```
 1: unit Selecao;
 2: interface
 3: uses DefDados;
 4:
 5: procedure SelecaoDireta(var Lista : Tipo_da_Lista);
 6: procedure HeapSort(var Lista : Tipo_da_Lista);
 7:
 8: implementation
 9:
10: (*************************************************************************)
11:
12: procedure SelecaoDireta(var Lista : Tipo_da_Lista);
13: var
14:    I, J, Menor : TamLista;
15: begin
16:    with Lista do
17:       for I := 1 to N - 1 do
18:          begin
19:             Menor := I;
20:             for J := I + 1 to N do
21:                if A[J].Chave < A[Menor].Chave then
22:                   Menor := J;
23:             if Menor <> I then
24:                Troque(A[I], A[Menor]);
25:          end;
26: end;
27:
28:
29: (*************************************************************************)
30:
31: procedure HeapSort(var Lista : Tipo_da_Lista);
32:
33:   function Pai(IndFilho : TamLista) : TamLista;
34:   begin
35:      Pai := IndFilho div 2;
36:   end;
37:
38:   function FilhoEsq(IndPai : TamLista) : TamLista;
39:   begin
40:      FilhoEsq := 2*IndPai;
41:   end;
42:
43:   function FilhoDir(IndPai : TamLista) : TamLista;
44:   begin
45:      FilhoDir := 2*IndPai + 1;
46:   end;
47:
48:   procedure InserirNoHeap(var Lista : Tipo_da_Lista; I : TamLista);
49:   var IndPai, IndFilho : TamLista;
50:   begin
51:      with Lista do
52:      begin
53:         IndFilho := I;
54:         IndPai   := Pai(I);
55:
56:         while (IndFilho > 1) and (A[IndPai].Chave < A[IndFilho].Chave) do
57:         begin
58:            Troque(A[IndPai], A[IndFilho]);
59:            IndFilho := IndPai;
60:            IndPai   := Pai(IndPai);
61:         end;
```

```
62:       end;
63:    end; { InserirNoHeap }
64:
65:    procedure ReposicionarRaizNoHeap(var Lista : Tipo_da_Lista; TamHeap :
    TamLista);
66:    var I : TamLista;
67:    begin
68:        with Lista do
69:        begin
70:           I := 1;
71:           while (FilhoDir(I) <= TamHeap) and
72:                 ( (A[I].Chave < A[FilhoEsq(I)].Chave) or
73:                   (A[I].Chave < A[FilhoDir(I)].Chave) ) do
74:           begin
75:              if (A[FilhoEsq(I)].Chave > A[FilhoDir(I)].Chave) then
76:                 begin
77:                    Troque(A[I], A[FilhoEsq(I)]);
78:                    I := FilhoEsq(I);
79:                 end
80:              else
81:                 begin
82:                    Troque(A[I], A[FilhoDir(I)]);
83:                    I := FilhoDir(I);
84:                 end
85:           end;
86:
87:           if (FilhoEsq(I) = TamHeap) then
88:              if A[I].Chave < A[FilhoEsq(I)].Chave then
89:                 Troque(A[I], A[FilhoEsq(I)])
90:        end;
91:    end; { ReposicionarRaizNoHeap }
92:
93: var I : TamLista;
94: begin { HeapSort }
95:
96:     for I := 2 to Lista.N do
97:        InserirNoHeap(Lista, I);
98:
99:     for I := Lista.N downto 2 do
100:    begin
101:       Troque(Lista.A[I], Lista.A[1]);
102:       ReposicionarRaizNoHeap(Lista, I - 1);
103:    end;
104: end; { HeapSort }
105:
106: end.
```