CS 270

Homework 04

Due 10/19/09

Colby Blair

I. Goal/Motivation

The goal of this homework is to become much more familiar with writing Perl scripts. Perl was built on the bash style, but resembles C as well. Some of the useful features are the Regular Expressions functionality built into Perl, as well as many built in and additional modules.

II. Code

See attached files archive.cpp.pdf and main.cpp

III. Data/Ouput

```
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -l
test.2
test1
test.1
test.4
test
test.3
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -r=test.4
mv /home/myriad360/trashcan//test.4 test.4
myriad360@anubis:~/CS_270/hw04$ ls
test  test.1  test.4  trashcan.pl  trashcan.pl~
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -r=test.2
mv /home/myriad360/trashcan//test.2 test.2
myriad360@anubis:~/CS_270/hw04$ ls
test  test.1  test.2  test.4  trashcan.pl  trashcan.pl~
myriad360@anubis:~/CS_270/hw04$ ls ../../trashcan/
test  test1  test.1  test.3
```

```
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -r=test
The file test already exists in the current directory:
total 28
-rw-r--r-- 1 myriad360 myriad360   32 2009-10-19 17:21 test
-rw-r--r-- 1 myriad360 myriad360   39 2009-10-19 18:18 test.1
-rw-r--r-- 1 myriad360 myriad360   46 2009-10-19 18:33 test.2
-rw-r--r-- 1 myriad360 myriad360   39 2009-10-19 18:33 test.4
-rw-r--r-- 1 myriad360 myriad360 5247 2009-10-19 18:52 trashcan.pl
-rw-r--r-- 1 myriad360 myriad360 1753 2009-10-18 21:44 trashcan.pl~
Would you like to overwrite (o), or abort (a)?: a
myriad360@anubis:~/CS_270/hw04$ ls ../../trashcan/
test  test1  test.1  test.3
myriad360@anubis:~/CS_270/hw04$ ls > test3
myriad360@anubis:~/CS_270/hw04$ ls > test.3
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -r=test.3
The file test.3 already exists in the current directory:
total 36
-rw-r--r-- 1 myriad360 myriad360   32 2009-10-19 17:21 test
-rw-r--r-- 1 myriad360 myriad360   39 2009-10-19 18:18 test.1
-rw-r--r-- 1 myriad360 myriad360   46 2009-10-19 18:33 test.2
-rw-r--r-- 1 myriad360 myriad360   59 2009-10-19 18:55 test3
-rw-r--r-- 1 myriad360 myriad360   66 2009-10-19 18:55 test.3
-rw-r--r-- 1 myriad360 myriad360   39 2009-10-19 18:33 test.4
-rw-r--r-- 1 myriad360 myriad360 5247 2009-10-19 18:52 trashcan.pl
-rw-r--r-- 1 myriad360 myriad360 1753 2009-10-18 21:44 trashcan.pl~
Would you like to overwrite (o), or abort (a)?: o
mv /home/myriad360/trashcan//test.3 test.3
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -l
test1
test.1
test
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -e
```

```
rm /home/myriad360/trashcan/test1 -r
rm /home/myriad360/trashcan/test.1 -r
rm /home/myriad360/trashcan/test -r
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -l
myriad360@anubis:~/CS_270/hw04$ ls > ../../trashcan/test1
myriad360@anubis:~/CS_270/hw04$ ls > ../../trashcan/test2
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -l
test1
test2
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -f
rm /home/myriad360/trashcan/ -r
myriad360@anubis:~/CS_270/hw04$ ls ../../trashcan/
ls: cannot access ../../trashcan/: No such file or directory
myriad360@anubis:~/CS_270/hw04$ ls ../../trashcan/
myriad360@anubis:~/CS_270/hw04$ ls > ../../trashcan/test1
myriad360@anubis:~/CS_270/hw04$ ls > ../../trashcan/test2
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -i
Delete test1? (y or n): y
rm /home/myriad360/trashcan/test1 -r
Delete test2? (y or n): n
myriad360@anubis:~/CS_270/hw04$ perl trashcan.pl -l
test2
```

IV. Revision of Data/Conclusion

The Perl assignment was a little challenging, in keeping the functionalities separate and well modulated. I used a built in Perl module called Get Options. I have used this module for numerous other Perl projects, and is a really great tool to use. Any option that is not attributed to a parent, i.e. parent -r or -l, is left in ARGV. This is extremely handy, as I didn't have to spend any time parsing and arranging file lists, and didn't have to spend much time messing with the options.

I also used a handy built in module called MD5 Sum. This makes file comparisons pretty easy. Overall, this assignment was a lot easier having previous experience with Perl. Knowing the built in functions and regular expressions easier.

```perl
use Getopt::Long;
use Env qw(PATH HOME TERM);
use Digest::MD5;

#-e - empty the trash. All files in the trashcan directory should be (really!) deleted.
#-f - flush. Like -e, except that the trashcan directory itself should also be deleted.
#-i - interactive. The program should ask the user before tc.ing any file (similar to
rm -i).
#-l - list. The files in trashcan should be listed, similar to issuing the command ls -
l trashcan. Directories should be listed, but their contents should not be.
#-r - retrieve. Copy the specified file from the trashcan directory to the current
directory. If a file with the same name as the retrieved file already exists in the
current directory, then your script should ask the user if the retrieved file should
replace the existing one. If the answer is .no, then nothing should happen.

my $opt_empty = "";
my $opt_flush = "";
my $opt_int = "";
my $opt_list = "";
my $opt_ret = "";
my $result = GetOptions("e"  =>  \$opt_empty,
      "f" => \$opt_flush,
      "i" => \$opt_int,
      "l" => \$opt_list,
      "r=s" => \$opt_ret
    );
my $trash_dir = $HOME . "/trashcan/";

die "ERROR: Could not parse command line options\n" unless $result;

sub process_filename
{
    my $filename = shift;

    if($filename =~ /.+[0-9]$/)
    {
  #create new postfix, .n + 1
  my $iend = length($filename) - 1;
  my $i = rindex($filename, '.');
  my $strx = substr($filename, $i + 1, $iend);
  my $pstfx = int($strx) + 1;
  #strip out the old post fix
  $filename = substr($filename, 0, $i) . ".$pstfx";
  #if this new filename with the new postfix exists in the directory, well, lets
keeping incrementing the postfix until it doesn't
  while(-e $filename)
  {
      $pstfx++;
      $filename = substr($filename, 0, $i) . ".$pstfx";
  }
  return($filename);
    }
    else
    {
  return($filename . ".1");
    }

}

sub md5sum
{
    my $filename = shift;
    my $md5sum = "";
    open(FILE, $filename) or die "ERROR: Could not open file $filename\n";
    my $md5handler = Digest::MD5->new;
    $md5handler->addfile(*FILE);
    $md5sum = $md5handler->hexdigest;
    close(FILE);

    return($md5sum);
}


sub make_trash_dir
{
    if(-e $trash_dir)
    {return;}
    else
    {
  my $cmd = "mkdir $trash_dir\n";
  print $cmd;
  system($cmd);
    }
}

sub move_files
{
    foreach(@ARGV)
    {
  my $srcsum = 0;
  my $dstsum = 1;
  if(-e $trash_dir . $_)
  {
      $srcsum = md5sum($_);
      $dstsum = md5sum($trash_dir . $_);
      #if the currently being deleted file, srcfile, has an md5
      #sum same as the dstfile (file already in the garbage can)
      if($srcsum == $dstsum)
      {
    print "File $_ was already deleted. Using currently being deleted file as trash 
can copy.\n";
    my $cmd = "rm $trash_dir$_ -r\n";
    print $cmd;
    system($cmd);
    $cmd = "mv $_ $trash_dir$_ \n";
    print $cmd;
    system($cmd);
      }
      else
      {
    my $oldfilename = $_;
    my $newfilename = process_filename($trash_dir . $_);
    print "File with name $oldfilename already exists, but is different. Archiving old 
$oldfilename file as $newfilename .\n";
    #archive file with same file name already in trach can
    my $cmd = "mv $trash_dir$oldfilename $newfilename \n";
    print $cmd;
    system($cmd);
    #move the user specified file into the trash can
    $cmd = "mv $oldfilename $trash_dir$oldfilename \n";
    print $cmd;
    system($cmd);
      }
  }
  else
  {
      my $cmd = "mv " . $_ . " " . $trash_dir . $_ . " \n";
```

```
        print $cmd;
        system($cmd);
    }
      }#end for each element in ARGV
}

sub empty_trash
{
    opendir(DIR, $trash_dir);
    my @FILES = readdir(DIR);
    foreach(@FILES)
    {
  if($_ ne '.' && $_ ne '..')
  {
        my $cmd = "rm " . $trash_dir . $_ . " -r\n";
        print $cmd;
        system($cmd);
  }
    }
}

sub flush_trash
{
    my $cmd = "rm " . $trash_dir . " -r \n";
    print $cmd;
    system($cmd);
}

sub empty_trash_interactive
{
    opendir(DIR, $trash_dir);
    my @FILES = readdir(DIR);
    foreach(@FILES)
    {
  if($_ ne '.' && $_ ne '..')
  {
        print "Delete " . $_ . "? (y or n): ";
        my $in_str = <STDIN>;
        if($in_str =~ /y/)
    {
          my $cmd = "rm " . $trash_dir . $_ . " -r\n";
          print $cmd;
          system($cmd);
    }
  }
    }
}

sub list_trash_contents
{
    opendir(DIR, $trash_dir);
    my @FILES = readdir(DIR);
    foreach(@FILES)
    {
  if($_ ne '.' && $_ ne '..')
  {
        my $cmd = $_ . "\n";
        print $cmd;
  }
    }
}

sub retrieve
{
    my $filename = $opt_ret;
    my $in_str = "";

    if (-e $filename)
    {
  print "The file $filename already exists in the current directory:\n";
  system("ls -l");

  #ask user what to do about conflict
  while($in_str !~ /o/ && $in_str !~ /a/)
  {
      print "Would you like to overwrite (o), or abort (a)?: ";
      $in_str = <STDIN>;
      if($in_str !~ /o/ && $in_str !~ /a/)
      {print "ERROR: Enter 'o' or 'a'\n";}
  }
    }
    if($in_str !~ /a/)
    {
  my $cmd = "mv $trash_dir/$filename $filename \n";
  print $cmd;
  system($cmd);
    }

}

#############main
make_trash_dir;

if($opt_empty)
{
    empty_trash;
}
elsif($opt_flush)
{
    flush_trash;
}
elsif($opt_int)
{
    empty_trash_interactive;
}

if($opt_list)
{
    list_trash_contents;
}
if($opt_ret)
{
    retrieve;
}

#all options not assigned to the above options in
#GetOptions will be moved into the trashcan
move_files;
```