Aims

This exercise aims to give you more practice with using the Unix shell for processing collections of files.

Assessment

Submission: `give cs2041 lab04 jpg2png.sh email_image.sh date_image.sh fix_id3_tags.sh [create_music.sh]`
Deadline: either during the lab, or Monday 22 August 11:59pm (midnight)
Assessment: Make sure that you are familiar with the lab assessment criteria (lab/assessment.html).

Exercise: Converting Images

Write a shell script `jpg2png.sh` which converts all images in JPEG (http://en.wikipedia.org/wiki/JPEG) format in the current directory to PNG (http://en.wikipedia.org/wiki/Portable_Network_Graphics) format.
You can assume that JPEG files and only JPEG files have the suffix `jpg` .

If the conversion is succesful the JPEG file should be removed.

Your script should stop with an appropriate error message and exit status if the PNG file already exists.

```
$ wget http://www.cse.unsw.edu.au/~cs2041/lab/sh/images/images.zip
$ unzip images.zip
Archive:  images.zip
  inflating: Johannes Vermeer - The Girl With The Pearl Earring.jpg
  inflating: nautilus.jpg
  inflating: panic.jpg
  inflating: penguins.jpg
  inflating: shell.jpg
  inflating: stingray.jpg
  inflating: treefrog.jpg
$ ./jpg2png.sh
$ ls
Johannes Vermeer - The Girl With The Pearl Earring.png  panic.png
email_image.sh                                          penguins.png
images.zip                                              shell.png
index.php                                               stingray.png
jpg2png.sh                                              treefrog.png
nautilus.png
$ cp -p /home/cs2041/public_html/lab/sh/images/penguins.jpg .
$ ./jpg2png.sh
penguins.png already exists
```

Hints

You may find sed and back quotes useful.

The tool `convert` will convert between many image formats, for example:

```
$ convert penguins.jpg penguins.png
```

Sample solution for jpg2png.sh

```
#!/bin/sh

for jpg_file in *.jpg
do
    png_file=`echo $jpg_file|sed 's/jpg$/png/'`
    if test -e "$png_file"
    then
        echo "$png_file" already exists
        exit 1
    fi
    convert "$jpg_file" "$png_file" && rm "$jpg_file"
done
```

Exercise: Emailing Images

Write a shell script `email_image.sh` which given a list of image files as arguments displays them one-by-one. After the user has viewed each image the script should prompt the user for an e-mail address. If the user does enter an email address, the script should prompt the user for a message to accompany the image and then send the image to e-mail address. to that address.

```
$ ./email_image.sh penguins.png treefrog.png
Address to e-mail this image to? andrewt@cse.unsw.edu.au
Message to accompany image? Penguins are cool.
penguins.png sent to andrewt@cse.unsw.edu.au
Address to e-mail this image to? andrewt@cse.unsw.edu.au
Message to accompany image? This is a White-lipped Tree Frog
treefrog.png sent to andrewt@cse.unsw.edu.au
```

Hints

The program `display` can be used to view image files

The program `mutt` can be used to send mail from the command line including attachments, for example:

```
$ echo 'Penguins are cool.'|mutt -s 'penguins!' -a penguins.png -e set copy=no -- nobody@nowhere.com
```

Sample solution for email_image.sh

```
#!/bin/sh

for png_file in "$@"
do
    display "$png_file"
    echo -n "Address to e-mail this image to? "
    read address
    if test -n "$address"
    then
        echo -n "Message to accompany image? "
        read message
        echo "$message"| mutt -s 'image' -a "$png_file" -e set copy=no -- "$address"
        echo "$png_file sent to $address"
    else
        echo "No email sent"
    fi
done
```

Python solution

```
#!/usr/bin/python

import smtplib, subprocess, sys
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart

from_address = "andrewt@unsw.edu.au"
for png_file in sys.argv[1:]:
    subprocess.check_output(['echo', 'display', png_file])
    sys.stdout.write("Address to e-mail this image to? ")
    sys.stdout.flush()
    to_address = sys.stdin.readline().strip()
    if to_address:
        sys.stdout.write("Message to accompany image? ")
        sys.stdout.flush()
        message = sys.stdin.readline().strip()
        msg = MIMEMultipart(message)
        msg['Subject'] = png_file
        msg['From'] = from_address
        msg['To'] = to_address
        with open(png_file) as f:
            attachment = MIMEText(f.read())
            attachment.add_header('Content-Disposition', 'attachment', filename=png_file)
            msg.attach(attachment)
        s = smtplib.SMTP('smtp.cse.unsw.edu.au')
        s.sendmail(from_address, [to_address], msg.as_string())
        s.quit()
    else:
        print("No email sent")
```

Exercise: Annotating Images

Write a shell script `date_image.sh` which, given a list of image files as arguments, changes each file so it has a label added to the image indicating the time it was taken. You can assume the last-modification time of the image file is the time it was taken.

So for example if we these commands were run:

```
$ cp -p /home/cs2041/public_html/lab/sh/images/penguins.jpg .
$ ls -l penguins.jpg
-rw-r--r-- 1 andrewt andrewt 58092 Mar 16 16:08 penguins.jpg
$ ./date_image.sh penguins.jpg
$ display penguins.jpg
```

Then `penguins.jpg` should have been be modified to look like this:

Hints

The program convert can be used to label an image like this:

```
$ convert -gravity south -pointsize 36 -draw "text 0,10 'Andrew rocks'" penguins.jpg temporary_file.jpg
```

Hint: `sed` and/or `cut` may be useful to extract the date&time from ls's output.

Hint: `convert` produce confusing messages if you don't get its option syntax exactly right

Sample solution for date_image.sh

```
#!/bin/sh

for image_file in "$@"
do
    last_modify_time=`ls -l "$image_file"|cut -d\  -f5-|sed 's/^ *//'|cut -d\  -f2-4`
    temporary_file="$image_file.tmp.$$"
    if test -e "$temporary_file"
    then
        echo "$temporary_file" already exists
        exit 1
    fi
    convert -gravity south -pointsize 36 -annotate 0 "text 0,10 '$last_modify_time'" $image_file $tempo
    touch -r $image_file $temporary_file &&  # preserve modification time (challenge question)
    mv $temporary_file $image_file
done
```

Challenge Question: preserving file modification times

Modify `date_image.sh` so it doesn't affect the image file's last-modification time. For example:

```
$ cp -p /home/cs2041/public_html/lab/sh/images/penguins.jpg .
$ ls -l penguins.jpg
-rw-r--r-- 1 andrewt andrewt 58092 Mar 16 16:08 penguins.jpg
$ ./date_image.sh penguins.jpg
$ ls -l penguins.jpg
-rw-r--r-- 1 andrewt andrewt 58092 Mar 16 16:08 penguins.jpg
```

Exercise: Organizing Music

Andrew's needs help fixing the ID3 (https://en.wikipedia.org/wiki/ID3) tags in the MP3 (https://en.wikipedia.org/wiki/MP3) files in his music collection.
You will write a shell script `fix_id3_tags.sh` which set appropriate ID3 tags for Andrew's music collection.

Your script will determine the appropriate ID3 tags from the directory names and file names of the music collection.

You assume the names follows a standard format. You can determine this format by downloading (lab/sh/music/music.zip) Andrew's music collection.

```
$ wget http://www.cse.unsw.edu.au/~cs2041/lab/sh/music/music.zip
...
$ unzip music.zip
Archive:  music.zip
   creating: music/
   creating: music/Triple J Hottest 100, 2006/
  inflating: music/Triple J Hottest 100, 2006/2 - Black Fingernails, Red
  inflating: music/Triple J Hottest 100, 2006/6 - Crazy - Gnarls Barkley
  inflating: music/Triple J Hottest 100, 2006/5 - I Don't Feel Like Danc
...
```

The command `id3` can be used list the value of ID3 tags in an MP3 file, for example:

```
$ id3 -l 'music/Triple J Hottest 100, 2013/1 - Riptide - Vance Joy.mp3'
music/Triple J Hottest 100, 2013/1 - Riptide - Vance Joy.mp3:
Title  : Andrew Rocks                   Artist: Andrew
Album  : Best of Andrew                 Year: 2038, Genre: Unknown (25
Comment:                                Track: 42
```

As you can see the ID3 tags of this music file have been accidentally over-written. The ID3 tags should be:

```
$ id3 -l 'music/Triple J Hottest 100, 2013/1 - Riptide - Vance Joy.mp3'
music/Triple J Hottest 100, 2013/1 - Riptide - Vance Joy.mp3:
Title  : Riptide                        Artist: Vance Joy
Album  : Triple J Hottest 100, 2013     Year: 2013, Genre: Unknown (25
Comment:                                Track: 1
```

Fortunately all the information needed to fix the ID3 tags is available in the name of the file and the name of the directory it is in.

You will write a shell script `fix_id3_tags.sh` which takes the name of 1 or more directories in Andrew's music collection as arguments and fixes the ID# tags of the all MP3 files in that directory. For example:

```
$ fix_id3_tags.sh 'music/Triple J Hottest 100, 2015'
$ id3 -l 'music/Triple J Hottest 100, 2015/4 - The Less I Know the Better - Tame Impala.mp3'
music/Triple J Hottest 100, 2015/4 - The Less I Know the Better - Tame I
Title  : The Less I Know the Better       Artist: Tame Impala
Album  : Triple J Hottest 100, 2015     Year: 2015, Genre: Unknown (25
Comment:                                Track: 4
$ fix_id3_tags.sh music/*
$ id3 -l 'music/Triple J Hottest 100, 1995/10 - Greg! The Stop Sign!! - TISM.mp3'
music/Triple J Hottest 100, 1995/10 - Greg! The Stop Sign!! - TISM.mp3:
Title  : Greg! The Stop Sign!!          Artist: TISM
Album  : Triple J Hottest 100, 1995     Year: 1995, Genre: Unknown (25
Comment:                                Track: 10
$ id3 -l 'music/Triple J Hottest 100, 1999/1 - These Days - Powderfinger.mp3'
music/Triple J Hottest 100, 1999/1 - These Days - Powderfinger.mp3:
Title  : These Days                     Artist: Powderfinger
Album  : Triple J Hottest 100, 1999     Year: 1999, Genre: Unknown (25
Comment:                                Track: 1
$ id3 -l 'music/Triple J Hottest 100, 2012/2 - Little Talks - Of Monsters and Men.mp3'
music/Triple J Hottest 100, 2012/2 - Little Talks - Of Monsters and Men
Title  : Little Talks                   Artist: Of Monsters and Men
Album  : Triple J Hottest 100, 2012     Year: 2012, Genre: Unknown (25
Comment:                                Track: 2
```

Your script should not change the *Genre* or *Comment* fields.

Your script should determine *Title*, *Artist Track Album* & *Year* from the directory & filename.

Hints

```
$ man id3
...
```

`cut` almost works for extracting *Title* and *Album* from the filename.

Handling the few MP3 files correctly where using `cut` doesn't work will be considered a **challenge exercise**.

It can be difficult debugging your script on Andrew's music collection. In cases like these it usually worth creating a smaller data set for initial debugging. Such a tiny data set is available in tiny_music.zip (lab/sh/music/tiny_music.zip) if you want to use it for debugging. This dataset is used in the first autotests.

```
$ wget http://www.cse.unsw.edu.au/~cs2041/lab/sh/music/tiny_music.zip
$ unzip tiny_music.zip
Archive:  tiny_music.zip
   creating: tiny_music/
   creating: tiny_music/Album1, 2015/
  inflating: tiny_music/Album1, 2015/2 - Little Talks - Of Monsters and
  inflating: tiny_music/Album1, 2015/1 - Riptide - Vance Joy.mp3
   creating: tiny_music/Album2, 2016/
  inflating: tiny_music/Album2, 2016/2 - Royals - Lorde.mp3
  inflating: tiny_music/Album2, 2016/1 - Hoops - The Rubens.mp3
$ id3 -l tiny_music/*/*.mp3
tiny_music/Album1, 2015/1 - Riptide - Vance Joy.mp3:
Title  : Andrew Rocks                  Artist: Andrew
Album  : Best of Andrew                Year: 2038, Genre: Unknown (25
Comment:                               Track: 42
tiny_music/Album1, 2015/2 - Little Talks - Of Monsters and Men.mp3:
Title  : Andrew Rocks                  Artist: Andrew
Album  : Best of Andrew                Year: 2038, Genre: Unknown (25
Comment:                               Track: 42
tiny_music/Album2, 2016/1 - Hoops - The Rubens.mp3:
Title  : Andrew Rocks                  Artist: Andrew
Album  : Best of Andrew                Year: 2038, Genre: Unknown (25
Comment:                               Track: 42
tiny_music/Album2, 2016/2 - Royals - Lorde.mp3:
Title  : Andrew Rocks                  Artist: Andrew
Album  : Best of Andrew                Year: 2038, Genre: Unknown (25
Comment:                               Track: 42
$ ./fix_id3_tags.sh tiny_music/*
$ id3 -l tiny_music/*/*.mp3
tiny_music/Album1, 2015/1 - Riptide - Vance Joy.mp3:
Title  : Riptide                       Artist: Vance Joy
Album  : Album1, 2015                  Year: 2015, Genre: Unknown (25
Comment:                               Track: 1
tiny_music/Album1, 2015/2 - Little Talks - Of Monsters and Men.mp3:
Title  : Little Talks                  Artist: Of Monsters and Men
Album  : Album1, 2015                  Year: 2015, Genre: Unknown (25
Comment:                               Track: 2
tiny_music/Album2, 2016/1 - Hoops - The Rubens.mp3:
Title  : Hoops                         Artist: The Rubens
Album  : Album2, 2016                  Year: 2016, Genre: Unknown (25
Comment:                               Track: 1
tiny_music/Album2, 2016/2 - Royals - Lorde.mp3:

Title  : Royals                        Artist: Lorde
Album  : Album2, 2016                  Year: 2016, Genre: Unknown (25
Comment:                               Track: 2
```

Sample solution for fix_id3_tags.sh

```
#!/bin/sh

for album_pathname in "$@"
do
    album=`basename "$album_pathname"`
    year=`echo "$album"|sed 's/.* //'`

    for mp3_pathname in "$album_pathname"/*.mp3
    do
        mp3_filename=`basename "$mp3_pathname" .mp3`
        # assume ' - ' doesn't occur in artist or album
        track=`echo "$mp3_filename"|sed 's/ - .*//'`
        title=`echo "$mp3_filename"|sed 's/^[0-9]* - //;s/ - .*//'`
        artist=`echo "$mp3_filename"|sed 's/.* - //'`
        id3 -t "$title" -T "$track" -a "$artist" -A "$album" -y "$year" "$mp3_pathname" >/dev/null
    done
done
```

You can run some tests on your script like this:

```
$ ~cs2041/bin/autotest lab04 fix_id3_tags.sh
```

You can also specify that only a single test be run:

```
$ ~cs2041/bin/autotest lab04 tiny_album1
```

Also do your own testing!

Challenge Exercise: Creating Music

The test data for the previous question is not really Andrew's music collection. All the mp3 files contain identical contents. The directories and filenames were created from the source of this web page (https://en.wikipedia.org/wiki/Triple_J_Hottest_100). Write a shell script `create_music.sh` which uses the above webpage to create exactly the same directories and files as in the test data set supplied above. Your script should take 2 arguments: the name of an MP3 file to use as the contents of the MP3 files you create and the directory in which to create the test data. For example:

```
$ wget http://www.cse.unsw.edu.au/~cs2041/lab/sh/music/music.zip
...
$ unzip music.zip
$ wget http://www.cse.unsw.edu.au/~cs2041/lab/sh/music/sample.mp3
$ ./create_music.sh sample.mp3 created_music
$ ls created_music
Triple J Hottest 100, 1993  Triple J Hottest 100, 1998  Triple J Hottest
Triple J Hottest 100, 1994  Triple J Hottest 100, 1999  Triple J Hottest
Triple J Hottest 100, 1995  Triple J Hottest 100, 2000  Triple J Hottest
Triple J Hottest 100, 1996  Triple J Hottest 100, 2001  Triple J Hottest
Triple J Hottest 100, 1997  Triple J Hottest 100, 2002  Triple J Hottest
$ ls 'created_music/Triple J Hottest 100, 2012'
1 - Thrift Shop - Macklemore and Ryan Lewis featuring Wanz.mp3  5 - I Wi
10 - My Gun - The Rubens.mp3                                    6 - Get
2 - Little Talks - Of Monsters and Men.mp3                      7 - Elep
3 - Breezeblocks - Alt-J.mp3                                    8 - Lost
4 - Holdin' On - Flume.mp3                                      9 - Feel
$ diff -r music created_music/
$
```

Hints

```
$ wget -q -O- 'https://en.wikipedia.org/wiki/Triple_J_Hottest_100?action=raw'
...
```

Sample solution for create_music.sh

```
#!/bin/sh

mp3_file="$1"
base_dir="$2"

wget -q -O- 'https://en.wikipedia.org/wiki/Triple_J_Hottest_100?action=raw'|
while read line
do
    # look for line which is start of Hottest 100 list for a year

    case "$line" in
    *'[[Triple J Hottest 100, '[0-9][0-9][0-9][0-9]'|'[0-9][0-9][0-9][0-9]*) ;;
    *) continue;;
    esac

    # create a directory for a Hottest 100
    album=`echo "$line"|sed 's/.*\[\[//;s/|.*//'`
    year=`echo "$album"|sed 's/.*\ //'`
    dir="$base_dir/Triple J Hottest 100, $year"
    mkdir -p -m 755 "$dir"

    # read top 10 songs for year
    track=1
    while read line && test $track -le 10
    do
        case "$line" in
        '#'*) ;;
        *) continue;;
        esac

        # remove links to wikipedia pages
        line=`echo "$line"|sed 's/[^[]*|//g'`

        # change slashes to hyphens - because can't have / in a filename
        line=`echo "$line"|sed 's/\//-/g'`

        # remove some formating characters
        line=`echo "$line"|tr -d '[]"#'`

        #break line in two at en dash byte codes
        artist=`echo "$line"|sed 's/\xe2\x80\x93.*//'`
        title=`echo "$line"|sed 's/.*\xe2\x80\x93//'`

        #trim leading spaces
        artist=`echo "$artist"|sed 's/^ *//'`
        title=`echo "$title"|sed 's/^ *//'`

        #trim trailing spaces
        artist=`echo "$artist"|sed 's/ *$//'`
        title=`echo "$title"|sed 's/ *$//'`

        file="$dir/$track - $title - $artist.mp3"
        cp -p "$mp3_file" "$file"
        track=$((track + 1))
    done
done
```

Finalising

You must show your solutions to your tutor and be able to explain how they work. Once your tutor has discussed your answers with you, you should submit them using

```
$ give cs2041 lab04 jpg2png.sh email_image.sh date_image.sh fix_id3_tags.sh  [create_music.sh]
```

Only submit `create_music.sh` if you attempt the challenge exercise. Whether you discuss your solutions with your tutor this week or next week, you must submit them before the above deadline.