```
    1: Script started on Tue Mar  6 21:04:05 2012
    2: bash-3.2$ cat -n me\007rgesort.pl
    3:      1  % $Id: mergesort.pl,v 1.3 2011-05-19 19:53:59-07 - - $ */
    4:      2
    5:      3  %
    6:      4  % Some sorting examples.
    7:      5  %
    8:      6
    9:      7  %
   10:      8  % Insertion sort's top level function accepts the Unsorted list
   11:      9  % and returns the Sorted list.  Insert inserts one element into
   12:     10  % list such that the output list is sorted.
   13:     11  %
   14:     12
   15:     13  insertion_sort( Unsorted, Sorted ) :-
   16:     14     insertion_sort_gather( Unsorted, [], Sorted ).
   17:     15
   18:     16  insertion_sort_gather( [], Gathered, Gathered ).
   19:     17  insertion_sort_gather( [Head|Tail], Gathered, Sorted ) :-
   20:     18     insert( Head, Gathered, NewGathered ),
   21:     19     insertion_sort_gather( Tail, NewGathered, Sorted ).
   22:     20
   23:     21  insert( Item, [], [Item] ).
   24:     22  insert( Item, [Head|Tail], [Item,Head|Tail] ) :-
   25:     23     Item =< Head.
   26:     24  insert( Item, [Head|Tail], [Head|NewTail] ) :-
   27:     25     Item > Head,
   28:     26     insert( Item, Tail, NewTail ).
   29:     27
   30:     28  %
   31:     29  % Merge sort divides the unsorted list into subparts and then
   32:     30  % merges the sublists back again in pairs.
   33:     31  %
   34:     32
   35:     33  mergesort( [], [] ).
   36:     34  mergesort( [Only], [Only] ).
   37:     35  mergesort( Unsorted, Sorted ) :-
   38:     36     split( Unsorted, UnsortedHalf1, UnsortedHalf2 ),
   39:     37     mergesort( UnsortedHalf1, SortedHalf1 ),
   40:     38     mergesort( UnsortedHalf2, SortedHalf2 ),
   41:     39     merge( SortedHalf1, SortedHalf2, Sorted ).
   42:     40
   43:     41  split( [], [], [] ).
   44:     42  split( [Only], [Only], [] ).
   45:     43  split( [First,Second|Tail], [First|Tail1], [Second|Tail2] ) :-
   46:     44     split( Tail, Tail1, Tail2 ).
   47:     45
   48:     46  merge( [], List, List ).
   49:     47  merge( List, [], List ).
   50:     48  merge( [Head1|Tail1], [Head2|Tail2], [Head1|NewTail] ) :-
   51:     49     Head1 =< Head2,
   52:     50     merge( Tail1, [Head2|Tail2], NewTail ).
   53:     51  merge( [Head1|Tail1], [Head2|Tail2], [Head2|NewTail] ) :-
   54:     52     Head1 > Head2,
   55:     53     merge( [Head1|Tail1], Tail2, NewTail ).
   56:     54
   57: bash-3.2$ gprolog
   58: GNU Prolog 1.3.1
   59: By Daniel Diaz
   60: Copyright (C) 1999-2009 Daniel Diaz
   61: | ?- [mergesort].
   62: compiling /afs/cats.ucsc.edu/courses/cmps112-wm/Languages/prolog/Examples/merges
ort.pl for byte code...
   63: /afs/cats.ucsc.edu/courses/cmps112-wm/Languages/prolog/Examples/mergesort.pl com
```

```
piled, 54 lines read - 4467 bytes written, 11 ms
   64:
   65: (1 ms) yes
   66: | ?- mergesort([8, 4, 3, 6, -9, 66, 10, 3], X).
   67:
   68: X = [-9,3,3,4,6,8,10,66] ?
   69:
   70: (1 ms) yes
   71: | ?-
   72:
   73:
   74: bash-3.2$ exit
   75:
   76: Script done on Tue Mar  6 21:05:39 2012
```