which.sh

The script begins by defining a usage message stored in the variable USAGE.

It then initializes a boolean variable SHOW_ALL to control whether to show all matches or just the first one found.

Option parsing is done using the built-in getopts function, allowing the script to handle options like -a to display all matches.

Error handling for unknown options is also implemented within the script.

A function named `find_in_path` is defined to check if a file is both a regular file and executable. This function is crucial for determining whether a given command exists in the system's PATH.

The script then shifts past the options and checks if any arguments are provided. If not, it prints the usage message and exits with an error code.

If the PATH variable is empty, the script sets it to a default list of directories commonly found in Unix-like systems.

The script then iterates through each argument provided to search for the corresponding executable file. For each argument, it first checks if it starts with a '/', which would indicate an absolute path. If not, it iterates through each directory in the PATH variable to construct the full path to the command.

Error handling is implemented to display a message if a command is not found. If all provided commands are not found, the script exits with code 2.

It should be noted that certain operations, such as string manipulation and path traversal, are achieved using built-in shell features. While some external commands like echo and cut are used, these are common utilities typically available in Unix-like environments.