# NSSA221 - System Administration I

# Scripting Assignment 03 – Symbolic Links

**The Basics:**

In Linux, symbolic links (similar to Windows shortcuts) are used for easy file access and managing different library versions. This scripting exercise focuses on teaching you how to find and create symbolic links, including identifying their target paths, using both Bash and Python commands within your Python script.

Example Commands:

Bash: **ln -s**

Python: **os.symlink**

Use the man pages and Python documentation to find the syntax for these commands.

**Script Requirements:**

* Language: Python 3
* Filename: shortcut.py
* Environment: Will be run and tested on the Rocky 8 VM in RLES. (It needs to work here; we don't care if it works on your laptop.)
* Points awarded will be based on the criteria in *“Table 1 – Script Grading Rubric.”*

**Functional Requirements:**

* Create a symbolic link.
* Delete a symbolic link.
* Run a report summarizing the symbolic links on the user's desktop.
* The script should not hard code paths; instead, it should be applicable for any user logged in. Use environment variables like $HOME or Python's path.home() for this.

**Additional Information:**

The script should simplify the process of creating symbolic links so that the end-user does not need to bother with Linux commands. Create a menu interface with options corresponding to the functional requirements above.

**Script Demo:**

To view the script in action, run the script command.

**scriptreplay –-timing=time.log shortcut.scr**

**Table 1** – Script Grading Rubric

|  |  |  |
| --- | --- | --- |
| Requirements | Points | Points Earned |
| Documentation | **15** |  |
| The script contains the shebang! | 2 |  |
| The script has the correct permissions set. | 2 |  |
| The script is commented with the student’s name and date. | 2 |  |
| The script is titled “*shortcut.py*”. | 2 |  |
| The script is sufficiently commented. | 5 |  |
| Written in Pythonic style. | 2 |  |
| User Experience | **15** |  |
| The terminal is cleared when the script is run. | 2 |  |
| The user is informed of their current working directory. | 5 |  |
| The menu is designed for end-user readability. | 5 |  |
| The script exits gracefully by typing “*quit*.” | 3 |  |
| Core Functionality | **30** |  |
| The user is prompted to enter the file to create the shortcut for. This file can be located anywhere on the system, so you want to search from the root (/). | 5 |  |
| The script checks to see if the file exits, and the user is informed if it does not (i.e., Error handling for file existence.) | 5 |  |
| Allows for symbolic link creation. Suggested commands, find, readlink, and ln. | 5 |  |
| The symbolic link(s) are only created on the user’s Desktop. | 5 |  |
| The menu includes the option to delete a symbolic link. | 5 |  |
| The script uses the subprocess, os, or other applicable modules. | 5 |  |
| Summary Report | **20** |  |
| The summary report lists all the symbolic links in the user’s Desktop, with the associated target path. | 10 |  |
| The summary report shows the number of links in the user’s home directory. | 5 |  |
| The report is a user-friendly, readable format. | 5 |  |
| Code Quality | **10** |  |
| Error Handling | 5 |  |
| The code is modular and well-organized. | 5 |  |
| Execution | **10** |  |
| The script runs with no errors. | 5 |  |
| The script is fully functional and runs as expected. | 5 |  |
| Final Grade | |  |