```
#!/usr/bin/perl

#26 Jun 08:  a new GO term parser that uses GeneIDs to collect  gene
symbol, gene name, organism, GO grouping (mf, bp, cc), modifiers to terms
(NOT, etc.), full 7 digit GO ID numbers, and evidence codes, condensed
into a single line for multiple instances of the same term/modifier pair
with different evidence codes.
#21 Oct 08: modded to run on accession numbers for paul.


#outstanding issues: currently none

use LWP::Simple;

#import a list of GeneIDs for search
print "GeneID list file: ";
$genelist = <STDIN>;
chomp $genelist;
open (GENELIST, "$genelist");
@genelist = (<GENELIST>);
close GENELIST;

#open each genedB entry by building a URL and retrieving it. identifies
false genedB terms and flags them, skips to the next in the list.
foreach $genelist (@genelist) {
    chomp $genelist;
    $URL =
"http://www.ncbi.nlm.nih.gov/sites/entrez?db=gene&cmd=search&term=$geneli
st";
    $file = get($URL);
    @file = split("\n",$file);
    foreach $file (@file) {
      if ($file =~ /No items found - Gene Results/) {print"***No GenedB
data found for GeneID "; last}
    }

#predeclare and clear variables
      $geneid = "";
    $organism = "";
    $putative = "";
    $HUGO = "";
    $mf = 0; #set go secondary grouping conditions to 'false'
    $bp = 0;
    $cc = 0;
    @go_group = (); #secondary grouping entry
    @go_id = (); #numerical value collected from genedB
    @go_term = (); #text description
    @go_evid = (); #three letter evidence codes
    @go_mod = (); #not, contributes_to, etc.
    @go_complete_entry = (); #an array containing lines with group, mod,
id, term, and evid all together, to be fed to condense_by_evidence
    @go = (); #the array returned by condensed_by_evidence

    foreach $file (@file) {
```

```
    if ($file =~ /GeneID: (\d*)/) {$geneid = $1} #pick up geneID for
formatting
      if ($file =~ /<th class="th1">Function<\/th>/) {$mf = 1; $bp = 0;
$cc = 0} #check to see what group we are in in the term list
      if ($file =~ /<th class="th1">Process<\/th>/) {$bp = 1; $mf = 0;
$cc = 0}
      if ($file =~ /<th class="th1">Component<\/th>/) {$cc = 1; $bp = 0;
$mf = 0}
      if ($file =~ /<span class="num">1: <\/span><span
class="gn">(.*?)<\/span> (.*?) \[ <span class="tax">(.*?)<\/span> \]
<\/h1>$/) {
          $HUGO = $1; $putative = $2; $organism = $3;
      }
        if ($file =~ /<td>\s(\w*?)\s<a/) {
          push @go_mod, $1; #presence of a modifier before the GO term
        }
      if ($file =~ /<td><a href="\/entrez/) {
          push @go_mod, " "; #confirmed absence of a modifier before the
GO term
      }
      if ($file =~
/amigo\/go.cgi\?view\=details\&amp;depth\=1\&amp;query=(\d*?)">(.*?)<\/a>
/) {
        push @go_id, $1; push @go_term, $2;
      while ($bp) {push @go_group, "BP"; last} #one group term per GO
term, according to the state measured above
      while ($mf) {push @go_group, "MF"; last}
      while ($cc) {push @go_group, "CC"; last}
        }
      if ($file =~ /<td>(.{2,3})<\/td>/) {
          push @go_evid, $1;
      }
    }
    print $genelist, "\t", $geneid, "\t", $HUGO, "\t", $putative, "\t",
$organism, "\n";
    $i = 0;
    foreach $go_id (@go_id) {seven_digit_GO($go_id)}
    foreach $go_id (@go_id) {
            $temp = "";
            $temp =
"$go_group[$i]\t$go_mod[$i]\t$go_id[$i]\t$go_term[$i]\t$go_evid[$i]\n";
            push @go_complete_entry, $temp;
            $i++;
    }
    condense_by_evidence(@go_complete_entry);
    foreach $go (@go) {print $go}
    print "\n";
}


###SUBROUTINES
#subroutine seven_digit_GO: adds leading zeroes to go id numbers to
produce canonical GO ID
sub seven_digit_GO(){
```

```
    my $id = shift (@_);
    chomp $go_id;
    if (length($go_id) == 6) {$go_id =~ s/\d{6}/0$go_id/}
    if (length($go_id) == 5) {$go_id =~ s/\d{5}/00$go_id/}
    if (length($go_id) == 4) {$go_id =~ s/\d{4}/000$go_id/}
    if (length($go_id) == 3) {$go_id =~ s/\d{3}/0000$go_id/}
    if (length($go_id) == 2) {$go_id =~ s/\d{2}/00000$go_id/}
    if (length($go_id) == 1) {$go_id =~ s/\d{1}/000000$go_id/}
    return $go_id;
}

#subroutine condense_by_evidence: compares two adjacent GO entries for
identical terms, then merges the evidence codes into a single line and
deletes the additional lines.
sub condense_by_evidence(){
    my @term = ();
    my @evid = ();
    my (@input_array) = (@_);
    foreach $input_array (@input_array) {
      chomp $input_array;
      if ($input_array =~ /^(\w{2}\t.*?\t\d{7}\t.*?\t)(\w*)$/) {
          push @term, $1; push @evid, $2; #splits the input into term and
evidence for comparison
      }
    }

#windows through and compares each pair of adjacent terms for identity.
if matched, the evidence codes are merged, and the second of the pair is
deleted
    $i = 0; $j = 1;
    while ($j < scalar(@term)) {
      foreach $term (@term) {
          if ($term[$j] eq $term[$i]) {
            $evid[$i] =~ s/$evid[$i]/$evid[$i]\/$evid[$j]/;
            splice (@evid,$j,1);
            splice (@term,$j,1);
            next;
          }
          $i++; $j++;
      }
    }

#returns the nicely formatted data in a new array
    $k = 0;
    foreach $term (@term) {
      push @go, $term[$k];
      push @go, $evid[$k];
      push @go, "\n";
      $k++;
   }
return @go;
}
```