```
 1: unit LstSeq;
 2:
 3: interface
 4:
 5: const MAX = 100;
 6:
 7: type
 8:    { Tipo de chave do item da lista }
 9:    Tipo_Chave = longint;
10:
11:    { Tipo do item da lista }
12:    Tipo_Item = record
13:                   Chave : Tipo_Chave;
14:                   Dado  : String[30]
15:                end;
16:
17:    { Tipo da lista sequencial }
18:    Lista_Sequencial = record
19:                          Itens   : array[1..MAX] of Tipo_Item;
20:                          Tamanho : 0..MAX
21:                       end;
22:
23: procedure Inicializar (var Lista : Lista_Sequencial);
24: function Inserir (Item : Tipo_Item; var Lista : Lista_Sequencial) : boolean;
25: function Remover (Chave : Tipo_Chave; var Lista : Lista_Sequencial) : boolean;
26: function Alterar (Item : Tipo_Item; var Lista : Lista_Sequencial) : boolean;
27: procedure Obter (Chave : Tipo_Chave; var Lista : Lista_Sequencial;
28:                  var Item : Tipo_Item; var Sucesso : boolean);
29: function Tamanho (var Lista : Lista_Sequencial) : longint;
30: function Cheia (var Lista : Lista_Sequencial) : boolean;
31: function Vazia (var Lista : Lista_Sequencial) : boolean;
32:
33: implementation
34:
35: procedure Inicializar (var Lista : Lista_Sequencial);
36: {
37:   Objetivo: Inicializa a lista passada, fazendo com que o tamanho seja
38:             igual a zero
39: }
40: begin
41:    Lista.Tamanho := 0
42: end;
43:
44:
45: function Inserir (Item : Tipo_Item; var Lista : Lista_Sequencial) : boolean;
46: {
47:   Objetivo: Insere o item passado como parametro na lista passada.
48:             Se o tamanho da lista ja for igual ao tamanho maximo,
49:             a funcao Inserir retorna false.
50: }
51: begin
52:    if Lista.Tamanho = MAX then
53:       Inserir := false
54:    else
55:       begin
56:          inc(Lista.Tamanho);
57:          lista.Itens[Lista.Tamanho] := Item;
58:          Inserir := true
59:       end
60: end;
61:
```

```
62:
63: function Remover (Chave : Tipo_Chave; var Lista : Lista_Sequencial) : boolean;
64: {
65:   Objetivo: Remove o item cuja chave coincide com o parametro Chave
66:             passado. Caso nao haja um item com essa chave, retorna
67:             false. Se o item foi removido, retorna true.
68: }
69: var
70:    I, J : longint;
71: begin
72:    Remover := false;
73:
74:    for I := 1 to Lista.Tamanho do
75:       if Lista.Itens[I].Chave = Chave then
76:          begin
77:             Remover := true;
78:
79:             for J := I + 1 to Lista.Tamanho do
80:                Lista.Itens[J - 1] := Lista.Itens[J];
81:
82:             dec(Lista.Tamanho);
83:             break
84:          end;
85: end;
86:
87:
88: function Alterar (Item : Tipo_Item; var Lista : Lista_Sequencial) : boolean;
89: {
90:   Objetivo: Altera os dados de um item existente na lista passada
91:             de forma que fique igual ao do item passado como parametro.
92:             Se o item for encontrado e alterado, retorna true. Caso
93:             contrario, retorna false.
94:
95: }
96: var
97:    I : longint;
98: begin
99:    Alterar := false;
100:
101:    for I := 1 to Lista.Tamanho do
102:       if Lista.Itens[I].Chave = Item.Chave then
103:          begin
104:             Lista.Itens[I] := Item;
105:             Alterar := true;
106:             break
107:          end;
108: end;
109:
110:
111: procedure Obter (Chave : Tipo_Chave; var Lista : Lista_Sequencial;
112:                  var Item : Tipo_Item; var Sucesso : boolean);
113: {
114:   Objetivo: Procura na lista usando a chave passada. Caso encontre
115:             Sucesso contem o valor true e Item contem o Item obtido.
116:             Caso contrario, Sucesso retorna true e Item nao e alterado
117: }
118: var
119:    I : longint;
120: begin
121:    Sucesso := false;
122:
```

```
123:    for I := 1 to Lista.Tamanho do
124:       if Lista.Itens[I].Chave = Chave then
125:          begin
126:             Sucesso := true;
127:             Item := Lista.Itens[I];
128:             break
129:          end
130: end;
131:
132:
133: function Tamanho (var Lista : Lista_Sequencial) : longint;
134: {
135:   Objetivo: Retorna o tamanho da lista passada
136: }
137: begin
138:    Tamanho := Lista.Tamanho
139: end;
140:
141:
142: function Cheia (var Lista : Lista_Sequencial) : boolean;
143: {
144:   Objetivo: Retorna true se a lista esta cheia (Tamanho = MAX)
145: }
146: begin
147:    Cheia := Tamanho(Lista) = MAX
148: end;
149:
150:
151: function Vazia (var Lista : Lista_Sequencial) : boolean;
152: {
153:   Objetivo: Retorna true se a lista esta vazia (Tamanho = 0)
154: }
155: begin
156:    Vazia := Tamanho(Lista) = 0
157: end;
158:
159: end.
```