Example - Merge Two Sorted Singly Linked Lists

* For this example, assume that lists do not have headers - a list is represented by a pointer to its first element.

Idea:

• Repeatedly compare the first elements of list1 and list2 and place the smaller at the end of list3; when one of the lists runs out, the other can be appended to the end of list3

• Care must be taken when lists are initially empty or become empty.

• The variable last maintains a pointer to the last vertex of list3.

(Even though the next field of the vertex pointed to by last may not be nil during the execution of the program, it always terminates by appending the remainder of list1 or list2 to list3, thus properly ending list3 with the next field of the last vertex nil.)
(list merge continued)

```plaintext
function listMERGE(list1,list2)
  list3 := last := nil
  while (list1≠nil) and (list2≠nil) do begin
    if (list1=nil) then begin
      if (list3=nil) then list3 := list2 else last.next := list2
      list2 := nil
    end
    else if (list2=nil) then begin
      <symmetric to the previous case>
    end
    else if (DATA(list1)<DATA(list2)) then begin
      if list3=nil then list3 := list1 else last.next := list1
      last := list1
      list1 := list1.next
    end
    else begin
      <symmetric to the previous case>
    end
  end
  return list3
end
```