MCQ Examination - Performance Report

Grader: Shuvam Banerji Seal
sbs22ms076@iiserkol.ac.in

May 28, 2025

Examinee Details

Name:	Duytika Das
Examination Date (as per script):	May 2025
Original Paper Setter:	Shuvam Banerji Seal

Overall Performance Summary

The examination consisted of 70 multiple-choice questions. The marking scheme was +4 for each correct answer, -1 for each incorrect answer, and 0 for unattempted questions.

- Total Questions: 70
- Attempted Questions: 68
- Correct Answers: 58
- Incorrect Answers: 10
- Unattempted Answers: 2

Score Calculation:

- Marks from Correct Answers: $58 \times (+4) = 232$
- Marks from Incorrect Answers: $10 \times (-1) = -10$
- Marks from Unattempted Answers: $2 \times (0) = 0$

Total Score: $232 - 10 + 0 = \mathbf{222}/280$

Comparative Analysis of Answers

The following table shows a comparison of the student's answers with the correct answers.

Q#	Correct Option	Student's Option	Result	Score
1	b	b	Correct	+4
2	b	b	Correct	+4
3	b	c	Incorrect	-1
4	b	b	Correct	+4
5	c	c	Correct	+4
6	c	c	Correct	+4
7	b	b	Correct	+4
8	c	c	Correct	+4
9	b	b	Correct	+4
10	b	d	Incorrect	-1
11	b	b	Correct	+4
12	c	c	Correct	+4

Continued on next page

Table 1 – continued from previous page

Q#	Correct Option	Student's Option	Result	Score
13	b	b	Correct	+4
14	c	c	Correct	+4
15	b	c	Incorrect	-1
16	c	c	Correct	+4
17	b	b	Correct	+4
18	b	b	Correct	+4
19	c	d	Incorrect	-1
20	b	b	Correct	+4
21	c	c	Correct	+4
22	b	b	Correct	+4
23	b	b	Correct	+4
24	b	b	Correct	+4
25	b	c	Incorrect	-1
26	b	b	Correct	+4
27	a	a	Correct	+4
28	b	b	Correct	+4
29	c	c	Correct	+4
30	c	c	Correct	+4
31	a	a	Correct	+4
32	a	a	Correct	+4
33	a	a	Correct	+4
34	c	c	Correct	+4
35	c	c	Correct	+4
36	a	a	Correct	+4
37	d	d	Correct	+4
38	c	c	Correct	+4
39	b	b	Correct	+4
40	b	b	Correct	+4
41	c	..	Unattempted	0
42	c	c	Correct	+4
43	c	c	Correct	+4
44	c	c	Correct	+4
45	b	b	Correct	+4
46	b	b	Correct	+4
47	c	c	Correct	+4
48	b	c	Incorrect	-1
49	c	b	Incorrect	-1
50	c	c	Correct	+4
51	b	b	Correct	+4
52	d	d	Correct	+4
53	b	b	Correct	+4
54	a	a	Correct	+4
55	b	b	Correct	+4
56	b	b	Correct	+4
57	b	b	Correct	+4
58	b	..	Unattempted	0
59	b	b	Correct	+4
60	c	c	Correct	+4
61	b	b	Correct	+4
62	c	c	Correct	+4
63	b	b	Correct	+4
64	d	b	Incorrect	-1
65	a	a	Correct	+4
66	c	a	Incorrect	-1
67	b	b	Correct	+4
68	b	b	Correct	+4
69	b	b	Correct	+4
70	b	b	Correct	+4

Detailed Feedback on Incorrect Answers

Here’s an analysis of the questions answered incorrectly, with explanations to help clarify the concepts.

Question 3 (Linux Commands)

Question: What does the command `mkdir -p project/docs/notes` do?
Your Answer: (c) Prompts before creating each directory.
Correct Answer: (b) Creates all directories `project`, `docs`, and `notes` if they don’t exist.

Explanation: The `-p` (or `--parents`) option for the `mkdir` command is used to create parent directories as needed. If `project` or `project/docs` do not exist, `mkdir -p` will create them along with `notes`. It does not prompt for confirmation by default; the `-i` option (interactive) is used for prompting with commands like `rm` or `mv`, but it’s not a standard option for `mkdir` to prompt for creation. **Example:** If only `project/` exists, `mkdir -p project/docs/notes` will create `docs/` inside `project/` and then `notes/` inside `project/docs/`. If none of them exist, it creates all three.

Question 10 (Linux Commands)

Question: To find lines containing the exact word ”error” (not ”errors” or ”terror”) in `log.txt`, which `grep` command is most appropriate?
Your Answer: (d) `grep -v "error" log.txt`
Correct Answer: (b) `grep -w "error" log.txt` **Explanation:**

- `grep -w "pattern" file`: This option tells `grep` to select only those lines containing matches that form whole words. So, ”error” would match, but ”errors” or ”terror” would not.
- `grep -v "pattern" file`: This option (invert match) selects lines that *do not* contain the pattern. Your choice would have shown lines *without* ”error”.

Example: If `log.txt` contains:

```
This is an error.
This line has errors.
No terror here.
```

`grep -w "error" log.txt` would output: ‘This is an error.‘
`grep -v "error" log.txt` would output: ‘No terror here.‘

Question 15 (Linux Commands)

Question: If you try to remove a directory with `rm mydir` and it’s not empty, what happens?
Your Answer: (c) You are prompted to confirm removal for each file.
Correct Answer: (b) An error message is shown, and the directory is not removed.
Explanation: By default, the `rm` command cannot remove directories. To remove a directory, you typically use `rmdir` (for empty directories) or `rm -r` (or `-R`, for recursive removal, which removes a directory and its contents). If you use `rm mydir` on a non-empty directory, it will output an error like ”`rm: cannot remove ’mydir’: Is a directory`” and will not remove it or prompt for file removal. Prompting usually happens with the `-i` (interactive) flag.

Question 19 (Linux Commands)

Question: Which command would you use to find all files named `config.ini` within your home directory and its subdirectories?
Your Answer: (d) `locate config.ini --home`
Correct Answer: (c) `find ~name "config.ini"` **Explanation:**

- `find ~name "config.ini"`: This is the correct command. `find` searches the directory tree in real-time. ~ represents the home directory, `-name "config.ini"` specifies searching for files with that exact name.

- `locate config.ini`: This command searches a pre-built database (usually updated by `updatedb`). While fast, it might not reflect very recent changes. The `--home` option is not a standard option for `locate`. Even if it were, `find` is more precise for targeted searches within a specific path and its subdirectories.

Question 25 (Linux Commands)

Question: Which command is used to change the current directory to the user's home directory?
Your Answer: (c) `cd ..`
Correct Answer: (b) `cd ~` or just `cd` **Explanation:**

- `cd ~` or simply `cd` (with no arguments) will change the current directory to your home directory. The tilde (~) is a shortcut for the current user's home directory path.
- `cd ..` changes the directory to the parent directory (one level up from the current directory).

Question 48 (Python Fundamentals)

Question: What will be the output of the following Python code?

```
my_set = {1, 2, 3, 3, 2}
print(len(my_set))
```

Your Answer: (c) 5
Correct Answer: (b) 3
Explanation: Python `set`s are unordered collections of *unique* elements. When you create `my_set = {1, 2, 3, 3, 2}`, the duplicate values (3 and 2) are automatically ignored. Therefore, `my_set` will effectively be `{1, 2, 3}`. The `len()` function returns the number of elements in the set, which is 3.

Question 49 (Python Fundamentals)

Question: Consider the code: `my_dict = {'a': 1, 'b': 2, 'c': 3}`. What does `my_dict.get('d', 0)` return?
Your Answer: (b) `None`
Correct Answer: (c) `0` **Explanation:** The `get()` method for dictionaries takes two arguments: the key to look for, and an optional default value.

- If the key is found, its value is returned.
- If the key is *not* found and a default value is provided, the default value is returned.
- If the key is not found and no default value is provided, then `None` is returned.

In this case, the key `'d'` is not in `my_dict`, and the default value provided is `0`. So, `0` is returned. If the code was `my_dict.get('d')`, then `None` would have been the answer.

Question 64 (Python Fundamentals)

Question: How are key-value pairs stored in Python?
Your Answer: (b) Set (Note: Your script wrote "Dictionary" next to "b)", but option (b) in the original question paper was "Set". The correct option for "Dictionary" was (d).)
Correct Answer: (d) Dictionary **Explanation:** Python uses **dictionaries** (type `dict`) to store data as key-value pairs.

- **Lists** store ordered sequences of items.
- **Sets** store unordered collections of unique items.
- **Tuples** store ordered, immutable sequences of items.
- **Dictionaries** store mappings from keys to values. Example: `{'name': 'Alice', 'age': 30}`

It seems there might have been a mix-up in matching your intended answer "Dictionary" with the option letter.

Question 66 (Python Fundamentals)

Question: Which of these functions from the `functools` module is commonly associated with functional programming for accumulating results?
Your Answer: (a) `map`
Correct Answer: (c) `reduce` **Explanation:**

- `map(function, iterable, ...)`: Applies `function` to every item of `iterable` and returns an iterator of the results. It doesn't accumulate.
- `filter(function, iterable)`: Constructs an iterator from elements of `iterable` for which `function` returns true. It doesn't accumulate.
- `reduce(function, iterable[, initializer])` (from `functools` module): Applies `function` of two arguments cumulatively to the items of `iterable`, from left to right, so as to reduce the iterable to a single value. This is the function for accumulation.

Example of `reduce`:

```
from functools import reduce
import operator
numbers = [1, 2, 3, 4]
sum_of_numbers = reduce(operator.add, numbers) # 1+2=3, 3+3=6, 6+4=10
print(sum_of_numbers) # Output: 10
```

Concluding Remarks & Encouragement

Dear Duytika,

You have put in a commendable effort in this examination, achieving a score of **222 out of 280**. This is a very good performance and demonstrates a solid grasp of the core concepts in Linux commands, Number Systems, and Python Fundamentals. Your understanding of many specific commands like `cp, mv, ls, sudo, pacman` and Python data structures like lists, tuples, and sets, along with control flow, is evident.

The few areas where points were lost often involved nuanced differences in command options (like `grep -w` vs. `-v`, or the behavior of `mkdir -p`) or specific Python method behaviors (like `dict.get()` with a default value, or the properties of sets). These are excellent learning opportunities! Reviewing these specific cases will further solidify your understanding.

Keep up the excellent work! Every challenge faced is a chance to grow stronger and more knowledgeable. Your dedication is clear, and with continued practice and attention to these finer details, you will undoubtedly master these topics even more thoroughly.

Here's a little something to keep you inspired:

> *"Believe in yourself. Not in the you who believes in me. Not the me who believes in you. Believe in the you who believes in yourself!"*
> – Kamina (Tengen Toppa Gurren Lagann)

Continue to learn, explore, and challenge yourself. Your progress is impressive!
Best regards,
Shuvam Banerji Seal