

```

1: unit Intercal;
2:
3: interface
4:
5: uses DefDados;
6:
7: procedure MergeSort(var Lista : Tipo_da_Lista; Primeiro, Ultimo : TamLista);
8:
9: implementation
10:
11: (*****)
12:
13: procedure Merge(var Lista : Tipo_da_Lista; Primeiro, Meio, Ultimo : TamLista);
14: var
15:     Temp          : Tipo_da_Lista;
16:     I,
17:     ProxDireita,
18:     ProxEsquerda : longint;
19:
20:     { Copia um elemento da lista origem para a lista destino e
21:       incrementa as posicoes de origem e destino passadas }
22: procedure CopiarElemento(var PosOrigem, PosDestino : longint;
23:                          var Origem, Destino      : Tipo_da_Lista);
24: begin
25:     Destino.A[PosDestino] := Origem.A[PosOrigem];
26:     inc(PosOrigem);
27:     inc(PosDestino);
28: end; { CopiarElemento }
29:
30: begin { Merge }
31:     ProxEsquerda := Primeiro;
32:     ProxDireita  := Meio + 1;
33:     I            := Primeiro;
34:     with Lista do
35:         begin
36:             { Copia até que uma das sublistas ser totalmente copiada }
37:             while (ProxEsquerda <= Meio) and (ProxDireita <= Ultimo) do
38:                 if A[ProxEsquerda].Chave < A[ProxDireita].Chave then
39:                     CopiarElemento(ProxEsquerda, I, Lista, Temp)
40:                 else
41:                     CopiarElemento(ProxDireita, I, Lista, Temp);
42:
43:             { Copia os que restaram na lista da esquerda }
44:             while ProxEsquerda <= Meio do
45:                 CopiarElemento(ProxEsquerda, I, Lista, Temp);
46:
47:             { Copia os que restaram na lista da direita }
48:             while ProxDireita <= Ultimo do
49:                 CopiarElemento(ProxDireita, I, Lista, Temp);
50:
51:             { Copia os elementos de volta para a lista original }
52:             for I := Primeiro to Ultimo do
53:                 Lista.A[I] := Temp.A[I];
54:         end;
55:     end; { Merge }
56:
57: (*****)
58:
59: procedure MergeSort(var Lista : Tipo_da_Lista; Primeiro, Ultimo : TamLista);
60: var Meio : TamLista;
61: begin

```

```
62:     if Primeiro < Ultimo then
63:         begin
64:             Meio := (Ultimo + Primeiro) div 2;
65:             MergeSort(Lista, Primeiro, Meio);
66:             MergeSort(Lista, Meio + 1, Ultimo);
67:             Merge(Lista, Primeiro, Meio, Ultimo);
68:         end;
69:     end;
70:
71: end.
```