Mandatory Assignment 2

Bash scripting

University of Oslo - IN3110/IN4110

Fall 2019

2.1: Climbing the directory tree (2 points)

When navigating the directory tree, we often want to go back several directories. In the command line, this is done with e.g. `cd ../../../` which takes us three directories up the tree. When we want to go back several steps, writing `../` many times can get tedious.

Make a bash function `climb` that takes care of this for you. It should take an optional numeric arguments, which defaults to 1. The argument tells the function how many directories it should go up.

Examples

- `climb` and `climb 1` is equivalent to `cd ..`
- `climb 5` is equivalent to `cd ../../../../../`

Note about running: Because this is something you want to manipulate your current bash session, you cannot run this as a script (i.e. with `bash climb.sh`). Instead, you must source the file containing the function before you run it. i.e. `source climb.sh` and then you can use the function as above in the command line. To make the function work whenever you open a new terminal, you can put it in your `.bashrc`.

Name of file: `climb.sh`

2.2: A simple time tracker (5 points)

Create a program for tracking time spent on tasks. It should be rather simple. We are going to assume that we are only working on one task at a time, and the timer is stopped, we cannot start it again without creating a new task. The time tracking data should be stored in a file which name is specified by an environment variable called `LOGFILE`.

Supported functionality

The following ways to use the program should be implemented

- `track start [label]`: Starts a new task with a label. Should print an error message if a task is aready running.
- `track stop`: Stops the current task, if there is one running.
- `track status`: Tells us what task we are currently tracking, or if we don't have an active task.

If any other arguments are given, a helpful message should be displayed to tell the user how the program works.

File format

Here is a suggestion for a format for the logfile

```
START Fri Aug 24 15:31:59 CEST 2018
LABEL This is task 1
END Fri Aug 24 15:32:18 CEST 2018

START Fri Aug 24 15:31:59 CEST 2018
LABEL This is task 2
END Fri Aug 24 15:32:18 CEST 2018
```

2.3 Making the time tracker useful (3 points)

Tracking time isn't really useful unless we can display the data in a useful way. Extend your script from 2.2 to support a `log` command, that displays the time spent on each task on the format `HH:MM:SS`. Example output:

```
Task 1: 02:30:12
Task 2: 00:03:32
```