OPPE System Commands: Topic Organization by Category

I've created a comprehensive study guide organizing all the most important topics from your OPPE question papers into 8 broad categories.

Quick Overview of the 8 Main Categories

Category	Priority	Key Focus
Bash Fundamentals	⭐⭐⭐	Reading input, loops, arithmetic, arrays, error handling
File & Directory Operations	⭐⭐⭐	mkdir, touch, mv, find, organizing files systematically
Permissions & Log Analysis	⭐⭐	Permission checks, parsing ls/du output, analyzing auth logs
sed Text Processing	⭐⭐⭐⭐	Field swapping, line insertion, pattern masking, multi-step replacements
awk Data Processing	⭐⭐⭐⭐	Grouping, counting, min/max calculations, CSV processing
grep & Pattern Matching	⭐⭐	Basic search, regex, extracting fields, pipeline combinations
Utility Scripts	⭐⭐	Nth line extraction, file swapping, log rotation, deduplication
Complex Multi-tool Problems	⭐⭐⭐	Chaining 3-4 commands together for integrated solutions

Key Insights from Analysis

Highest Impact Areas (Focus First):

- **sed mastery** is critical—appears in nearly every OPPE-2 set with field swapping, line editing, and regex substitution
- **awk grouping patterns** (min/max per group, counting per category) repeat across multiple employee/school data problems
- **Bash file operations & loops** form the foundation for at least 60% of all questions

Most Repeated Exam Patterns:

1. **Employee data analysis** (leaves by year, min leaves, email generation)—appears 4+ times across practice sets
2. **File organization by extension**—common structural question

3. **Chaining grep → cut → awk → sort | uniq**—standard multi-tool pipeline

Study Progression (if exam is 2-3 weeks away):

- **Week 1:** Master Bash basics + file operations
- **Week 2:** Deep dive into sed and awk
- **Week 3:** Practice multi-tool problems + mock exams under time pressure

The study guide includes specific code examples, frequency ratings, and a final emergency cheat sheet with the most essential commands. Use it as your reference while practicing past papers!

Key Topics Frequently Asked in OPPE System Commands (OPPE-1 & OPPE-2)

Most questions across these papers revolve around a small, very consistent core of command-line and text-processing topics. These are the areas to focus on.

---

1. Core Bash Scripting Fundamentals

Patterns appearing repeatedly:

- Reading input from STDIN (`read`, multiple lines, reading until EOF)
- Using command line arguments and `getopts` style option parsing
- Conditional checks with `if`, `elif`, `else` and `test` / `[ ]`
- Arithmetic using `expr`, `$(())`, modulo, comparisons etc.
- `for`, `while` loops over:
  - numeric sequences (`seq`)
  - arrays
  - filenames from globbing or command output
- Returning status via exit codes (`exit` with appropriate code)

Common problem types:

- Classifying or filtering numbers: positive/negative, special conditions, divisible by multiple numbers, etc.
- Summation / aggregation:
  - sum of numbers at odd/even positions in an array or in arguments
  - max/min in a list of numbers

- Generating combinations of filenames by nested loops (e.g. `fileXYZ.txt` style patterns)

---

2. Working with Files and Directories

Very frequent themes:

- Creating and organizing directories with `mkdir`, `mkdir -p`
- Creating empty files with `touch`
- Moving files based on patterns or extensions with `mv`
- Using `ln -s` to create symbolic links
- Checking permissions (`-r`, `-w`, `-x`), file/directory tests (`-f`, `-d`, `-s` etc.)
- Deleting files (`rm`) carefully based on conditions

Typical question patterns:

- Construct a given directory tree and files exactly as specified.
- Organize "mixed" files in current directory into folders by extension.
- Move selected files from `source` to `destination` based on name pattern (e.g. `imageXrgb.txt`).
- Remove duplicates in a directory using a helper script (`printdup.sh`) and lexicographic order.
- Implement "log rotation" pattern:
  - For a single file (`network.log.0` → `.1`, `.2`, …) based on size using `du -b`.
  - For per-service logs using highest suffix logic and then making symlinks.

---

3. `find`, `ls` and Permissions / Attributes

Often examined concepts:

- Using `find` to:
  - list empty files, then print and remove them
  - filter by name patterns, extensions, size, type (file vs directory)
- Interpreting `ls -l` output (fields: permissions, size, date, filename)
- Summing file sizes with `awk` based on date ranges and size thresholds.

Key subskills:

- Understanding permission strings and mapping to read/write/execute for user
- Writing functions (like `perms`) that look only at *user* permissions
- Filtering by month and day ranges in `ls -l` output.

---

4. Text Processing with `grep`, `egrep`, `sed` and `awk`

This is the **single biggest cluster** for OPPE-2.

4.1 `grep` / `egrep` / `grep -oP`

Important ideas:

- Pattern extraction from log files:
  - Logins (`session opened for user ...`)
  - Failed/Successful logins
  - Filtering by date or word (`installed`, `not-installed`, `guest`, `su`)
- Using:
  - `grep`, `egrep`, `grep -o` to capture specific fields/words
  - `grep -v` to exclude lines

Common tasks:

- Extract list of unique users from auth logs (`sort -u`).
- Extract a specific last/first match and then print subset of fields (`tail -n 1`, `awk '{print ...}'`).

4.2 `sed` – Very High Priority

Papers repeatedly use `sed` for:

1. **Field and delimiter manipulation**
   - Swapping first and second fields with custom separators (`:` or spaces).
   - Changing multi-space delimiters to include a pipe symbol.
   - Converting CSV with quotes to TSV.
2. **Line-based editing**
   - Replacing `?` at end of line with `!`.
   - Inserting or appending lines before/after certain patterns:

    - Inserting headers/footers (`CONFIDENTIAL`, copyright lines).
    - Inserting separators every Nth line (`#####` after every 5th or 10th line).
  - Deleting specific lines (block separators, lines with certain patterns).
3. **Pattern substitution / masking sensitive information**
  - Case-insensitive replacements (e.g., weekdays `sun` → `1` etc.).
  - Matching and masking PAN-like IDs using complex regex.
  - Replacing lines containing `Password`, `Address`, or all-digits with `REDACTED`.
4. **Date and month manipulation**
  - Shifting dates across months (e.g., "July/August schedule shifted by +1 month, handling 31st specially").
  - Paying attention to replacement order.
5. **Function boundary marking in shell code**
  - Detecting function definition lines by pattern and inserting:
    - `START FUNCTION` before
    - `END FUNCTION` after the closing `}` line
  - Reusing this logic in multiple variants (single file, or all `.sh` files).

Conclusion: Master:

- `s///` substitutions (with flags: `g`, `I`, etc.)
- Address ranges (`FROM,TO`)
- `i`, `a`, `c` commands (insert/append/change line)
- Line numbering and conditional inserts (`10i`, `10a` patterns, `~` addressing).

4.3 `awk` – Very High Priority

Heavily used for:

1. **CSV-like data processing**
  - Setting field separator `FS` to comma or space.
  - Reading and grouping by keys (school code, department, year).
  - Aggregations:
    - Count per group (e.g., students > 300 marks per school).
    - Min/Max per group (min leaves per department).
    - Global min / average with integer truncation.
  - Output formats like:
    - `Schoolcode,count`
    - `departmentIDMINLEAVESTAKEN`

    - `year averageLeaves`
2. **EmployeeDetails patterns (repeated multiple times):**
    - Computing average leaves by birth year (1997–2000).
    - Finding employees with minimum leaves.
    - Printing female employees' email IDs from employee ID.
3. **Text structure manipulation**
    - Swapping consecutive lines (keys and values).
    - Swapping columns conditionally on NF (field count).
    - Identifying unintentionally repeated words (case-insensitive, print in lowercase).
4. **Log analysis**
    - Counting or summing over fields with date and size filters.
    - Printing only roll numbers from marks CSV.
    - Designing small `awk` scripts embedded in bash for scanning `.c` file line lengths.

Key techniques:

- `BEGIN` and `END` blocks.
- Arrays for counting or tracking minima per key.
- Checking `NF`, `length()`, `int()`.
- Using `tolower()` for case-insensitive comparisons.
- Maintaining per-record state (previous word/line etc.).

---

5. Arrays and Command Line Arguments in Bash

Recurrent ideas:

- Arrays created by "suffix code" (you are given array `numberarr` and must iterate).
- Summing specific positions (odd indices, etc.).
- Distinguishing between:
    - Positional parameters `$1`, `$2`, …, `$#`
    - Last argument `${!#}` style (or `eval`/indirect reference trick) when writing custom logic.
- Using `getopts` to parse options like `-l`, `-w`, `-n`, `-s str` and then:
    - Counting lines, words, numeric-only lines, and occurrences of a substring using `sed` rather than `wc`/`awk`.

---

6. Logs and `dpkg.log` / `network.log` / `myauth.log`

These appear many times with variations:

- Extract installed package names on specific date with `grep`, `cut`, `sort | uniq`.
- Decide if a package is **still installed** based on the last `installed`/`not-installed` entry for that package in a date range.
- Rotate and rename network logs when size crosses threshold.
- Mine `myauth.log` for:
  - usernames for successful `su`
  - count or list of users with certain patterns
  - last guest login line, etc.

Important skills:

- Combining `grep/egrep` with `cut`, `awk`, `tail`, sorting.
- Understanding log line structure (fields: date, time, status, package name).

---

7. Exam-Style "Utility" Scripts

Frequently tested "classic" utilities you should be comfortable writing:

- **n-th line from a file** given `n` as variable (`head` + `tail`, or `sed -n "${n}p"`).
- **Swapping contents of two files** with error handling:
  - Wrong number of arguments → specific error message and exit code.
  - Non-existent files → different code/message.
  - Non-writable files → different code.
- **Checking / fixing execute permission** on a script before running and redirecting/appending output.
- **Log rotation utilities** as mentioned.
- Simple classification script reading a value and printing label (`PNUM`, `NNUM`, `STRING`, etc.).

---

8. NPPE / Practice Question Patterns (What they Emphasize)

From the combined NPPE practice sets:

- `sed` for:
  - swapping uppercase/lowercase segments around `_`
  - selecting word ranges between markers (`FROM` and `TO`)
  - editing functions with START/END markers
  - line counting and pattern counting instead of `wc`
- `awk` for:
  - repeated words
  - sums from CSV
  - min/max per grouping
  - employee leaves & emails (again)
- Bash for:
  - sums of CLI arguments with validation via regex
  - small utilities like max/min finder, prime checker
  - pattern-based log analysis

These confirm that if you are strong in `sed`, `awk`, **Bash loops/conditionals, file/log handling, and regex**, you cover the majority of OPPE questions.

How to Use This for Exam Prep

If time is short, prioritize in this order:

1. **Sed mastery**
   Field swap, regex patterns (with `[]`, `\d` equivalents, quantifiers, `I` flag), `i/a/c`, line numbers and ranges.
2. **Awk data processing**
   Grouping by keys, counting, min/max, averages, using arrays and `BEGIN/END` blocks.
3. **Bash scripting basics & file handling**
   `read`, loops, conditions, arrays, `getopts`, permissions, directory and file manipulations.
4. **Log and CSV style parsing using grep/sed/awk combinations**

If you want, next step can be a concise **topic-wise checklist with 1–2 representative practice questions per topic** mapped specifically to your PDFs.

1. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/50059553/123dcacc-203a-4585-9cfe-aacc28fa4408/OPPE-1-set-1-and-set-2-Q-sol-_2.pdf

2. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/50059553/2a54939a-ec5e-4f28-963e-b602dfe41a45/OPPE-2-set-1-and-set-2-Q-sol.pdf
3. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/50059553/9939d479-e3bb-4c38-8da1-4265b05e611a/OPPE-2-ALL-NPPEs.pdf
4. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/50059553/d337adc9-740f-40b3-a801-1129467d336a/OPPE-2-Ans-Key.pdf
5. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/50059553/a15c938c-7533-412b-a25c-85bb4f126de8/SC-OPPE-22T1.pdf