Functions

exit vs return

`exit` can be used both in scripts and in functions. In both cases, it causes *exiting* from the script.

`return` can be used to exit from a function (but not from a script). Both `exit` and `return` accept a number which acts as the exit status for the function or script.

```
# Define a function.
function greet () {
  printf '%s\n' 'Hello!'
}

# Invoke the function.
greet

# Take a peek at its exist status.
echo $?
# → 0
```

1. The `greet` function prints a simple text. `printf` returns 0, meaning "success", that is, "the printf command was executed successfully."
2. Invoke the function greet. Because it does not provide an explicit return value, it returns the exit status of the last command.
3. `$?` contains the exit status of the last command, in this case, the return/exit status of the `greet` invocation.

exit inside a function

```
function greet () {
  printf '%s\n' 'Hello'
  exit 1
  echo 'world'
}

greet

echo 'The Force'
```

Prints "Hello", exits with 1, and does not even print "world". Even though `exit` was used inside the function, it exits from the entire script itself. That is why we never reach the line `echo 'The Force'` either.