Assignment 2: Implementing more of linux `ls`

EC602 Design by Software

Fall 2021

Contents

1 Introduction

1.1 Group Size

For this assignment, the maximum group size is 2.

1.2 Due Date

This assignment is due 2021-9-29 at 23:59, i.e. one minute before midnight.

This assignment will be accepted through a grace period of $H = 60$ hours.

The grade will be scaled on a linear scale from 100% at the deadline to 50% at the end of the grade period.

If the *natural grade* on the assignment is g, the number of hours late is h, and the grace period is H, then the grade is

```
grade = (h > H) ? 0 : g * (1- h/(2*H));
```

in C++ or

```
grade = 0 if h>H else g * (1- h/(2*H))
```

in python.

If the same assignment is submitted ontime *and* late, the grade for that component will be the maximum of the ontime submission grade and the scaled late submission grade.

1.3 Submission Link

Assignment Two Submission Page

1.4 Points

This assignment is worth 5 points.

2 The Assignment

This assignment will implement various versions and parts of the unix command `ls`.

2.1 About `ls`

All information about how `ls` works can be found using two methods

- reading the man page with `man ls`
- experimenting with `ls` in the virtual machine, to basically "reverse-engineer" its properties.

We have done both of these in class: you will probably need to do more investigations using these methods to complete the assignment.

To about confusion between our programs and the real `ls`, all of the assignment components will have names similar to but different from `ls`.

Note: the `ls` provided by macOS does not function in precisely the same way as `ls` as provided in the VM (which is GNU coreutils 8.32)

Note: `ls` provides short style options like `-t` as will as long-style options like `--time=WORD`. For this assignment, we will assume only short style options.

No testing will be done with any long style options, and you are not expected to handle these cases.

2.1.1 Quotes

Some filenames cause `ls` to add quote symbols around the actual filenames ('filename' or "filename"). This creates substantial additional complication to the problems described below.

Therefore, you can assume that the option to turn off this mechanism `-N` or `--literal` is always specified and so you should not attempt to add quotes around the filenames you print.

2.2 Process file command line options: lsfiles

Write an executable python script `lsfiles` that works the same way as `ls` when files and/or directories are specified on the command line.

Example:

```
lsfiles firstfile anotherfile adirectory morefilenames
```

No command line options will be specified.

Your program should automatically switch between one-line and column output in the same way that `ls` does.

2.3 Long Format: lslong

Write an executable python script `lslong` that works the same way as `ls -l`

3 Program Restrictions

You may not access the actual program `ls` as part of any of your solutions, i.e. inside the python scripts themselves. This means that, for example, you may not use:

- `os.system`
- `subprocess.run`
- `subprocess.Popen`

or any other such facility in python to directly access the `ls` utility. You may use `os.listdir` and other features of the `os` and `sys` modules as needed.