Control flow in Bash

Bio724D: Fall 2023

2023-11-21

Starting script

```
#!/usr/bin/env bash

lines=$(ls $1 | wc -l)

if [[ $# -ne 1 ]]
then
      echo "Error: please provide a single valid directory path"
      exit 1
fi

echo "The directory $1 contains ${lines} objects."
```

If-then and If-then-else statements in Bash

```
if [ condition ]
then
  [ commands ]
fi
```

and

```
if [ condition ]
then
  [ commands ]
else
  [ other commands ]
fi
```

You can add additional conditions through the inclusion of `elif` blocks.

If-then and If-then-else examples

```
# if statement
if [[ $(ls *.txt | wc -l ) -gt 5 ]]
then
  echo "You've got more than 5 text files in your home directory!"
fi
```

```
# if-else statement
if [[ $(ls *.txt | wc -l ) -lt 5 ]]
then
  echo "You are text file poor. Better step up your text game!"
else
  echo "You have an abundance of text files in your home directory!"
fi
```

The operators `-gt` (“greater than”) and `-lt` (“less than”) in these examples are used for arithmetic comparisons. Other arithmetic logical operators include `-eq` (“equal to”), `-le` (“less than or equal to”) `-ge` (“greater than or equal to”) and `-ne` (“not equal to”).

Building on our earlier script

```
#!/usr/bin/env bash

lines=$(ls $1 | wc -l)

if [[ $# -ne 1 ]] || [[ ! -d $1 ]]  ## || = "or", && = "and"
then
      echo "Error: please provide a single valid directory path"
      exit 1
fi

echo "The directory $1 contains ${lines} objects."
```

For loops

```
for name in [ words ]
do
  [ commands ]
done
```

Meaning:

Expand `[ words ]` as needed and apply `[ commands ]` to each element of the resultant list; the variable `name` gets bound to each element.

For loop examples

```
# loop over the numbers 1 to 4
for item in {1..4}
do
  echo ${item} potato!
done
```

```
# loop over the .txt files in your current directory
# where the list of files is generated via command substitution
for filename in $(ls *.txt)
do
  echo ${filename} backward is $(echo ${filename} | rev)
done
```

```
# loop over a list of strings
for word in Who What Where When Why How
do
  echo "Newspaper writers are taught to discuss:" $word
done
```

Yet more improviements on our earlier script

```
#!/usr/bin/env bash

lines=$(ls $1 | wc -l)

if [[ $# -ne 1 ]] || [[ ! -d $1 ]]  ## || = "or", && = "and"
then
  echo "Error: please provide a single valid directory path"
  exit 1
fi

echo -e "The directory $1 contains ${lines-1} objects:\n"

for filename in $(realpath $1/*)
do
  if [[ -d ${filename} ]]
  then
    echo "${filename} is a directory."
  else
    echo "${filename} is a file. Its size in bytes is : " $(wc -c < ${filename})
  fi
done
```