pckstat Command Line Tool

This is a command line tool that allows users to query statistics of packages obtained from a Debian mirror. The program downloads the compressed "Contents" file associated with a given architecture, parses the file, and outputs the statistics of the top 10 packages that have the most files associated with them.

Time spent in the development of this command line tool is roughly 4 hours.

Development of this command line tool would include:

1. Understanding the problem:
    a. Learning about the structure of the Debian mirror and the compressed "Contents" file
2. Developing functions to be able to:
    a. download
    b. decompress
    c. parse

    the "Contents" file from the Debian mirror.
3. Developing the command line tool
4. Using pylint to check for conformity of Python's coding best practices
5. Doing up the documentation including the README.md file and this report

Understanding the problem:
Based on the description provided as well as the documentation for the contents indices, The general idea of the task is to download and parse the "Contents" file associated with the architecture specified as a command line argument.
However, the "udeb" package type is used by the Debian Installer, which is a program that helps install Debian on a computer. The udeb packages contain only the files needed by the installer, while the regular packages contain all the files needed to run a program. Based on this, I determined that this task is not asking me to analyse contents for Debian Installers. Therefore, I would only need to download files like “Contents-amd64.gz” and not “Contents-udeb-amd64.gz”.

Developing functions:
I decided to have a separate file called “api.py” to have the main functions that downloads, decompresses and analyses the statistics for the "Contents" file as this would have a layer of abstraction between the actual command line logic and the core functions. This would help with further development of the command line tool in the future.

Developing the command line tool:
I have decided to use Python’s “argparse” library to build a command line tool to be able to take advantage of the argument parsing functionality. This could later be used to easily develop other features by adding different arguments or subcommands if there is a need to. I have also made

this project installable by “pip” (More details of the installation steps in the README.md file). This would make the project an actual command line tool without having to have to run “python3” in front of the python file to call the program.

Using pylint to check for conformity:
I used pylint to check for the conformity of the best practices for Python. This command line tool is the one that is recommended to me online and is able to pick up inconsistent indentations and unconventional naming practices.