Day 6

Script continuation:

Multiple condition under IF

Eg.

```
>> nano moreif.sh
# to use more than one condition within a if statement
echo Enter a number:
read num
if [ $num -ge 0 -a `expr $num % 2` -eq 0 ]
then
echo $num is even and positive
else
echo $num is odd or negative
fi
>> sh moreif.sh
```

Check the code yourself.

Nested IF

```
>> nano nestedif.sh
# to use nested if statement
echo Enter a number:
```

```
read num
if [ $num -ge 0 ]
then
if [ `expr $num % 2` -eq 0 ]
then
echo $num is even and positive
fi
else
echo $num is odd or negative
fi
>> sh nestedif.sh
```

This code should work like the previous one as well.

test something

string:

```
test $string = "string"
test $string1 = $string2
```

returns bool value

bool value is either True or False

files:

syntax

```
test -option file_name
```

test -option path_of_file

test -option folder_name

test -option path_of_folder

[folder is called directory too]

-option and function

-f = returns true if the given file exists

-d = returns true if the given folder exists

-r = returns true if the user has read access to the given file/folder

-w = returns true if the user has write access to the given file/folder

-x = returns true if the user has execute access to the given file/folder

-s = returns true if the file's size is more than 0-bit

eg. test -f file.txt   ----checks if the file exists

eg. test -frw file.txt   ----checks if the file exists, and is read-writable

Use of IF and TEST

eg.

```
>> nano testif.sh
# Program to delete a file if it exists
echo Script to DELETE a file
echo Enter file name
read file
if test -f $file
```

```
then
echo Do you really want to delete this file? [y/n]
read answer
if test $answer = "y"
then
rm $file
echo file deleted successfully
fi
else
echo Invalid file path/name.
fi
>> sh testif.sh
```

Looping

For loop in bash:

```
>> nano forbash.sh
# For loop in Bash
#!/bin/bash
for i in {1..10}
do
echo $i
done
```

```
>> bash forbash.sh
```

For loop in sh:

inefficient

```
>> nano for.sh
# For in sh
for i in 1 2 3 4 5 6 7 8 9 10
do
echo $i
done
>> sh for.sh
```

While looping in sh:

```
>> nano while.sh
# while
i=1
while [ $i -le 10 ]
do
echo $i
i=`expr $i + 1`
done
>> sh while.sh
```