We're one step away from being able to use our scripts and functions as shell commands, but first we need to learn about environmental variables. An environmental variable is a variable that Bash creates where data about your current computing environment is stored. Environmental variable names use all capitalized letters. Let's look at the values for some of these variables. The HOME variable contains the path to our home directory, and the PWD variable contains the path to our current directory.

```
echo $HOME
echo $PWD

## /Users/sean
## /Users/sean/Code

```

If we want one of our functions to be available always as a command then we need to change the PATH variable. Let's take a look at this variable first.

```
echo $PATH

## /usr/local/bin:/usr/bin:/bin:/usr/local/git/bin

```

The PATH variable contains a sequence of paths on our computer separated by colons. When the shell starts it searches these paths for executable files, and then makes those executable commands available in our shell. One approach to making our scripts available is to add a directory to the PATH. Bash scripts in the directory that are executable can be used as commands. We need to modify PATH every time we start a shell, so we can ammend our ~/.bash_profileso that our directory for executable scripts is always in the PATH. To modify an environmental variable we need to use the export keyword.

First let's create a new directory called Commands in our Codedirectory where we can keep our executable scripts. Then we'll add a line to our ~/.bash_profile so that Commands is added to the PATH.

```
mkdir Commands
nano ~/.bash_profile

alias docs='cd ~/Documents'
alias edbp='nano ~/.bash_profile'

export PATH=~/Code/Commands:$PATH
```

Save ~/.bash_profile and close nano. Now let's source our Bash profile (we only need to do this once) and move short into the Commands directory. Then we should be able to use short as a command!

```
source ~/.bash_profile
short

## a small program

```

Looks like it works!

Alternatively to making individual scripts executable we can add a source command to our ~/.bash_profile so that we can use a Bash function on the command line. Let's use nano to open up our ~/.bash_profile again.

```
nano ~/.bash_profile

alias docs='cd ~/Documents'
alias edbp='nano ~/.bash_profile'

export PATH=~/Code/Commands:$PATH
source ~/Code/addseq2.sh

```

Save the ~/.bash_profile, quit nano, and now let's source our ~/.bash_profile so we can test if we can use addseq2.

```
source ~/.bash_profile
addseq2 9 8 7

## 24

```

Again it works! If you have multiple Bash functions that you'd like to be able to use on the command line then it's a good idea to define these functions in one of a few files so that you don't have to source every individual function that you want to have available.

Summary

- According to the Unix Philosophy you should keep your programs short, simple, and quiet.
- Use chmod to make your programs executable.
- You can modify your ~/.bash_profile in order to make scripts and functions available to use on the command line.
- Use export to change an environmental variable.

Mark as completed