School of Computer Science

COMP30640

Lab 3
Bash Scripts

Teaching Assistant:	Thomas Laurent
Coordinator:	Anthony Ventresque
Date:	Friday 28th September, 2018
Total Number of Pages:	4

To write your bash scripts you will need a text editor (vi? gedit? nano?). You can pick any, really, as long as they are **text** editors and not word processors (e.g., Microsoft Word), as the latter will add some undesirable elements to the documents.

- create a new file with your text editor (convention: give it a ".sh" extension; e.g., name the file "hello.sh").
- write a simple script/program that will display "hello everyone" in the terminal (standard output – use the echo command for that). Make sure that the first line contains `#!/bin/bash`.
- make the file executable by running the following command in the same directory as your hello.sh file:

```
$> chmod u+x hello.sh
```

- run your script by running the following command in the same directory as your hello.sh file:

```
$> ./hello.sh
```

Arguments.

In this exercise you will play with the elements that are given as input to a script/program (by the user, from a file, from the script's arguments).

- Write a script that takes an argument and is used as follows:

```
$> ./hello.sh Yoda
Hello Yoda!
```

Solution

```
#!/bin/bash

echo "hello $1!"
```

- Modify your script in such a way it can now accept as many arguments as the user wants:

```
$> ./hello.sh Yoda Leia Han Padme
Hello Yoda!
Hello Leia!
Hello Han!
Hello Padme!
```

Or:

```
$> ./hello.sh Yoda Leia "Han Solo" "Padme Amidala"
Hello Yoda!
Hello Leia!
Hello Han Solo!
Hello Padme Amidala!
```

You will need to use one of the structures you've seen during the lecture (for loop). This is a rough "skeleton" of your script (a.k.a., pseudo-code):

```
#! /bin/bash
#
# for each argument from the list of arguments given as input of the script
#   print Hello followed by the current argument followed by !
```

Solution

```
#!/bin/bash
for i in "$@"; do
    echo "Hello $i!"
done
```

- Modify your script: if there is no argument, your script should say hello to the Sith Lord, and in general the output should look like that:

```
$> ./hello.sh Yoda
Hello Yoda!
$> ./hello.sh
Hello Sith Lord!
```

The pseudo-code of your script is now:

```
#! /bin/bash
#
# if the script has no argument then
#   print Hello followed by "Sith Lord" followed by !
# else
#   for each argument from the list of arguments given as input of the script
#     print Hello followed by the current argument followed by !
```

Solution

```
#!/bin/bash
if  [ "$#" -eq 0 ]; then
    echo "Hello Sith Lord!"
else
    for i in "$@"; do
        echo "Hello $i!"
    done
fi
```

- Now I want your script to be... rude. Make sure it says hello only to every second character:

```
$> ./hello.sh Yoda Leia "Han Solo" "Padme Amidala"
Hello Leia!
Hello Padme Amidala!
```

```
#! /bin/bash
#
# if the script has no argument then
#   print Hello followed by "Sith Lord" followed by !
# else
#   while the number of arguments is greater than 0
#     print Hello followed by the current argument followed by !
#     shift the arguments (command shift)
#     shift the arguments (command shift)
#
```

Solution

```
#!/bin/bash
if  [ "$#" -eq 0 ]; then
    echo "Hello Sith Lord!"
else
    while [ "$#" -gt 0 ]; do
        echo "Hello $2!"
        shift
        shift
    done
fi
```

- Now, let's implement the same behaviour but using a different method. Use the pseudo code below:

```
#! /bin/bash
#
# if the script has no argument then
#   print Hello followed by "Sith Lord" followed by !
# else
#   for every argument
#     if the index of the argument in the list of arguments is even
#        print Hello followed by the current argument followed by !
#
```

Solution

```
#!/bin/bash
if  [ "$#" -eq 0 ]; then
```

```
    echo "Hello Sith Lord!"
else
    for (( i=1; i<=$#; i++ )); do
        if [ $((i % 2)) -eq 0 ]; then
            echo "Hello ${!i}!"
        fi
    done
fi
```