CSCB09H3: Software Tools and Systems Programming Assignment 1: Shell Programming

Due Date: 11:59 pm Sunday, Jan 27, 2019
Worth: 10% of your final grade.

Overview:

In this assignment you will write **three** shell programs that operate on a collection of photographs. The first program, `mkpics` will generate an html file to display a table of photos. The second, `filepics` will organize the photos into directories by the date that they were created. The third program `mkpics2` will extend `mkpics` to handle the directories created by the second program.

To get full marks, your code must be well-documented, and use idiomatic shell programming. This means that you should avoid creating processes when you don't need to. It also means that you should use the special variables when appropriate. **Using `ls` and `cat` in your programs is not allowed.**

All programs are to be written using `#!/bin/sh` to be run on mathlab or the lab machines.

Setting up your working environment:

Follow the instructions carefully, so that we receive your work correctly. The instructions below will setup your svn work environment. The final submission will include the three shell scripts (`mkpics`, `filepics, mkpics2).`

Your first step should be to log into mathlab or any of the lab machines in BV 473 using your UtorID and password. Create a directory called `cscb09` (directories are created using `mkdir`). Change into that directory (using `cd`). Verify that you are actually in the right directory (running `pwd` should return `/courses/courses/cscb09w19/your_utorid/cscb09`). The following commands should accomplish the above:

```
mkdir ~/cscb09_space/cscb09
cd ~/cscb09_space/cscb09
pwd
```

Now check out your SVN repo using the following command:

```
svn co svn+ssh://your_utor_id@mathlab.utsc.utoronto.ca/svn/cscb09w19/your_utor_id
```

You will be asked for a password which is your usual utorid password. The repository we have created is empty, so all you will see after checking out you repo is an empty directory (inside your `/courses/courses/cscb09w19/your_utorid/cscb09` directory) with the name of your UtorID. (Use `ls` to check that the directory is there). This directory is a working copy of your svn repository. Start by changing into the directory and create a new directory called a1 by executing the following commands:

```
cd ~/cscb09_space/cscb09/your_utor_id
mkdir a1
```

Next you want to add your new directory `a1` to the repository (using `svn add`), check in this change to the svn server (using `svn ci`), and change into your new `a1` directory:

```
svn add a1
svn ci -m "I have created the a1 directory"
cd a1
```

For this assignment you will be asked to hand in three different files `mkpics, filepics` and `mkpics2`. These three files are the three shell scripts we are asking you to implement as part of this assignment and these files should be submitted in the `a1` directory using `svn add` and then `svn ci` the change.

Script 1: mkpics -- creating a web page of photographs (35%)

The program `mkpics` will write to standard output valid html that will display a table of photos. The first command line argument to `mkpics` is the number of columns in the table, and the remaining arguments are the files to be displayed.

Details:

- If there are not enough pictures to fill all of the rows, then the incomplete row may be the first row or the last row.
- Please include the `height=100` attribute in each `img` tag in your table so that the page will be displayed reasonably.
- There does not need to be any fancy formatting of the table or any additional text except for the title.
- Your html output does not need to look pretty but should be readable. In other words don't worry too much about indentation and white space when writing the output.
- Your program should only include jpeg pictures in the table. It should verify that a file is a picture by checking the file type using the `file` program.
- Errors should be handled reasonably.
    - Your program should not crash.
    - It should write all error messages to standard error and exit in appropriate situations.
    - It should handle 0 or more pictures.
    - If your program is run with an argument (other than the first one) that is not a jpeg picture, it should report this to standard error with an appropriate message and continue creating the table using the valid filenames. If none of the arguments are valid filenames your program should create and write an empty table to standard output.

A possible html file that might be produced by your program is shown:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Pictures</title>
  </head>

  <body>
    <h1>Pictures</h1>
```

```
<table>
  <tr>
        <td><img src="pictures/IMGP0959.jpg" height=100></td>
        <td><img src="pictures/IMGP2739.jpg" height=100></td>
        <td><img src="pictures/IMGP2850.jpg" height=100></td>
  </tr>
  <tr>
        <td><img src="pictures/IMGP3064.jpg" height=100></td>
        <td><img src="pictures/IMGP3069.jpg" height=100></td>
        <td><img src="pictures/IMGP3528.jpg" height=100></td>
  </tr>
 </table>
</body> </html>
```

Some Helpful Tips

A directory of images that you may use for testing can be found at:

```
/courses/courses/cscb09w19/nizamnau/a1/pics
```

To check that your output produced the desired results, you can open the html files that your script creates in a web browser to verify that the formatting is as expected.

For this assignment you will likely need to do some simple integer arithmetic on variables. As the shell naturally operates on strings, not integers, you will need the command `expr`. See the man pages for how to use it. As an example, the following will assign the result of `5+3` to the variable `y`.

```
y=`expr 3 + 5`
```

Script 2: filepics -- sorting photos into directories (35%)

Digital cameras typically add data to a jpeg file known as Exif (Exchangeable Image File) data. This data includes the time the picture was taken or generated.

In `/courses/courses/cscb09w19/bin` you will find a program called `exiftime` that prints to standard output the time information stored in a jpeg file. To use this program (without typing the full path everytime) you will need to add `/courses/courses/cscb09w19/bin` to your `PATH` variable.

The example below shows how to run exiftime (before adding its location to PATH) on an image file called `pic1.jpg` and the output it produces:

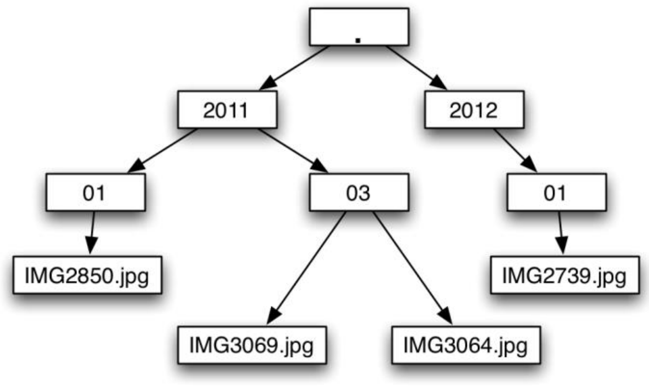
```
/courses/courses/cscb09w19/bin/exiftime -tg pic1.jpg
Image Generated: 2006:07:09 00:03:29
```

The `filepics` program will take an existing directory as an argument. For each picture in that directory, `filepics` will use `exiftime` to get the time that the picture was generated. It will move the pictures to directories by year, and within year by month. The year directories will be subdirectories of the directory from which the script was run.

For example, if the directory `vacation` contains the following files with the `exiftime -tg` output shown beside them:

```
IMGP2739.jpg: Image Generated: 2012:01:02 16:14:03
IMGP2850.jpg: Image Generated: 2011:01:01 18:17:44
IMGP3064.jpg: Image Generated: 2011:03:10 11:39:40
IMGP3069.jpg: Image Generated: 2011:03:10 12:15:24
```

Then the result of running `filepics vacation` would be the following:



Details:

- Your program will create the appropriate directories, but should create only the directories that are needed.
- Your program should do reasonable error handling.
- Any non-picture files in the original directory should be ignored.

Hint: You may find the `cut` program useful for this script.

Script 3: mkpics2 -- extending mkpics (30%)

The `mkpics2` program is a modified version of your `mkpics` program that takes the number of columns and a directory as an argument instead of the number of columns and list of files. The directory is the root of a tree of ``filed" pictures from running `filepics`. It contains subdirectories by year which themselves contain subdirectories by month which contain the pictures.

Your program will write to standard output one html document that has a table of photos for each year. Each table will be preceded by an `<h2>` header with the year. All of the details from part 1 apply to this program as well.

Running `mkpics2 2 .` on the tree above would produce the following html page.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Pictures</title>
  </head>

  <body>
    <h1>Pictures</h1>

<h2>2012</h2>
<table>
  <tr>
        <td><img src="2012/01/IMGP2739.jpg" height=100></td>
  </tr>
</table>

<h2>2011</h2>
<table>
  <tr>
        <td><img src="2011/01/IMGP2850.jpg" height=100></td>
        <td><img src="2011/03/IMGP3064.jpg" height=100></td>
  </tr>
  <tr>
        <td><img src="2011/03/IMGP3069.jpg" height=100></td>
  </tr>
</table>
</body>
</html>
```

Again, your program should do **reasonable** error checking. You'll notice that this instruction is quite different from what you may have encountered in your previous CS courses. In B09, we will often not be specifying exactly what is reasonable error checking or answering questions about every specific case. You must make your own judgement calls about what assumptions are acceptable and what error checking is required. As a general rule, a user should not be able to crash your program and you should provide the user with a helpful usage message when the command-line arguments are incorrect.

Learning objectives

- Shell control structures and variables
- Shell control structures and variables
- Running external programs from a shell program
- Shell quoting
- Working with files and directories using the shell

What to hand in

We will look for the following three files in your svn repository (based on your UtorID):

- `mkpics`
- `filepics`
- `mkpics2`

Don't forget to run `svn add` and `svn ci` for the three shell scripts after adding them to your a1 directory.

Please remember to make your shell scripts executable by ensuring that the first line is `#!/bin/sh`.

You are strongly encouraged to take advantage of the version control system and commit your work frequently so that you can keep track of your progress. Please note that it is perfectly fine (and even recommended) that you keep any additional files related to this assignment under version control. The markers will simply ignore these files.