

```

1: unit MatEsp;
2:
3: interface
4:
5: type
6:     Tipo_do_Dado = longint;
7:
8:     PNo = ^No;
9:
10:    No = record
11:        Linha,
12:        Coluna : word;
13:        Valor  : Tipo_do_Dado;
14:        ProxLinha,
15:        ProxColuna : PNo;
16:    end;
17:
18:    Matriz_Esparsa = PNo;
19:
20: procedure Inicializar (var Matriz : Matriz_Esparsa; NumLinhas, NumColunas :
    word);
21: function ObterAcima (var Matriz : Matriz_Esparsa; Linha, Coluna : word) : PNo;
22: function ObterEsquerda (var Matriz : Matriz_Esparsa; Linha, Coluna : word) :
    PNo;
23: function ObterNo(var Matriz : Matriz_Esparsa; Linha, Coluna : word) : PNo;
24: procedure InserirDepois (PL, PC : PNo; Valor : Tipo_do_Dado);
25: procedure ApagarDepois (PL, PC : PNo);
26: procedure Apagar (var Matriz : Matriz_Esparsa);
27:
28: function Obter (var Matriz : Matriz_Esparsa; Linha, Coluna : word;
29:     var Valor : Tipo_Do_Dado) : boolean;
30: procedure Mudar (var Matriz : Matriz_Esparsa; Linha, Coluna : word;
31:     Valor : Tipo_Do_Dado);
32: procedure Limpar (var Matriz : Matriz_Esparsa; Linha, Coluna : word);
33:
34:
35: implementation
36:
37: procedure Inicializar (var Matriz : Matriz_Esparsa; NumLinhas, NumColunas :
    word);
38: {
39:     Objetivo: Inicializa a Matriz Esparsa com o numero de linhas e colunas
40:     especificados. A linha 0 e a coluna 0 contem
41: }
42: var
43:     P, PAnterior : PNo;
44:     I : word;
45: begin
46:     New(Matriz);
47:     Matriz^.Linha := 0;
48:     Matriz^.Coluna := 0;
49:     Matriz^.ProxLinha := Matriz;
50:     Matriz^.ProxColuna := Matriz;
51:
52:     PAnterior := Matriz;
53:
54:     for I := 1 to NumColunas do
55:         begin
56:             New(P);
57:             P^.Linha := 0;
58:             P^.Coluna := I;

```

```

59:     PAnterior^.ProxColuna := P;
60:     P^.ProxLinha := P;
61:     P^.ProxColuna := Matriz;
62:     PAnterior := P
63: end;
64:
65: PAnterior := Matriz;
66:
67: for I := 1 to NumLinhas do
68: begin
69:     New(P);
70:     P^.Linha := I;
71:     P^.Coluna := 0;
72:     PAnterior^.ProxLinha := P;
73:     P^.ProxLinha := Matriz;
74:     P^.ProxColuna := P;
75:     PAnterior := P
76: end;
77: end;
78:
79:
80: function ObterAcima (var Matriz : Matriz_Esparsa; Linha, Coluna : word) : PNo;
81: {
82:     Objetivo: Retorna o No que esta acima da posicao especificada por Linha e
83:         Coluna
84: }
85: var
86:     PC, PL, PAnterior : PNo;
87: begin
88:     PC := Matriz;
89:     while PC^.Coluna < Coluna do
90:         PC := PC^.ProxColuna;
91:
92:     PAnterior := PC;
93:     PL := PC^.ProxLinha;
94:
95:     while (PL^.Linha < Linha) and (PL^.Linha <> 0) do
96:         begin
97:             PAnterior := PL;
98:             PL := PL^.ProxLinha
99:         end;
100:
101:     ObterAcima := PAnterior
102: end;
103:
104:
105: function ObterEsquerda (var Matriz : Matriz_Esparsa; Linha, Coluna : word) :
    PNo;
106: {
107:     Objetivo: Retorna o No que esta a esquerda da posicao especificada por
108:         Linha e
109:             Coluna
110: }
111: var
112:     PC, PL, PAnterior : PNo;
113: begin
114:     PL := Matriz;
115:     while PL^.Linha < Linha do
116:         PL := PL^.ProxLinha;
117:
118:     PAnterior := PL;

```

```

118:   PC := PL^.ProxColuna;
119:
120:   while (PC^.Coluna < Coluna) and (PC^.Coluna <> 0) do
121:   begin
122:       PAnterior := PC;
123:       PC := PC^.ProxColuna
124:   end;
125:
126:   ObterEsquerda := PAnterior
127: end;
128:
129:
130: procedure InserirDepois (PL, PC : PNo; Valor : Tipo_do_Dado);
131: {
132:   Objetivo: Cria um No e insere abaixo de PL e a direita de PC.
133:   O numero da linha do No inserido e igual a PC^.Linha
134:   e o numero da coluna igual a PL^.Coluna
135: }
136: var
137:   P : PNo;
138: begin
139:   New(P);
140:   P^.Valor := Valor;
141:
142:   P^.Linha := PC^.Linha;
143:   P^.Coluna := PL^.Coluna;
144:
145:   P^.ProxLinha := PL^.ProxLinha;
146:   P^.ProxColuna := PC^.ProxColuna;
147:
148:   PL^.ProxLinha := P;
149:   PC^.ProxColuna := P;
150: end;
151:
152:
153: function ObterNo(var Matriz : Matriz_Esparsa; Linha, Coluna : word) : PNo;
154: {
155:   Objetivo: Retorna o No contido na posicao (Linha e Coluna) especificada.
156:   Caso nao exista, retorna nil
157: }
158: var
159:   PAcima : PNo;
160: begin
161:   PAcima := ObterAcima(Matriz, Linha, Coluna);
162:
163:   { Testa se o existe um No na Linha e Coluna especificadas,
164:     retornando o seu valor caso verdadeiro }
165:   if (PAcima^.ProxLinha^.Linha = Linha) then
166:       ObterNo := PAcima^.ProxLinha
167:   else
168:       ObterNo := nil
169:   end;
170:
171:
172: procedure ApagarDepois (PL, PC : PNo);
173: {
174:   Objetivo: Apaga o No que esta a direita de PC e abaixo de PL, fazendo
175:   com que PC^.ProxColuna e PL^.ProxLinha apontem para os valores
176:   contidos no No a ser apagado
177: }
178: var

```

```

179:     P : PNo;
180: begin
181:     P := PC^.ProxColuna;
182:     PC^.ProxColuna := P^.ProxColuna;
183:     PL^.ProxLinha := P^.ProxLinha;
184:     Dispose(P)
185: end;
186:
187:
188: procedure Apagar (var Matriz : Matriz_Esparsa);
189: {
190:     Objetivo: Apaga toda a matriz da memoria
191: }
192: var
193:     PL, PC, PAnterior : PNo;
194: begin
195:
196:     { Apaga, linha por linha os elementos contidos na matriz esparsa }
197:     PL := Matriz^.ProxLinha;
198:     while PL^.Linha <> 0 do
199:     begin
200:         PC := PL^.ProxColuna;
201:
202:         while PC^.Coluna <> 0 do
203:         begin
204:             PAnterior := PC;
205:             PC := PC^.ProxColuna;
206:             Dispose(PAnterior);
207:         end;
208:
209:         PL := PL^.ProxLinha;
210:     end;
211:
212:     { Apagando a linha 0 }
213:     PC := Matriz^.ProxColuna;
214:     while PC^.Coluna <> 0 do
215:     begin
216:         PAnterior := PC;
217:         PC := PC^.ProxColuna;
218:         Dispose(PAnterior);
219:     end;
220:
221:     { Apagando a coluna 0 }
222:     PL := Matriz^.ProxLinha;
223:     while PL^.Linha <> 0 do
224:     begin
225:         PAnterior := PL;
226:         PL := PL^.ProxLinha;
227:         Dispose(PAnterior);
228:     end;
229:
230:     { Apaga o elemento [0,0] }
231:     Dispose(Matriz);
232:     Matriz := nil
233: end;
234:
235: function Obter (var Matriz : Matriz_Esparsa; Linha, Coluna : word;
236:     var Valor : Tipo_Do_Dado) : boolean;
237: {
238:     Objetivo: Obtem o Valor contido na posicao (Linha e Coluna)
239:     especificada da Matriz. Se o No existir, retorna true

```

```

240:         e Valor recebe o valor contido no No. Caso o No nao
241:         exista, retorna false e Valor fica inalterado
242:     }
243:     var P : PNo;
244:     begin
245:         P := ObterNo(Matriz, Linha, Coluna);
246:         if P = nil then
247:             Obter := false
248:         else
249:             begin
250:                 Obter := true;
251:                 Valor := P^.Valor;
252:             end;
253:         end;
254:
255:
256:     procedure Mudar (var Matriz : Matriz_Esparsa; Linha, Coluna : word;
257:         Valor : Tipo_Do_Dado);
258:     {
259:         Objetivo: Faz com que exista um No com o valor passado na
260:         posicao (Linha e Coluna) especificada
261:     }
262:     var PAcima : PNo;
263:     begin
264:         PAcima := ObterAcima(Matriz, Linha, Coluna);
265:
266:         { Verifica se ja existe um No na posicao (Linha, Coluna) especificada.
267:         Se existir, seu valor sera modificado. Caso contrario sera inserido
268:         nessa posicao um novo No contendo o Valor }
269:         if PAcima^.ProxLinha^.Linha = Linha then
270:             PAcima^.ProxLinha^.Valor := Valor
271:         else
272:             InserirDepois(PAcima, ObterEsquerda(Matriz, Linha, Coluna), Valor);
273:         end;
274:
275:
276:     procedure Limpar (var Matriz : Matriz_Esparsa; Linha, Coluna : word);
277:     {
278:         Objetivo: Apaga da Matriz o No existente na posicao especificada por
279:         Linha e Coluna
280:     }
281:     var PAcima : PNo;
282:     begin
283:         PAcima := ObterAcima(Matriz, Linha, Coluna);
284:
285:         { Testa se existe um No na posicao especificada. Caso exista,
286:         Apaga o No }
287:         if PAcima^.ProxLinha^.Linha = Linha then
288:             ApagarDepois(PAcima, ObterEsquerda(Matriz, Linha, Coluna));
289:         end;
290:
291:     end.

```