Step one:

We use a while loop with `case` statements to handle different options:

When the user writes different commands like –n, -v or both.

We set Boolean flags when options are detected

We handle for combined options like `-vn` or `-nv`

--help option displays usage information and exits immediately

```bash
show_line_numbers=false
invert_match=false
search_string=""
filename=""
usage="Usage: $0 [-n] [-v] <search_string> <filename>"

# Process command line options
while [[ $# -gt 0 ]]; do
    case "$1" in
        -n)
            show_line_numbers=true
            shift
            ;;
        -v)
            invert_match=true
            shift
            ;;
        -vn|-nv)
            show_line_numbers=true
            invert_match=true
            shift
            ;;
        --help)
            echo "mygrep.sh - A simplified grep implementation"
            echo "$usage"
            echo "Options:"
            echo "  -n        Show line numbers"
            echo "  -v        Invert match (show non-matching lines)"
            echo "  --help    Show this help message"
            exit 0
            ;;
        -*)
            echo "Error: Unknown option $1" >&2
            echo "$usage" >&2
            exit 1
            ;;
        *)
            if [[ -z "$search_string" ]]; then
                search_string="$1"
            elif [[ -z "$filename" ]]; then
                filename="$1"
            else
                echo "Error: Too many arguments" >&2
                echo "$usage" >&2
                exit 1
            fi
            shift
            ;;
    esac
```

Step two:

Then we validate the input; making sure that the required inputs are written (the search_string and the filename) and handles when one argument is missing or both

```
# Validate input
if [[ -z "$search_string" ]]; then
    echo "Error: Missing search string" >&2
    echo "$usage" >&2
    exit 1
fi

if [[ -z "$filename" ]]; then
    echo "Error: Missing filename" >&2
    echo "$usage" >&2
    exit 1
fi

if [[ ! -f "$filename" ]]; then
    echo "Error: File '$filename' not found" >&2
    exit 1
fi
```

Step three:

We perform the search; for case insensitive we use grep –i, then we read the given file; line by line

–r: prevents backslash interpretation,

if the grep finds match it sets match = true

If there are no matchings it sets match = false

```
# Perform the search (using grep for reliable case-insensitive matching)
grep_command="grep -i"
if [[ $invert_match == true ]]; then
    grep_command="$grep_command -v"
fi

line_number=0
while IFS= read -r line; do
    ((line_number++))
    # Use grep for reliable pattern matching
    if echo "$line" | $grep_command -q "$search_string"; then
        match=true
    else
        match=false
    fi

    # Handle invert match option (already handled by grep -v)
    if [[ $match == true ]]; then
        if [[ $show_line_numbers == true ]]; then
            printf "%d:" "$line_number"
        fi
        echo "$line"
    fi
done < "$filename"

exit 0
```