**Assignment 13: Script**

Hello, I am Joshua Randle and this is my submission for portfolio project.

So this is my work lunch roulette program. It is a command line interface and the user is able to interact with it by entering the prompts requested at each step of the program.

For example, to continue from the home page you would either enter “fast” to be taken right to your random choice or “slow” to be taken through every step of the program. This also covers the first cognitive style heuristic of explaining the benefits of features to the user.

**Demonstrate fast function**

It is also explained in the main page the cost of using the fast method over the slow method would be that no user data could be recorded for future use of the program.

The option to take a quick answer over a walkthrough of the program also satisfies the third cognitive style heuristic, letting people gather as much information as they want and no more than they want. This is also satisfied later in the program giving the user the option to either use saved data or start a new roulette file.

Cognitive style heuristic 4 is satisfied by staying consistent with input text throughout the program. For example, the back function is always enabled by entering ‘back’ and the roulette option is always enable by entering ‘engage’.

By enabling a back function on every step of the program as seen this satisfies cognitive style heuristic 5, making an undo/redo option.

Cognitive 6, providing an explicit path through the task, is satisfied by giving three simple directions; an instant result, previous user data, and a fresh data run. All three have their own specific path.

Cognitive 7, providing ways to try out different approaches is also satisfied in this way. There are three main options with the potential for sub-options if you are a returning user as seen by this option to retain current user data or reset and start over (demonstrate).

Cognitive 8, encouraging tinkerers to tinker mindfully, is seen as well. As you can see from the start screen a warning for the fast method is provided which lets the user know data will not be saved if that method is chosen. Also, when current users choose the reset option they are taken to a second verification before that reset is allowed to happen.

Learnability is reflected in the program by the amount of detail users have access to for each operation in the program. For instance, the options for a slow or fast process and also the options for new and existing users are detailed and clear so the user knows what each option will give them. In the future they may not even have to read the prompts they will just know the process they are wanting to perform at that time.

**Demonstrate slow function**

**Demonstrate new user**

**Demonstrate existing user**

**Demonstrate how microservice updates**

A couple of other features that I have added is the full implementation of the mulligan function as well as an add/remove function which is shown here.

That is what I have for my portfolio project. Thank you!