```
# The work below belongs to Josh Evnin, so don’t copy it without asking.
# © Josh Evnin, 2004

#!/usr/bin/perl -w

@cmdline = @ARGV;              #take in words from the command line
$lenCmdline = @cmdline;        #find number of tokens are on the command
line

#===========================================================#
# Corner case test, gives error if there are more or less
# than 2 entries on the command line

if ($lenCmdline != 2){
      print "------------------------------------\n";
      print "I'm sorry, i'm just a dumb computer :-(.  It would
really help me if you entered your input like this:\n \n perl proj1.pl
[name of input file] [name of output file]\n
then I'll be able to do better next time.\n";
      print "------------------------------------\n";
      exit;
}
#=========================================================#


# if ($cmdline[0] eq 'josh'){print "i'm exiting\n"; exit;}

open(FILE,$cmdline[0]);
open(WRITE,">$cmdline[1]");

$count = 0;             #later used to count which task i'm on

@lines = <FILE>;  #takes in the lines from the input file
chomp(@lines);          #takes off the end of line chars

$length = @lines;
for($i = 0;$i<$length;$i++){
     #print "looking at $lines[$i]";
     @cues = split/,/,$lines[$i];   #breaks each line into indivual
chars
      #print "cues[$i]: @cues";

#===========================================================#
# This is a test case for letter or strange character input

      $lenCues = @cues;
      $tester = 0;
      for($k=0;$k<$lenCues;$k++){
            #print "k=$k, cues[k]=$cues[$k]\n";
            if ($cues[$k] =~ /[^0-9+*\/%!-]/){
            #     print "&& $cues[$k]\n";
                  &letterInput;
                  $tester = 1;      # this avoids the main math loop
            }
      }

# These are  test cases for wrong entry format
```

```
      if($tester != 1){
              if ($cues[0] !~ /[+*\/%!-]/){
                  &wrongFormat;
                  $tester = 1;      # this avoids the main math loop
            }
      }
      if($tester != 1){
            for($z=1;$z<$lenCues;$z++){
                  if($cues[$z] !~ /[0-9]/){
                        &wrongFormat;
                        $tester = 1;
                  }
            }
      }

#============================================================#
# Enter main math loop

    #========================================================#
    # Corner case test for missing operands
     if($tester != 1){
      if($lenCues < 3 && $cues[0] ne "!"){
            &missingOperand;
      }
      elsif($cues[0] eq "!"){
       #print "$cues[1] $cues[0]";
        &factorial;
        }
      else{
          #print "$cues[1] $cues[0] $cues[2]";
          if($cues[0] eq "+"){&add;}
          elsif($cues[0] eq "-"){&subtract;}
          elsif($cues[0] eq "*"){&multiply;}
          elsif($cues[0] eq "/"){&divide;}
          elsif($cues[0] eq "**"){&power;}
          elsif($cues[0] eq "%"){&mod;}
        }
     }
}

#======================================================#
#These messages are for the errors
sub wrongFormat{
      &taskCounter;           #increments $count
      print WRITE "Task $count: Uh oh! Please use op,num,num format\n";
}


sub missingOperand{
      &taskCounter;           #increments $count
      print WRITE "Task $count: D'oh! Input missing operand. Try:
operator,operand,operand\n";
}

sub letterInput{
      &taskCounter;
      print WRITE "Task $count: Oops!  Please input numbers and
operators only!\n";
}

#=====================================================#
# These messages are for the math

sub add{
      &taskCounter;
      $answer = $cues[1] + $cues[2];
      print WRITE "Task $count: $cues[1] + $cues[2] = $answer\n";
      #print "###", $answer;
      #print "\n";
}

sub subtract{
      &taskCounter;
      $answer = $cues[1] - $cues[2];
      print WRITE "Task $count: $cues[1] - $cues[2] = $answer\n";
      #print "###", $answer, "\n";
}

sub multiply{
      &taskCounter;
      $answer = $cues[1] * $cues[2];
      print WRITE "Task $count: $cues[1] * $cues[2] = $answer\n";
      #print "###", $answer, "\n";
}

sub divide{
      &taskCounter;
      $answer = $cues[1] / $cues[2];
      print WRITE "Task $count: $cues[1] / $cues[2] = $answer\n";
}

sub power{
      &taskCounter;
      $answer = $cues[1] ** $cues[2];
      print WRITE "Task $count: $cues[1] ** $cues[2] = $answer\n";
}

sub mod{
      &taskCounter;
      $answer = $cues[1] % $cues[2];
      print WRITE "Task $count: $cues[1] % $cues[2] = $answer\n";
}

sub factorial{
      &taskCounter;
      $answer = 1;
      for($j=1;$j<=$cues[1];$j++){
            $answer *= $j;
      }
      print WRITE "Task $count: $cues[1]! = $answer\n";
      #print "###", $answer, "\n";
}
```

```perl
# This increments $count manually
sub taskCounter{
      $count += 1;
}

print "\n";
close(FILE);      # Close off the input file
close(WRITE);     # Close off the output file
#print "I did it!\n"
```