MP09

UF03

THREADING PROPOSED EXERCISES

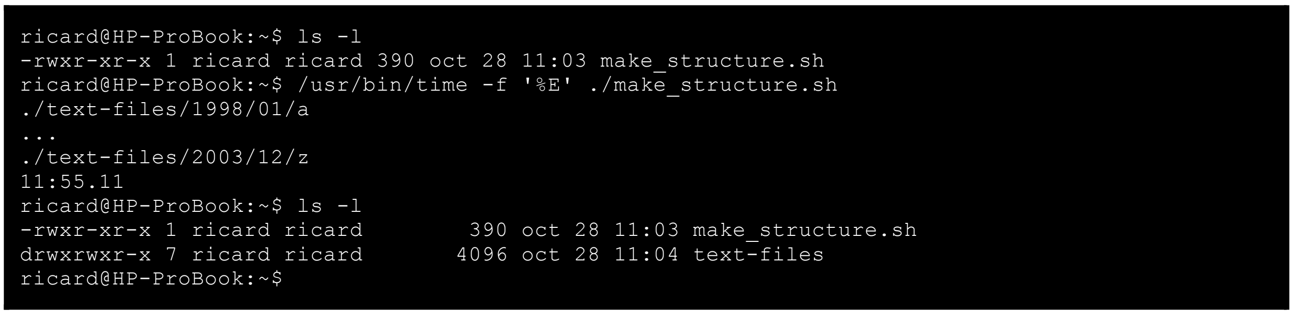
01. The following shell script generates a directory structure with files:

```
#!/bin/bash

rm -rf ./text-files/
mkdir -p ./text-files/{1998..2003}/{01..12}/{a..z}/

file_input="/usr/share/dict/american-english"
for d in $(find ./text-files/ -type d | grep '[a-z]$' | sort); do
  for n in {00..99}; do
    file_output="$d/text_file_${n}"
    shuf -n $((RANDOM % 10000 + 1)) $file_input | tr '\n' ' ' > $file_output
    echo "end." >> $file_output
  done
  echo $d
done
```

Execution example:



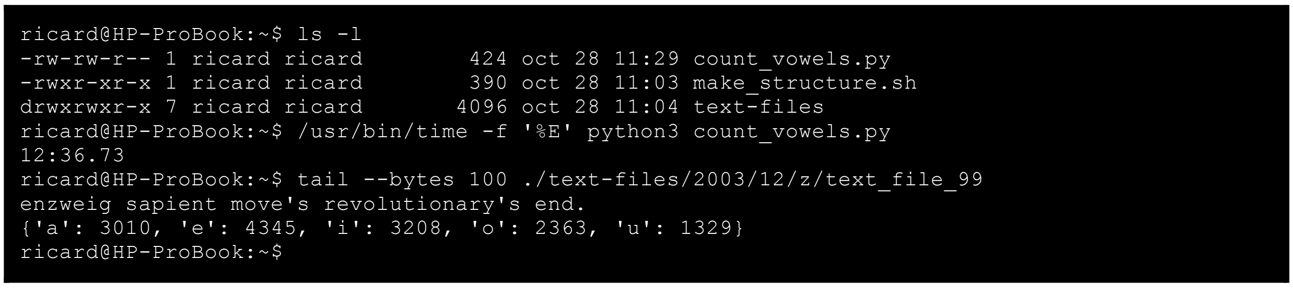
```
ricard@HP-ProBook:~$ ls -l
-rwxr-xr-x 1 ricard ricard 390 oct 28 11:03 make_structure.sh
ricard@HP-ProBook:~$ /usr/bin/time -f '%E' ./make_structure.sh
./text-files/1998/01/a
...
./text-files/2003/12/z
11:55.11
ricard@HP-ProBook:~$ ls -l
-rwxr-xr-x 1 ricard ricard        390 oct 28 11:03 make_structure.sh
drwxrwxr-x 7 ricard ricard       4096 oct 28 11:04 text-files
ricard@HP-ProBook:~$
```

The following program (count_vowels.py) goes through the entire structure in search of plain text files, every time it finds one it reads it to count the number of vowels and adds the result to the end of the file:

```
import os

for dirpath, dirnames, filenames in os.walk("./text-files/"):
    for filename in filenames:
        file = os.path.join(dirpath, filename)
        vowels = {'a': 0, 'e': 0, 'i': 0, 'o': 0, 'u': 0}
        with open(file) as f:
            data = f.read()
        for c in data:
            if c in vowels:
                vowels[c] += 1
        with open(file, 'a') as f:
            f.write(f"{str(vowels)}\n")
```

Execution example:



```
ricard@HP-ProBook:~$ ls -l
-rw-rw-r-- 1 ricard ricard        424 oct 28 11:29 count_vowels.py
-rwxr-xr-x 1 ricard ricard        390 oct 28 11:03 make_structure.sh
drwxrwxr-x 7 ricard ricard       4096 oct 28 11:04 text-files
ricard@HP-ProBook:~$ /usr/bin/time -f '%E' python3 count_vowels.py
12:36.73
ricard@HP-ProBook:~$ tail --bytes 100 ./text-files/2003/12/z/text_file_99
enzweig sapient move's revolutionary's end.
{'a': 3010, 'e': 4345, 'i': 3208, 'o': 2363, 'u': 1329}
ricard@HP-ProBook:~$
```

Create a Python program (count_vowels_with_threads.py) that performs the same work as the previous program using threading (one main thread and child threads, where child additional threads share the work). Indicate whether the execution time of the program that uses three threads is better than that which only uses one.

02. The following program (download_multiple_files.py) download some images from a web page:

```
# download_multiple_files.py

import requests

images_to_download = [
    {"url": "https://esahubble.org/media/archives/images/original/heic1502a.psb", "filename": "andromeda.psb"},
    {"url": "https://esahubble.org/media/archives/images/original/heic1909a.tif","filename": "universe.tif"},
    {"url": "https://esahubble.org/media/archives/images/original/heic1901a.tif", "filename": "triangulum.tif"},
    {"url": "https://esahubble.org/media/archives/images/original/heic1620a.tif", "filename": "goods_south.tif"}
]

for file_info in images_to_download:
    file_url = f"{file_info['url']}"
    file_name = file_info['filename']

    r = requests.get(file_url, headers={'user-agent': 'Mozilla/5.0'})
    with open(file_name, 'wb') as f:
        f.write(r.content)
```

Execution example:

```
ricard@HP-ProBook:~$ ls -l
-rw-rw-r-- 1 ricard ricard 1,3K oct 28 12:48 download_multiple_files.py
ricard@HP-ProBook:~$ /usr/bin/time -f '%E' python3 download_multiple_files.py
3:10.35
ricard@HP-ProBook:~$ ls -lh
-rw-rw-r-- 1 ricard ricard 4,4G oct 28 12:49 andromeda.psb
-rw-rw-r-- 1 ricard ricard 1,3K oct 28 12:48 download_multiple_files.py
-rw-rw-r-- 1 ricard ricard 928M oct 28 12:50 goods_south.tif
-rw-rw-r-- 1 ricard ricard 1,6G oct 28 12:50 triangulum.tif
-rw-rw-r-- 1 ricard ricard 921M oct 28 12:50 universe.tif
ricard@HP-ProBook:~$
```

Create a Python program (download_multiple_files_with_threads.py) that performs the same work as the previous program using threading (one main and one for each image to download). Indicate whether the execution time of the program that uses three threads is better than that which only uses one.

03. Create a Python program (square.py) that draws a square, to do this it will first ask the length of the side and then draw it (curses) using a thread (holding the drawing on the screen for five seconds).

Execution example:

```
ricard@HP-ProBook:~$ python3 square.py
side? <CR>
side must be a number
side? a<CR>
side must be a number
side? 1<CR>
side must be a number between 3 and 10
side? 4<CR>
```

```
****
*  *
*  *
****
```

```
side? quit<CR>
ricard@HP-ProBook:~$
```

<CR> (carriage return) indicates that a line jump has to be inserted (press the Enter key).

04. Create a program that solves a real-life problem with threading.