

# File Editor Tool(Tted)

## Command-Line Text File Manipulation

Version 1.0  
Malinga RK

December 30, 2025

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Key Features . . . . .	2
<b>2</b>	<b>Operating Modes</b>	<b>2</b>
2.1	Terminal Mode (Default) . . . . .	2
2.2	In-Place Edit Mode . . . . .	2
<b>3</b>	<b>Command Reference</b>	<b>2</b>
3.1	Reading Operations . . . . .	2
3.1.1	View Single Line . . . . .	2
3.1.2	View Line Range . . . . .	3
3.2	Modification Operations . . . . .	3
3.2.1	Delete Lines . . . . .	3
3.2.2	Append Text . . . . .	3
3.2.3	Replace Text . . . . .	4
<b>4</b>	<b>Usage Examples</b>	<b>4</b>
4.1	Workflow Example . . . . .	4
4.2	Batch Operations . . . . .	4
<b>5</b>	<b>Error Handling</b>	<b>5</b>
5.1	Common Errors . . . . .	5
5.2	Error Messages . . . . .	5
<b>6</b>	<b>Technical Details</b>	<b>5</b>
6.1	Line Numbering . . . . .	5
6.2	Buffer Limitations . . . . .	5
6.3	File Operations . . . . .	5
<b>7</b>	<b>Best Practices</b>	<b>6</b>
7.1	Safety Guidelines . . . . .	6
7.2	Performance Tips . . . . .	6
<b>8</b>	<b>Compilation</b>	<b>6</b>
<b>9</b>	<b>Quick Reference</b>	<b>6</b>

## 1 Introduction

The File Editor Tool is a command-line utility written in C for viewing and manipulating text files. It provides two distinct operating modes: **Terminal Mode** (preview only) and **In-Place Edit Mode** (file modification).

### 1.1 Key Features

- View specific lines or line ranges
- Delete lines from files
- Append text after specific lines
- Replace text throughout files
- Safe preview mode (default)
- In-place editing with `-i` flag

## 2 Operating Modes

### 2.1 Terminal Mode (Default)

Terminal mode is the **safe default** operating mode. It displays the result of operations without modifying the original file. This allows you to preview changes before committing them.

#### Safety Feature

All modification commands (`--delete`, `--append`, `--replace`) run in preview mode by default. The file is never modified unless the `-i` flag is explicitly provided.

### 2.2 In-Place Edit Mode

To actually modify files, prefix your command with the `-i` flag. This mode creates a temporary file, performs the operation, and replaces the original file.

## 3 Command Reference

### 3.1 Reading Operations

These operations always work in terminal mode (read-only).

#### 3.1.1 View Single Line

##### Syntax

```
./program --line [line_number] [file]
```

**Description:** Displays a specific line from the file.

**Example:**

```
./program --line 5 myfile.txt
```

### 3.1.2 View Line Range

#### Syntax

```
./program --lines [start,end] [file]
```

**Description:** Displays a range of lines with line numbers.

**Example:**

```
./program --lines 10,20 myfile.txt
```

Output format:

```
10:This is line 10  
11:This is line 11  
12:This is line 12  
...
```

## 3.2 Modification Operations

### 3.2.1 Delete Lines

#### Terminal Mode (Preview)

```
./program --delete [start,end] [file]
```

#### In-Place Mode (Modifies File)

```
./program -i --delete [start,end] [file]
```

**Description:** Removes the specified line range from the file.

**Examples:**

```
# Preview deletion  
./program --delete 5,10 document.txt  
  
# Actually delete lines 5-10  
./program -i --delete 5,10 document.txt
```

### 3.2.2 Append Text

#### Terminal Mode (Preview)

```
./program --append [line_number] [file] "text"
```

#### In-Place Mode (Modifies File)

```
./program -i --append [line_number] [file] "text"
```

**Description:** Inserts text as a new line after the specified line number.

**Examples:**

```
# Preview append
./program --append 3 notes.txt "This is a new line"

# Actually append
./program -i --append 3 notes.txt "This is a new line"
```

**3.2.3 Replace Text****Terminal Mode (Preview)**

```
./program --replace "old_text/new_text" [file]
```

**In-Place Mode (Modifies File)**

```
./program -i --replace "old_text/new_text" [file]
```

**Description:** Replaces the first occurrence of `old_text` with `new_text` on each line.

**Examples:**

```
# Preview replacement
./program --replace "hello/goodbye" letter.txt

# Actually replace
./program -i --replace "hello/goodbye" letter.txt
```

## 4 Usage Examples

### 4.1 Workflow Example

A typical workflow involves previewing changes before applying them:

```
# 1. View the current state
./program --lines 1,50 config.txt

# 2. Preview a deletion
./program --delete 10,15 config.txt

# 3. If satisfied, apply the change
./program -i --delete 10,15 config.txt

# 4. Verify the result
./program --lines 1,50 config.txt
```

### 4.2 Batch Operations

```
# Preview multiple changes
./program --delete 5,10 data.txt
./program --append 20 data.txt "New section header"
./program --replace "old_version/new_version" data.txt
```

```
# Apply all changes
./program -i --delete 5,10 data.txt
./program -i --append 20 data.txt "New section header"
./program -i --replace "old_version/new_version" data.txt
```

## 5 Error Handling

### 5.1 Common Errors

**Invalid line number** The specified line number exceeds the total lines in the file.

**Invalid line range** The start line is greater than the end line, or either exceeds the file length.

**Cannot open file** The specified file does not exist or lacks read permissions.

**Cannot create temporary file** Insufficient permissions or disk space for in-place editing.

### 5.2 Error Messages

```
invalid line number
invalid line range
Invalid range: Operation is not permitted
Error: Cannot open file
Error: Cannot create temporary file
Error: Text to append is required
Error: Replace format should be "OldText/NewText"
```

## 6 Technical Details

### 6.1 Line Numbering

- Line numbers start at 1 (not 0)
- Line ranges are inclusive: 5,10 includes both lines 5 and 10
- Maximum line number supported: 999,999,999 (limited by `unsigned int`)

### 6.2 Buffer Limitations

- Maximum line length: 512 characters (including newline)
- Lines exceeding this limit will be processed in chunks
- Maximum line number length in commands: 8 digits

### 6.3 File Operations

#### In-Place Mode Process:

1. Read from original file
2. Write modified content to `temp_file.tmp`
3. Close both files
4. Remove original file
5. Rename temporary file to original filename

## 7 Best Practices

### 7.1 Safety Guidelines

- Always preview changes with terminal mode first
- Keep backups of important files before using `-i` mode
- Test operations on sample files before production use
- Use version control systems for critical files

### 7.2 Performance Tips

- For large files, specify exact line ranges rather than reading entire files
- Batch multiple preview operations before applying changes
- Consider file size when performing replace operations

## 8 Compilation

To compile the program:

```
gcc -o fileedit rtt.c -Wall -Wextra
```

Recommended compilation flags:

- `-Wall`: Enable all warnings
- `-Wextra`: Enable extra warnings
- `-O2`: Optimization level 2 (optional)
- `-std=c99`: Use C99 standard (optional)

## 9 Quick Reference

Command	Mode	Effect
<code>--line N file</code>	Terminal	View line N
<code>--lines N,M file</code>	Terminal	View lines N-M
<code>--delete N,M file</code>	Terminal	Preview deletion
<code>-i --delete N,M file</code>	In-place	Delete lines
<code>--append N file "text"</code>	Terminal	Preview append
<code>-i --append N file "text"</code>	In-place	Append text
<code>--replace "a/b" file</code>	Terminal	Preview replace
<code>-i --replace "a/b" file</code>	In-place	Replace text

Table 1: Command Quick Reference