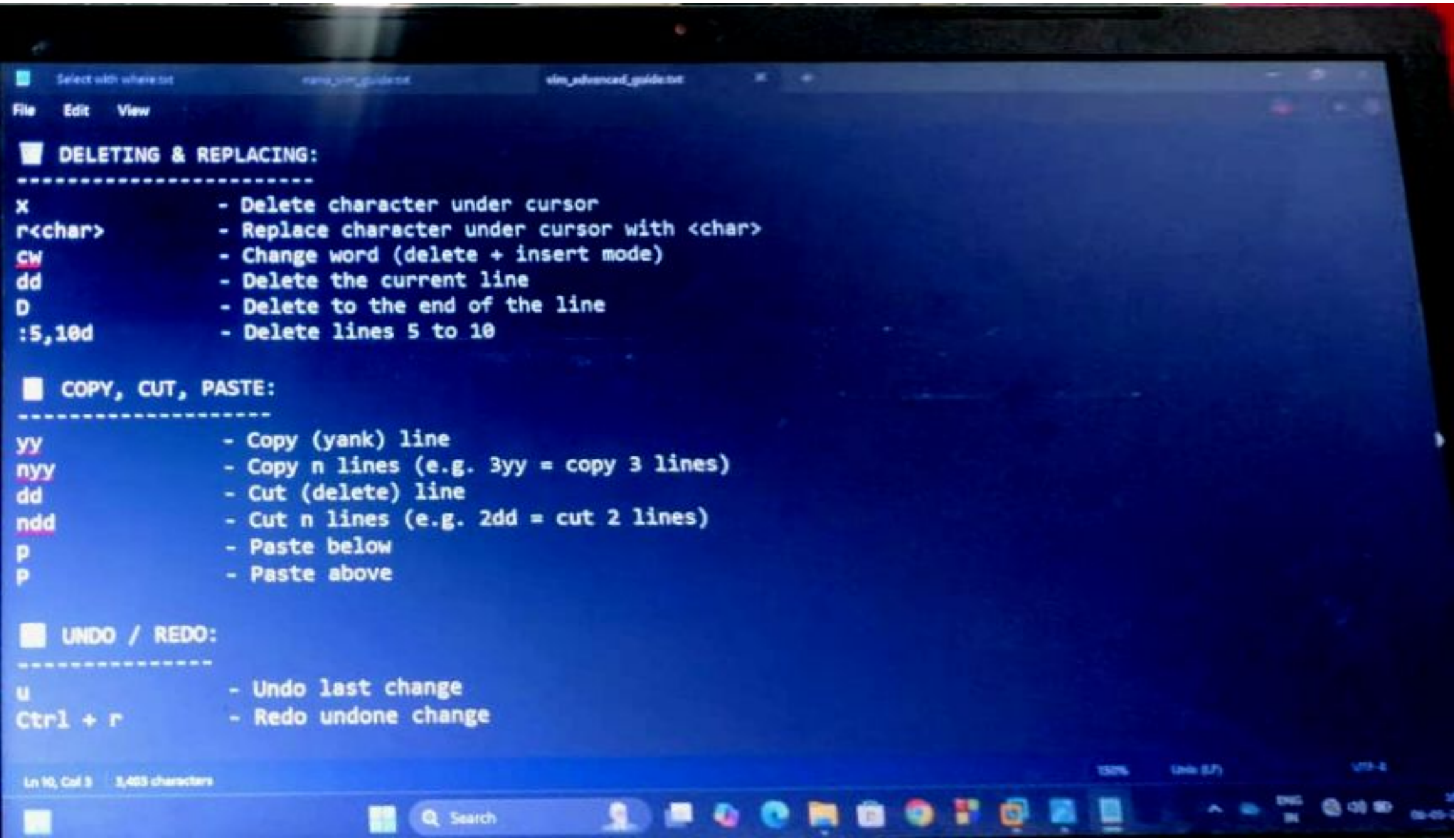

Select with where.txt
nano_vim_guide.txt
File Edit View
Basic Command:
vim [OPTIONS] [FILE]
Common Options:
-v Start in view (read-only) mode
-e Start in Ex mode
-E Start in improved Ex mode
-R Read-only mode
-m Modifiable but no file saving
-y Easy mode
-Z Restricted mode
+NUM Start at line number NUM
+ Go to end of file
+/{pattern} Search for a pattern
Examples:
vim file.txt # Open file
vim +10 file.txt # Open file at line 10
vim +/searchterm file.txt # Open and search term
vim -R file.txt # Open in read-only mode
Explanations:

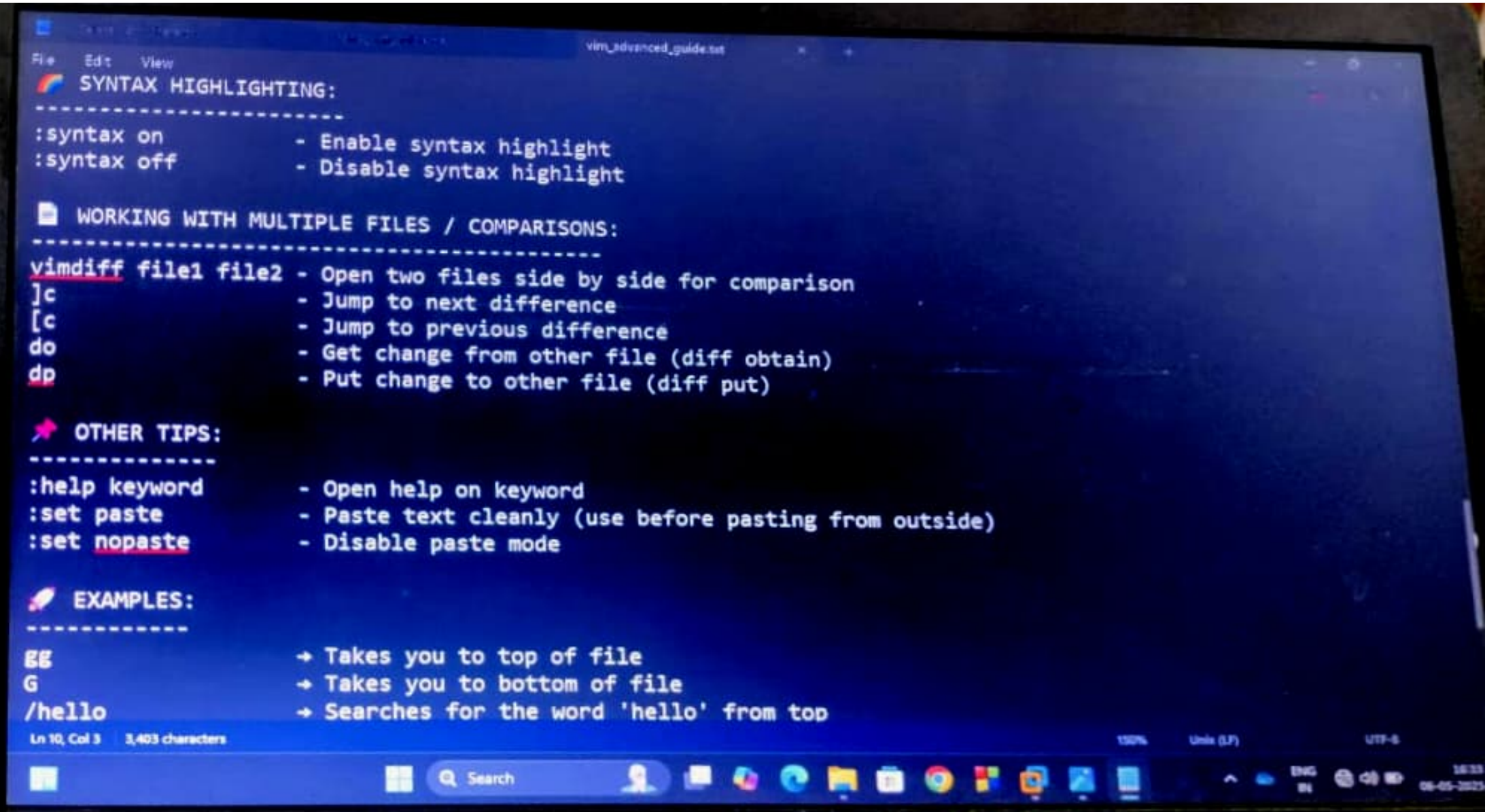

nano_vim_guide.txt
File Edit View
12. Press ESC # Normal mode
13. :5 # Go to line 5
14. yy # Copy line
15. :8 # Go to line 8
16. p # Paste line
17. dd # Delete pasted line
18. u # Undo deletion
19. :set nu # Show line numbers
20. :wq # Save and quit
Resulting notes.txt:
Meeting Notes
Date: April 22, 2025
Topic: Project Planning
Action Items:
- Review timeline
- Assign tasks
- Schedule follow-up
Notes:
- Nano (nano test1.txt) is simpler, non-modal; Vim (vim test2.txt) is modal and powerful.
- Use :help or vimtutor for learning.
- Customize via ~/.vimrc (e.g., set number).
Ln 2, Col 1 4,685 characters
Search

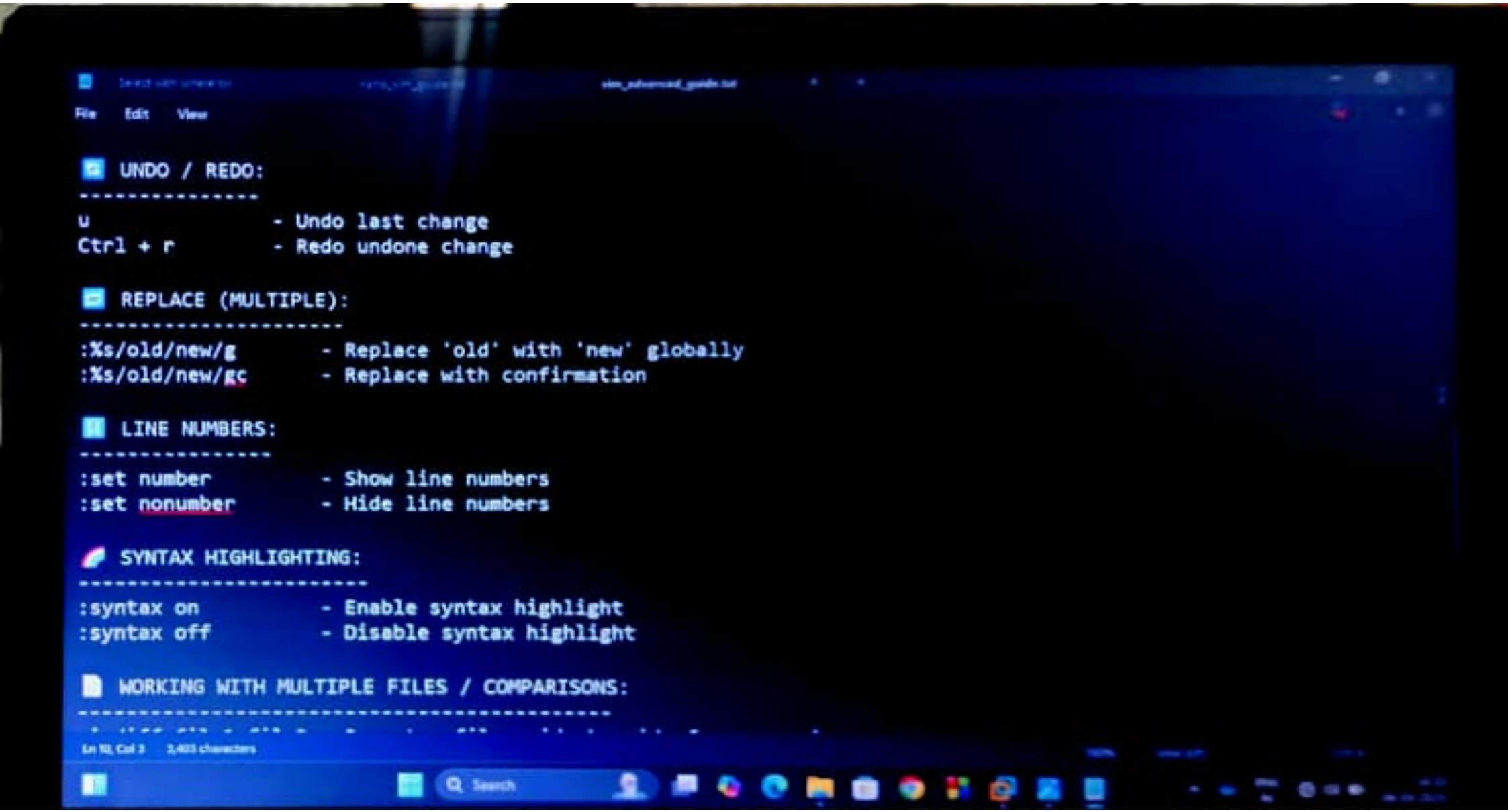

Select with where.txt
nano_vim_guide.txt
File Edit View
Date: April 22, 2025
Topic: Project Planning
Action Items:
- Review timeline
- Assign tasks
4. Press ESC # Return to Normal mode
5. :w # Save file
6. :5 # Go to line 5
7. /timeline # Search for "timeline"
8. n # Next search result (if any)
9. :6 # Go to line 6
10. Press 'i' # Insert mode
11. Type: - Schedule follow-up
12. Press ESC # Normal mode
13. :5 # Go to line 5
14. yy # Copy line
15. :8 # Go to line 8
16. p # Paste line
17. dd # Delete pasted line
18. u # Undo deletion
19. :set nu # Show line numbers
20. :wq # Save and quit
Resulting notes.txt:
Meeting Notes
Ln 2, Col 1
4,685 characters
140%
Unix (LF)
UTF-8
Search
ENG
IN
16:31
08-05-2025

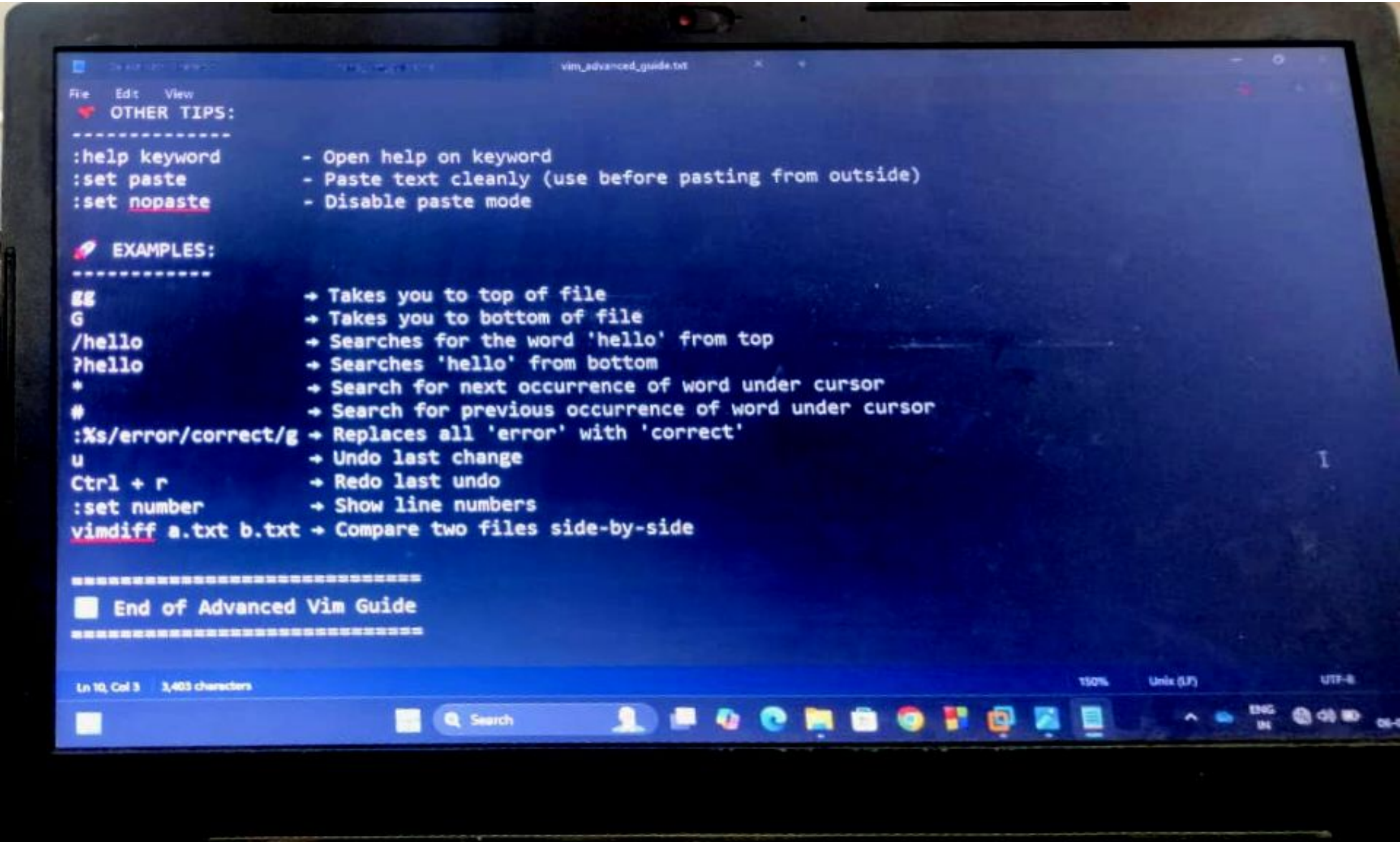

nano_vim_guide.txt
File Edit View
Important Commands (in Normal mode):
i Enter insert mode before cursor
a Enter insert mode after cursor
:w Save file
:q Quit
:wq Save and quit
:q! Quit without saving
:set nu Show line numbers
:set nonu Hide line numbers
dd Delete current line
yy Copy (yank) current line
p Paste
u Undo
/word Search word
n Next search result
Practice Example:
1. vim notes.txt # Open/create notes.txt
2. Press 'i' # Enter Insert mode
3. Type:
Meeting Notes
Date: April 22, 2025
Topic: Project Planning
Action Items:
Ln 2, Col 1
4,685 characters
Search

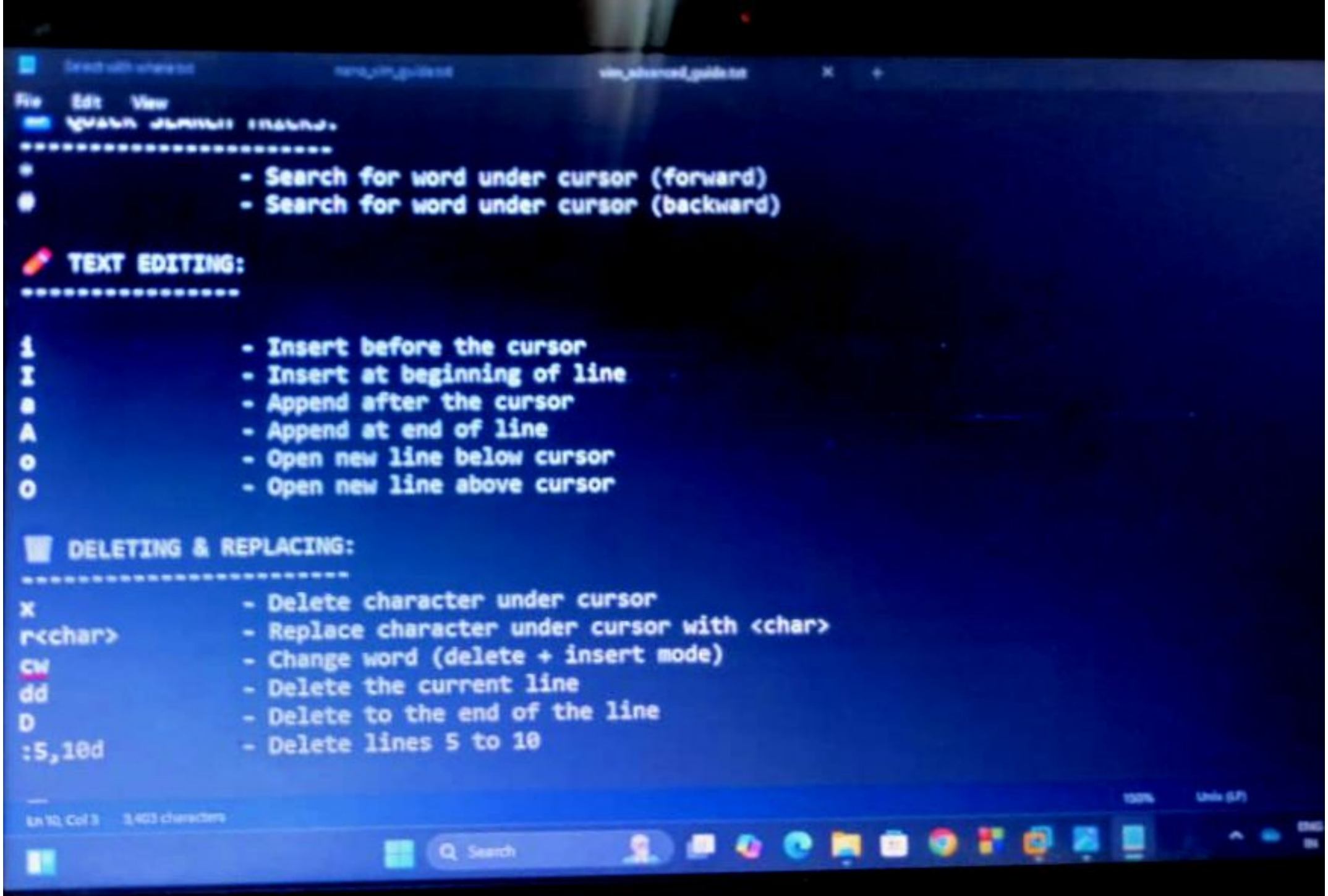

Select with where.txt
nano_vim_guide.txt
vim_advanced_guide.txt
File
Edit
View
==========
🔹 Advanced Vim Command Guide 🔹
==========
🧠 BASIC MOVEMENT COMMANDS:
gg - Go to the first line of the file
G - Go to the last line of the file
:n - Jump to line number n (e.g. :25)
🔍 SEARCHING:
/pattern - Search forward for 'pattern'
?pattern - Search backward for 'pattern'
n - Repeat last search (same direction)
/search + Enter → Top to bottom
?search + Enter → Bottom to top
🔄 QUICK SEARCH TRICKS:
* - Search for word under cursor (forward)
# - Search for word under cursor (backward)
Ln 1, Col 1
3,403 characters
Search

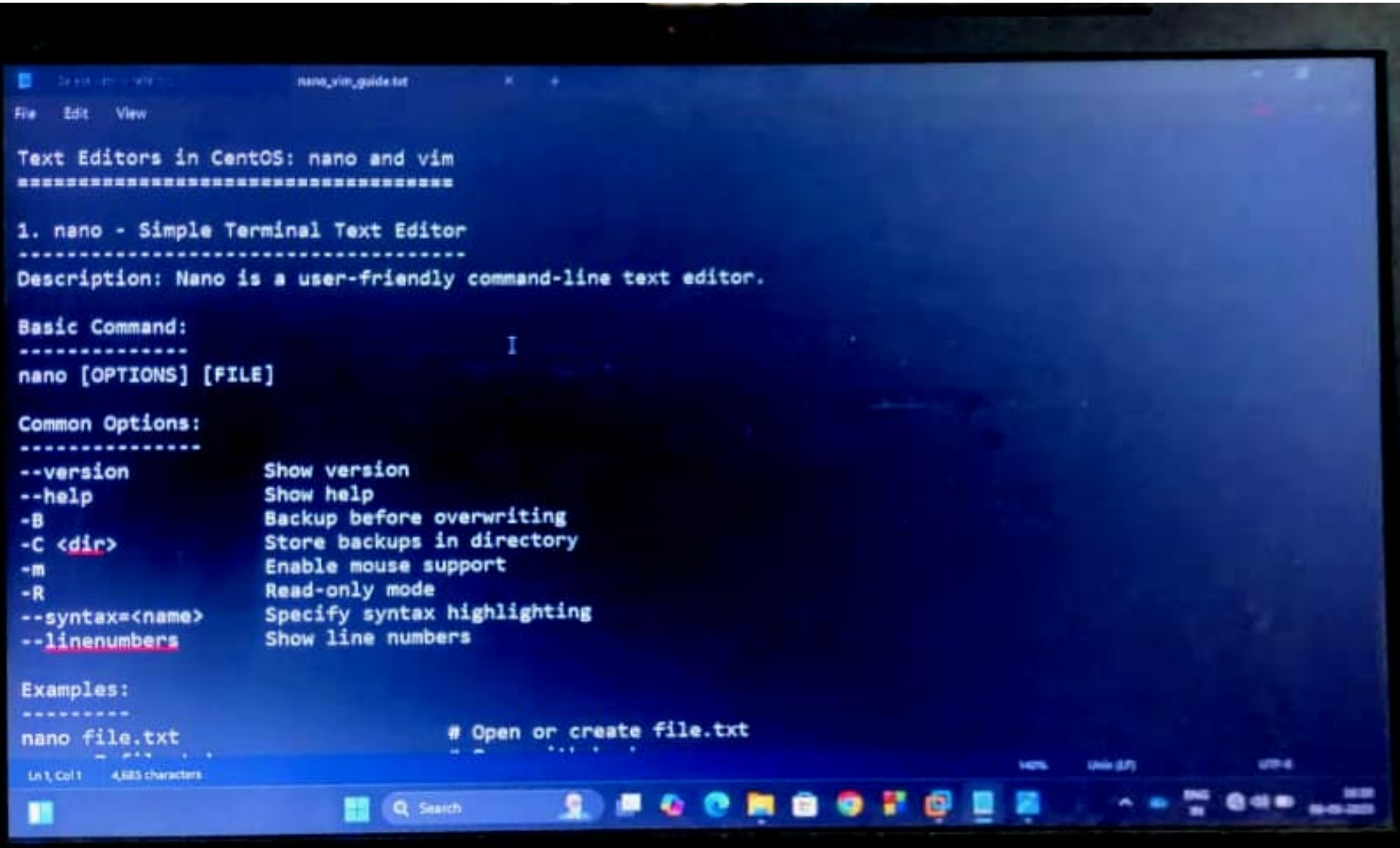

File Edit View
DELETING & REPLACING:
x - Delete character under cursor
r<char> - Replace character under cursor with <char>
cw - Change word (delete + insert mode)
dd - Delete the current line
D - Delete to the end of the line
:5,10d - Delete lines 5 to 10
COPY, CUT, PASTE:
yy - Copy (yank) line
nyy - Copy n lines (e.g. 3yy = copy 3 lines)
dd - Cut (delete) line
ndd - Cut n lines (e.g. 2dd = cut 2 lines)
p - Paste below
P - Paste above
UNDO / REDO:
u - Undo last change
Ctrl + r - Redo undone change
Search


File Edit View
vim_advanced_guide.txt
SYNTAX HIGHLIGHTING:
:syntax on - Enable syntax highlight
:syntax off - Disable syntax highlight
WORKING WITH MULTIPLE FILES / COMPARISONS:
vimdiff file1 file2 - Open two files side by side for comparison
]c - Jump to next difference
[c - Jump to previous difference
do - Get change from other file (diff obtain)
dp - Put change to other file (diff put)
OTHER TIPS:
:help keyword - Open help on keyword
:set paste - Paste text cleanly (use before pasting from outside)
:set nopaste - Disable paste mode
EXAMPLES:
gg → Takes you to top of file
G → Takes you to bottom of file
/hello → Searches for the word 'hello' from top
Ln 10, Col 3
3,403 characters
150%
Unix (LF)
UTF-8
Search


File Edit View
UNDO / REDO:
u - Undo last change
Ctrl + r - Redo undone change
REPLACE (MULTIPLE):
:%s/old/new/g - Replace 'old' with 'new' globally
:%s/old/new/gc - Replace with confirmation
LINE NUMBERS:
:set number - Show line numbers
:set nonumber - Hide line numbers
SYNTAX HIGHLIGHTING:
:syntax on - Enable syntax highlight
:syntax off - Disable syntax highlight
WORKING WITH MULTIPLE FILES / COMPARISONS:
Search


vim_advanced_guide.txt
File Edit View
OTHER TIPS:
:help keyword - Open help on keyword
:set paste - Paste text cleanly (use before pasting from outside)
:set nopaste - Disable paste mode
EXAMPLES:
gg → Takes you to top of file
G → Takes you to bottom of file
/hello → Searches for the word 'hello' from top
?hello → Searches 'hello' from bottom
* → Search for next occurrence of word under cursor
# → Search for previous occurrence of word under cursor
:%s/error/correct/g → Replaces all 'error' with 'correct'
u → Undo last change
Ctrl + r → Redo last undo
:set number → Show line numbers
vimdiff a.txt b.txt → Compare two files side-by-side
End of Advanced Vim Guide
Ln 10, Col 3
3,403 characters
150%
Unix (LF)
UTF-8
Search


File Edit View
* - Search for word under cursor (forward)
# - Search for word under cursor (backward)
TEXT EDITING:
i - Insert before the cursor
I - Insert at beginning of line
a - Append after the cursor
A - Append at end of line
o - Open new line below cursor
O - Open new line above cursor
DELETING & REPLACING:
x - Delete character under cursor
r<char> - Replace character under cursor with <char>
cw - Change word (delete + insert mode)
dd - Delete the current line
D - Delete to the end of the line
:5,10d - Delete lines 5 to 10
Search


nano_vim_guide.txt
File Edit View
Text Editors in CentOS: nano and vim
1. nano - Simple Terminal Text Editor
Description: Nano is a user-friendly command-line text editor.
Basic Command:
nano [OPTIONS] [FILE]
Common Options:
--version Show version
--help Show help
-B Backup before overwriting
-C <dir> Store backups in directory
-m Enable mouse support
-R Read-only mode
--syntax=<name> Specify syntax highlighting
--linenumbers Show line numbers
Examples:
nano file.txt # Open or create file.txt
Ln 1, Col 1 4,685 characters
Search