

Comprehensive Python course for AI

Final Project(Python Basic)

Deadline: 2023 23 November

Score: 1000 + 400

2023 09 November

Comprehensive Python course for AI

Python CLI Tool for File Manipulation

Score: 1000 + 400

Python CLI Tool for File Manipulation

- ❖ In this project, you will develop a Python Command Line Interface (CLI) tool that facilitates file manipulation and directory navigation, similar to command-line functionalities in Unix-like operating systems. A key component of your tool will be an advanced logs system that tracks each command's usage.

Python CLI Tool for File Manipulation

❖ You are required to implement the following commands in your CLI tool:

1. `ls [path]` — List directory contents at ``path``, or the current directory if no ``path`` is given.
2. `cd [path]` — Change the working directory to ``path``.
3. `mkdir [path]` — Create a new directory at ``path``.
4. `rmdir [path]` — Remove the directory at ``path`` if it is empty.
5. `rm [file]` — Remove the file specified by ``file``.
6. `rm -r [directory]` — Remove the directory at ``directory`` and its contents recursively.
7. `cp [source] [destination]` — Copy a file or directory from ``source`` to ``destination``.
8. `mv [source] [destination]` — Move a file or directory from ``source`` to ``destination``.
9. `find [path] [pattern]` — Search for files or directories matching ``pattern`` starting from ``path``.
10. `cat [file]` — Output the contents of the file ``file``.

Python CLI Tool for File Manipulation

❖ Logs file:

- ✓ Log each command, its arguments, the time of execution, and the outcome (success or error).
- ✓ The log file should be human-readable and well-organized for ease of analysis and debugging.

Points:

- ✓ Use the `argparse` module to parse command-line arguments.
- ✓ Each command should be a function.
- ✓ Handle exceptions and provide clear error messages.
- ✓ write comment for your code.
- ✓ Prepare a `README.md` file that documents the setup of the project

Python CLI Tool for File Manipulation

+400

❖ Team working and Challenges (For Extra Scores):

1. working as a team (2 or 3):

- ✓ Use Git and GitHub for version control.
- ✓ Commit changes regularly with descriptive messages.
- ✓ Use GitHub Issues and Projects for task management.
- ✓ Explore GitHub Actions for continuous integration and automatic unit testing

2. Add additional commands for more complex file manipulations or system information retrieval.

3. Create a function that allows for content search within files.

Best of luck, we look forward to seeing your solutions 😊

Thanks

Good Luck!

Don't give up on your dreams 😊