Markdown Quick Reference (for LLM Outputs)

Markdown is a lightweight way to format plain text.
You write simple symbols (like #, *, and -) and the platform renders clean formatting (headings, lists, code blocks, links, etc.).

Why this matters in this course:

- LLMs often return answers in Markdown.
- Knowing Markdown helps you quickly read, edit, and reuse model output.
- Markdown is used in GitHub, Jupyter notebooks, docs, and many chat tools.

---

1) Headings

Use # symbols at the start of a line.

```
# Heading 1
## Heading 2
### Heading 3
```

Rendered:

Heading 1

Heading 2

Heading 3

Tip: Most documents use one `# Heading 1` title, then ## for major sections.

---

2) Paragraphs and Line Breaks

- A blank line creates a new paragraph.
- A single Enter usually stays in the same paragraph.
- Two trailing spaces at the end of a line can force a line break in many renderers.

```
This is paragraph one.

This is paragraph two.
```

---

3) Emphasis (Bold / Italic)

```
*italic* or _italic_
**bold** or __bold__
***bold italic***
```

Rendered examples:

- *italic*
- **bold**
- ***bold italic***

---

4) Lists

Unordered (bulleted)

```
- Item A
- Item B
  - Sub-item B1
  - Sub-item B2
```

Ordered (numbered)

```
1. First
2. Second
3. Third
```

Tip: Indent nested list items with two spaces (or four, depending on style).

---

5) Links

```
[OpenAI](https://openai.com)
[Course README](../README.md)
```

Format: `[link text](URL-or-path)`

---

6) Images

```
![Alt text describing image](images/example.png)
```

Format: `![alt text](path-or-url)`

Why alt text matters:

- Accessibility (screen readers)
- Context when image fails to load

---

7) Code Formatting

Inline code

Use backticks around short code references:

```
Use `pip install -r requirements.txt` to install dependencies.
```

Code blocks

Use triple backticks for multi-line code. Add language name for syntax highlighting.

````
```python
def greet(name):
    return f"Hello, {name}!"
```
````

```python
def greet(name):
    return f"Hello, {name}!"
```

---

8) Blockquotes

```
> This is a quoted note.
> You can continue on multiple lines.
```

Rendered:

> This is a quoted note. You can continue on multiple lines.

Useful for tips, warnings, or excerpts.

---

9) Horizontal Rules

Use three dashes (or ***) on a line by itself:

```
---
```

---

10) Tables (Basic)

```
| Tool | Use Case | Week |
|------|----------|------|
| Chat model | Q&A | 2 |
| Notebook | Coding exercises | 2 |
| Markdown | Documentation | 2 |
```

Rendered:

Tool	Use Case	Week
Chat model	Q&A	2
Notebook	Coding exercises	2
Markdown	Documentation	2

---

11) Task Lists

```
- [x] Install dependencies
- [ ] Run notebook cells
- [ ] Submit assignment
```

Rendered (on platforms that support task lists):

- [x] Install dependencies
- [ ] Run notebook cells
- [ ] Submit assignment

---

12) Escaping Special Characters

If you need to show Markdown symbols literally, add a backslash (`\`).

```
\*not italic\*
\# not a heading
```

Common “LLM Output” Patterns You’ll See

1. **Headings + bullets** for organized summaries
2. **Code fences** with language labels (like `python`, `bash`, `json`)
3. **Tables** for comparisons
4. **Blockquotes** for notes or caveats

When in doubt:

- Copy the output into a Markdown preview (GitHub, VS Code preview, or notebook Markdown cell).
- Look for simple patterns: `#`, `-`, `**`, backticks, `[text](link)`.

Mini Practice Exercise

Try to read and then edit this Markdown snippet:

````
## Week 2 Summary

In this week, we practiced:
- Prompting a model
- Interpreting **Markdown** output
- Running `python` code in notebooks

> Tip: Ask the model to “format your response as Markdown.”

### Useful command
```bash
pip install -r requirements.txt
```
````

Questions for students:

1. Which text is bold?
2. Where is the blockquote?
3. Which part is a code block?
4. Add one new bullet item.

One-Line Cheat Sheet

- `#` heading
- `*italic*` and `**bold**`
- `-` bullet list
- `1.` numbered list
- `` `code` `` inline code
- ```
  ```
- `[text](url)` link
  ```

- `![alt](img.png)` image
- `>` quote
- `---` divider

If you can recognize these symbols, you can read most Markdown generated by an LLM.