Exist Statuses and Return Codes

Exit Status

Return code, exit code or exit status is an INTEGER RANGING from 0 to 255

- 0 -> Successfully
- Non-Zero -> Error

Error Checking

The return codes can be used in a script for error checking

```
To consult exit statuses documentation the comamnds "man" and "info" can be
useful.
```

- The special variable **$?** contains the return code of the previously executed command.

```
ls /not/here
echo "Error code: $?"
```

```
output: ls: /not/here: no such file or directory
Error code: 2
```

```
HOST="google.com"
ping -c 1 $HOST

if["$?" -eq "0"]
then
    echo "$HOST reachable."
else
    echo "$HOST unreachable."
fi
```

Chaining Commands

Operators

- & (Ampersand): Sends a process/script/command to the background
- ; (Semi-colon): The command following this operator will execute even if the command preceding is not succesfully executed.

- && (LOGICAL AND): A command following a double ampersand will only run **if the previous command exists with a 0 exit status**
- || (LOGICAL OR): The command succeeding this operator is only executed** if the command preceding it has failed**
- ! (NOT): Negates an expression within a command.
- | (Pipe): The output of the first command acts as input to the second command
- < >> > (Redirection): Redirects the output of a command or a group of commands to a file or stream
- &&-|| (AND-OR): It is a combination of AND OR operator. Similar to the if-else statement
- \ (Concatenation): Used to concatenate large commands over several lines in the shell
- () (Precedence): Allows command to execute in precedence order
- {} (Combination): The execution of the command succeeding this operator depends on the execution of the first command

& Operator

Used to run a command **in the background** -> another commands can be executed

- Increases the effective utilization of system resources and speeds up the script execution.
- This is called child process creation or forking

```
ping -cl google.com & # Change the command before &
ping -c1 google.com & ping -c1 geeksforgeeks.org &
```

&& AND Opertaor

The command succeeding this operator will only execute if the command preceding it gets succesfsfully executed. It is helpful when we want to execute a command iif the first comand has executed successfully

Piping || Opertaor

```
ls -l | wc -l
```

Seds the ouput of the first command to the input of the second command.

NOT ! Operator

```
touch a.txt b.txt c.txt d.txt e.txt
rm -r !(a.txt)
```

Negate an expression in command.

Redirection Operator

Used to redirect the ouput of a command or a group of commands to a stream or file. This operator can be used to redirect either standard input or standard output or both. Almost all comands accept input with redirection operators

```
cat >> filename
sort < filename
```

If-Else condition

```
[ ! -d ABC ] && mkdir ABC || cd ABC
```

This command will first check if the directroy ABC exists or not. If it does not exist then a new directory is created, else ABC becomes the current directory.

The exit Command

Control the exit status of your shell scripts using the exit command.

If a exit command not specify a return code with the exit command, then the exit status of the previously executed command is used as exit status

If you not include a exit command in a shell script, the exit commmand of the previously executed command is used as exit status

exit command can be used anywhere in a shell script. When a exit command is reached, the script will stop running

```
#!/bin/bash

HOST="GOOGLE.COM"
ping -c 1 $HOST
if[ "$?" -ne "0" ]
then
    echo "$HOST unreacheable."
    exit 1
fi
exit 0
```