Shell Scripting - Hello World Demo

Show we can execute the script by typing `bash`

```
bash Desktop/CS-Foundations/lectures/week6/hello_world
```

This is essentially the same as when we say `python script.py`. We are passing the `hello_world` script (which is just a text file) to the `bash` interpreter.

Running the script without adding to $PATH or making executable

Now what happens if we try to run `./hello_world`? If we don't have the right path, we get an error saying `no such file or directory: ./hello_world`. We can change directories to ensure we have the right path: `cd ~/Desktop/CS-Foundations/lecture/week6/`. If we have the right path, we still get an error saying `permission denied: ./hello_world`.

Make the script executable

We showed how we can make the script executable using the octal method: `chmod 755 hello_world`, but we can also add executable permissions to all other permissions in all other user groups with: `chmod +x hello_world`. Now if I try to execute the command with `./hello_world`, we see that it executes without error and prints `Hello World!` to the console. However, note that we get a `command not found: hello_world` error if we try to run `hello_world`.

Adding the script to your $PATH

If I type `echo $PATH` to see what directories are in my `$PATH` variable, I can move the script to one of the directories I know I have access to:

```
mv hello_world /usr/local/bin
```

Now, no matter what directory I'm in, I can simply type `hello_world` and the script runs without error. Note now if I type `which hello_world`, it tells me `/usr/local/bin/hello_world`.

For loops

all on one line

```
for i in A,B,C,D; do echo $i; done
```

parameter expansion

```
for i in {A..D}; do echo $i; done
```

multiple lines with do on first

```
for i in {A..D}; do
echo $i
done
```

all multiple lines

```
for i in {A..D}
do
    echo $i
done
```