

```

optype stream = (int); # type of data streams

module Merge[i = 1 to n]
  op in1 stream, in2 stream; # input streams
  op initialize(cap stream); # link to output stream
body
  process Filter {
    int v1, v2; # values from input streams
    cap stream out; # capability for output stream
    in initialize(c) -> out = c ni
    # get first values from input streams
    in in1(v) -> v1 = v; ni
    in in2(v) -> v2 = v; ni
    while (v1 != EOS and v2 != EOS)
      if (v1 <= v2)
        { call out(v1); in in1(v) -> v1 = v; ni }
      else # v2 < v1
        { call out(v2); in in2(v) -> v2 = v; ni }
    # consume the rest of the non-empty input stream
    if (v1 == EOS)
      while (v2 != EOS)
        { call out(v2); in in2(v) -> v2 = v; ni }
    else # v2 == EOS
      while (v1 != EOS)
        { call out(v1); in in1(v) -> v1 = v; ni }
    call out(EOS);
  }
end Merge

```

**Figure 8.9** Merge sort filters using rendezvous.