# BASH Lab 3 Variables

First, do the exercise in slide 4 of the Cyber Aces Bash variables lesson. To save typing, you can use this for the script:

#!/bin/bash

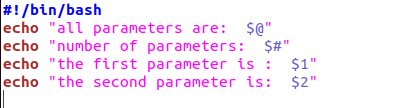
echo "all parameters are: $@"

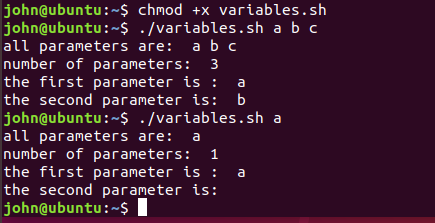
echo "number of parameters: $#"

echo "the first parameter is : $1"

echo "the second parameter is: $2"

Run the script with different numbers of arguments and examine the result.

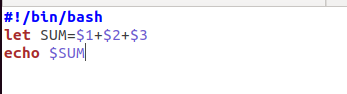


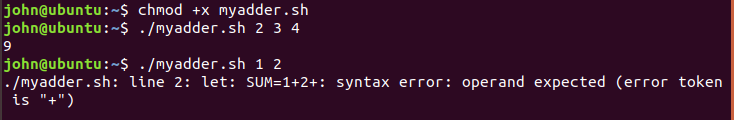


Now, write a new script that adds the parameters (up to three) together and outputs the result. For example:

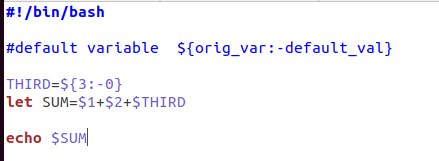
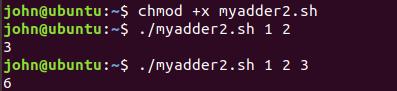
./myadder 1 2 outputs 3

./myadder 2 3 4 outputs 9

First try  




Hmm, doesn’t work when there are only two parameters. Looks like we need to fix it. One way is to use a default value of 0 for $3 so we don’t get errors if $3 is missing. Note: The default value line sets the default for \*all\* variables.

Normally variables are only available in the current process. These commands are a modified version of the review question from page 8. Run these commands one at a time, not in a script, and look at the results. (Note: if you run these commands from a script, the /bin/sh line will jump to a Bourne shell (sh) and your script will hang until you exit sh.)

DATA=' some stuff '

echo $DATA

/bin/sh

echo $DATA

/bin/bash

Now, change one of the commands so that both echo commands output "some stuff".

# Hand in

1. Hand in the script you wrote to add parameters.  
     
   See above.
2. How did you change the command DATA=' some stuff ' so that it worked for both your original BASH shell, and for the new Bourne shell (sh) you opened?

export DATA=’ some stuff’