LINUX PROGRAMMING ASSIGNMENT 10

NAME: V S SAIDATTA USN: ENG24CY0175 ROLL NO: 28 SECTION: 3C

1. brew - History & background

Homebrew was created by Max Howell in 2009 to simplify installing open-source software on macOS. Over time, it became the dominant community package manager for macOS. Linux support was developed and later merged back into Homebrew, providing official support for Linux and WSL.

Key milestones (summary):

- - 2009: Homebrew initial development and release.
- - 2013: Project growth, Kickstarter support for infrastructure.
- - 2019: Linux brew merged back into Homebrew and became Homebrew on Linux (official support).

Homebrew continues to be maintained by a team of contributors, and the project's documentation is the authoritative source for feature and behavior changes.

2. brew

Homebrew, commonly known as brew, is a free open-source package manager originally for macOS, that offers a simple way to install, update, and manage command-line software. On Linux it has commonly been referred to as either Linux brew, or Homebrew on Linux.

Homebrew simplifies installing and managing packages that are not available, not up to date, or not super easy to install with a distribution of native package managers. Homebrew will install packages into its own prefix, allowing installations of software without requiring root permissions.

Homebrew users drive installation of packages through the brew command. Packages have a definition, called a formula, that provides brew with command line instructions to build and install the software for the user, as opposed to the pre-compiled binaries, called bottles. Workflow for homebrew is typically to begin by installing homebrew, run brew install <package>, then after some time run brew update, and brew upgrade to refresh and upgrade packages.

3. brew - Command output details

Common brew outputs, and what they mean:

- brew –version: displays Homebrew client version and Ruby/other components version information.
- brew install: displays download/build progress, installation process for the formula, keg-only messages, caveats, and installation instruction (from a bottle, or from source).
- brew info: displays detailed formula metadata (version, dependencies, install prefix, installed files, patches, and caveats).
- brew list: lists installed formulae and casks.
- brew doctor: lists warnings and suggestions for fixing installation problems.

Note: the specific output text and formatting may differ across versions of Homebrew and/or across platforms.

4. brew - Option flags

- brew -install: to install a package.
- brew -uninstall or brew remove uninstall a package.
- brew -update: update Homebrew and taps.
- brew -upgrade [package]: upgrade installed packages (or one package).
- brew -search: search packages / taps.
- brew -info: get information about a package.
- brew -list: lists installed packages.
- brew -tap and brew untap: add / remove additional formulae smart taps.
- brew -cleanup: remove old versions of installed packages, packets etc and free space on that project.

Some useful flags and options:

- --verbose: print the extra output when running commands.
- --debug: Run with debug output / drop into a shell even if it fails (useful for debugging builds).
- --force-bottle: prefer precompiled binaries ("bottles") instead of source builds, if possible.
- --build-from-source: force a package system package to be built from source instead of precompiled binaries.
- --formula / --cask limits this command to formulae or casks (when applicable).

1. sleep - History & background

The sleep utility and the underlying sleep() system/library function are longstanding Unix features. Authors of modern GNU implementations include contributors such as Jim Meyering and Paul Eggert. The sleep command is standardized by POSIX and implemented across Unix-like systems.

2. sleep

Sleep is a standardized POSIX command that halts the execution for a specified amount of time. It is provided as both a utility shell and a function in the C library to sleep.

Sleep is employed to introduce delayed execution into scripts and command sequences for instance; you might poll after a service has been waiting to start, rate limit retries or put delays in-between actions.

Usage examples (shell): sleep 5 will wait for 5 seconds. Sleep 1m will wait for 1 minute. It understands numbers and typical suffixes (s, m, h, d) and fraction seconds, or some other argument that sums up to a delay of 1.5 seconds.

3. sleep - Command output details

Behavior and outputs:

- -sleep: does not produce any stout when run in regular mode, it will simply suspend the caller for the requested period.
- -it returns exit code 0 when sleep completes normally, but if interrupted by a signal, exit codes may change to indicate failure.
- -sleep --help will print the usage box and return.
- -sleep --version will print version information.

4. sleep - Option flags

Common options that may appear in a variety of implementations:

Multiple positional arguments: sleep 0.5 will sleep for the sum of each positional argument.

- --help: Show help and exit.
- --version: Show version and exit.