Package Statistics Command-Line Tool

Interview Test Submission Report

Daksh Maheshwari

May 6, 2025

Overview

This report details the design, implementation, and testing of the `package_statistics.py` command-line tool. It downloads Debian ‘Contents-<arch>.gz‘ indices, parses file-to-package mappings, aggregates file counts per package, and presents the top-N packages in a formatted table.

Key Features

- **Configurable:** Customize architecture, mirror, distribution, component, and top-N via CLI arguments.
- **Robust Parsing:** Handles large indices, paths with spaces, and skips malformed/header lines.
- **Error Handling:** Gracefully handles network errors and broken pipes (e.g., piping to ‘head‘).
- **Clear Output:** Presents results in a formatted table with dynamic column widths.
- **Quality Assurance:** Type-annotated (Python 3.10+), formatted (`black`), linted (`ruff`), and type-checked (`mypy`).

Implementation Details

Modular functions handle specific tasks:

`download_contents:` Fetches the index file using `urllib.request`, handling HTTP/URL errors.

`parse_contents:` Iterates decompressed lines, uses `str.rsplit(None, 1)` to separate path/packages, extracts base package names.

`get_top_packages:` Orchestrates download, decompression (`gzip`), text wrapping (`TextIOWrapper`), parsing, and counting (`Counter`).

`main:` Handles CLI arguments (`argparse`), logging, calls core logic, formats output table, and manages `BrokenPipeError`.

Testing & Coverage

Comprehensive `pytest` suite (`test_package_statistics.py`) covers parsing logic, download simulation (mocking), aggregation, and CLI behavior. Achieves **93% line coverage** (via `pytest-cov`).

Tools & Configuration

Project automation and configuration are centralized for consistency:

- **Makefile:** Targets for install (`requirements-dev.txt`), format (`black`, `ruff -fix`), lint, typecheck, test, check, and clean.
- **pyproject.toml:** Unified configuration for `black`, `ruff` (lint), `isort`, `mypy`, and `pytest`.
- **requirements-dev.txt:** Defines dev dependencies: `black`, `ruff`, `mypy`, `pytest`, and `pytest-cov`.
- **.gitignore:** Excludes Python caches, coverage reports, virtual environments, OS files, and LaTeX auxiliary outputs.

Time Log

Task	**Time Spent**
Requirements Review	0h 20m
Initial Coding	0h 45m
Parsing Logic Refinement	0h 20m
CLI, Error Handling, Pipe Support	0h 30m
Unit Tests Development	2h 15m
Test Coverage Improvements	0h 20m
Documentation (README, Report)	0h 30m
Makefile & Build Enhancements	0h 15m
Output Formatting	0h 25m
Final Debugging & Cleanup	0h 20m
Total Estimated Time	**6h**