FIT2100 Assignment 1

Aidan Taylor-Lynch
24205230

User documentation **Command Line Interpreter**

Welcome and thank you for reading! This guide is meant to provide specific information to help provide an in-depth understanding of how to use this command line interpreter.

Basic Command Line Interpreter

The basic command line interpreter (CLI), provides a user interface to the operating system. Users enter commands on a single line followed by the “return/enter” key in order to execute it. The basic command line interpreter functions very similarly to shells found in Linux/Unix environments. The basic command line interpreter provides the following commands:

clear, cd, ls, echo, help and quit. See this section for the correct usage.

Setup - Basic CLI

To **setup** the Basic CLI, open a terminal and navigate to the directory in which the files were extracted. Once inside the directory, type “make” and press “return/enter” to build the executable files. To **run** the Basic CLI, simple type “./task1” in the same directory.

Commands - Basic CLI

A list of the Basic CLI commands and their description are listed below. Please follow the guide to avoid incorrect program behaviour.

clear

Function: The **clear** command will clear all prior output from the terminal, leaving a blank screen.
Assumptions/Notes: This command will work from any directory.
Correct Usage:

```
%aidantaylor-lynch$ clear
```

cd <directory>

Function: The **cd <directory>** command will change the default directory to the directory specified by <directory> argument. If no directory is specified, the default directory will be changed to the path given by the **HOME** environment variable in UNIX.
Assumptions/Notes: If ".." is entered as the <directory> argument, the default directory will be changed to the directory up one level.
Correct Usage:

```
%aidantaylor-lynch$ cd /Users
```

ls <directory>

Function: The **ls <directory>** command will list the contents of the directory specified by the <directory argument>. If no directory is specified, the contents of the current working directory will be listed.
Assumptions/Notes: Note that in the output, "." stands for current directory, and ".." stands for one level up. These can be ignored.
Correct Usage:

```
%aidantaylor-lynch$ ls /Users
.
..
.localized
aidantaylor-lynch
Shared
aidantaylor-lynch$
```

echo <statement>

Function: The **echo <statement>** can be used to print a sequence of characters specified by the <statement> argument, onto the screen.
Assumptions/Notes: The <statement> argument must be enclosed with double quotation marks, or echo will output and error.

Correct Usage:

```
%aidantaylor-lynch$ echo "Hello World!"
Hello World!
aidantaylor-lynch$ $
```

help

Function: The **help** command will display the user manual for this Basic CLI.
Assumptions/Notes: This command will only work if the user is in the directory associated with the Basic CLI program. Nothing will be displayed otherwise.
Correct Usage:

```
%aidantaylor-lynch$ help
```

quit

Function: The **quit** command exits the command line interpreter.
Assumptions/Notes: This command works from any directory.
Correct Usage:

```
%aidantaylor-lynch$ quit
```

Advance Command Line Interpreter

The advanced command line interpreter (CLI), provides a more powerful user interface to the operating system. Users enter commands on a single line followed by the “return/enter” key in order to execute it. The advanced command line interpreter functions very similarly to shells found in Linux/Unix environments. The advanced command line interpreter provides the same commands and usage as the Basic CLI, with the addition of the following commands:
Create, copy, search, run and halt. See this section for the correct usage.

Setup - Advanced CLI

To **setup** the Advanced CLI, open a terminal and navigate to the directory in which the files were extracted. Once inside the directory, type “make” and press “return/enter” to build the executable files. To **run** the Basic CLI, simple type “./task2” in the same directory.

Commands - Advanced CLI

A list of the Advanced CLI commands and their description are listed below. Please follow the guide to avoid incorrect program behaviour. **NOTE** The advanced command line interpreter supports all the commands that the Basic command line interpreter supports.

create <file>

Function: The **create <file>** command will create a file with the name specified by the <file> argument.
Assumptions/Notes: This command will not create a new file if <file> already exists. For good practice, specifying the

file extension is recommended, but not necessary.

Correct Usage:

```
%aidantaylor-lynch$ create myFile.txt
```

copy <old> <new>

Function: The **copy <old> <new>** command will copy the contents of the file specified by the <old> argument, and create a new file specified by the <new> argument, appending the contents of the old file to it.

Assumptions/Notes: The <old> argument must specify a file extension, and will not work if the file does not exist. The command will fail if the file specified by the <new> argument already exists, however no extension is necessary.

Correct Usage:

```
%aidantaylor-lynch$ copy help.txt myCopy.txt
```

search <pattern> <file>

Function: The **search <pattern> <file>** command will display the number of occurrences of the <pattern> argument, in the file specified by the <file> argument.

Assumptions/Notes: The <pattern> argument must take the format of a sequence of characters that does not contain any whitespace or new lines. If there is whitespace, the command will return an error. The file specified by <file> must include the file extension.

Correct Usage:

```
%aidantaylor-lynch$ search usage help.txt
the functionality and correct usage of this program.
sample usage: USER$ clear
sample usage: USER$ cd /Users
sample usage: USER$ cd
sample usage: USER$ ls /Desktop
sample usage: USER$ ls
sample usage: USER$ echo "Hello world!"
sample usage: USER$ help
sample usage: USER$ quit
sample usage: USER$ create myFile.txt
sample usage: USER$ copy help.txt myCopiedFile.txt
sample usage: USER$ search "sample usage" help.txt
**WARNING** This may cause unknown side effects as this usage is not supported. **WARNING**
sample usage: USER$ run ls
sample usage: USER$ halt Mail
aidantaylor-lynch$
```

run <program>

Function: The **run <program>** command will execute an executable program named <program>.

Assumptions/Notes: This command supports Unix system programs such as “ls, pwd, pkill, etc”, and does not require a prefix to run them. However to run a normal executable file, the prefix “./” must be entered in the <program> argument or the program will not be found.

Correct Usage:

```
%aidantaylor-lynch$ run ./myCExecutable
```

```
%aidantaylor-lynch$ run ls
```

halt <program>

Function: The **halt <program>** command will terminate the execution of a program specified by the <program> argument.

Assumptions/Notes: The <program> is case sensitive and must be identical to the target.

Correct Usage:

```
%aidantaylor-lynch$ halt Mail
```