

Continuação do Programa 7.5

```

begin { RetiraAresta }
  AuxAnterior := Grafo.Adj[V1].Primeiro;
  Aux := Grafo.Adj[V1].Primeiro.Prox;
  EnquantoAresta := false;
  while (Aux <> nil) and (EncontrouAresta = false) do
    begin
      if V2 = Aux.Item.Vertex
      then begin
        Retira (AuxAnterior, Grafo.Adj[V1], x);
        Grafo.NumArestas := Grafo.NumArestas - 1;
        EncontrouAresta := true;
      end;
      AuxAnterior := Aux; Aux := Aux.Prox;
    end;
  end; { RetiraAresta }

procedure LiberaGrafo (var Grafo: TipoGrafo);
var AuxAnterior, Aux: TipoApontador;
begin
  for i := 0 to Grafo.NumVertices-1 do
    begin
      Aux := Grafo.Adj[i].Primeiro.Prox;
      dispose (Grafo.Adj[i].Primeiro); { Libera célula cabeça }
      while Aux <> nil do
        begin
          AuxAnterior := Aux; Aux := Aux.Prox; dispose (AuxAnterior);
        end;
      end; { LiberaGrafo }
    end;

procedure ImprimeGrafo (var Grafo: TipoGrafo);
var i: integer; Aux: TipoApontador;
begin
  for i := 0 to Grafo.NumVertices-1 do
    begin
      write ('Vertex', i:2, ':');
      if not Vazia (Grafo.Adj[i])
      then begin
        Aux := Grafo.Adj[i].Primeiro.Prox;
        while Aux <> nil do
          begin
            write (Aux.Item.Vertex:3, ' (', Aux.Item.Peso, ')');
            Aux := Aux.Prox;
          end;
        end;
        writeln;
      end;
    end; { ImprimeGrafo }
  end;

```

7.2.3 Implementação por meio de Listas de Adjacência Usando Arranjos

As Figuras 7.9(a) e 7.9(b) apresentam a representação para listas de adjacência usando arranjos para um grafo direcionado contendo quatro vértices e três arestas e para um grafo não direcionado contendo quatro vértices e duas arestas, respectivamente. Note que cada aresta é representada duas vezes no grafo não direcionado.

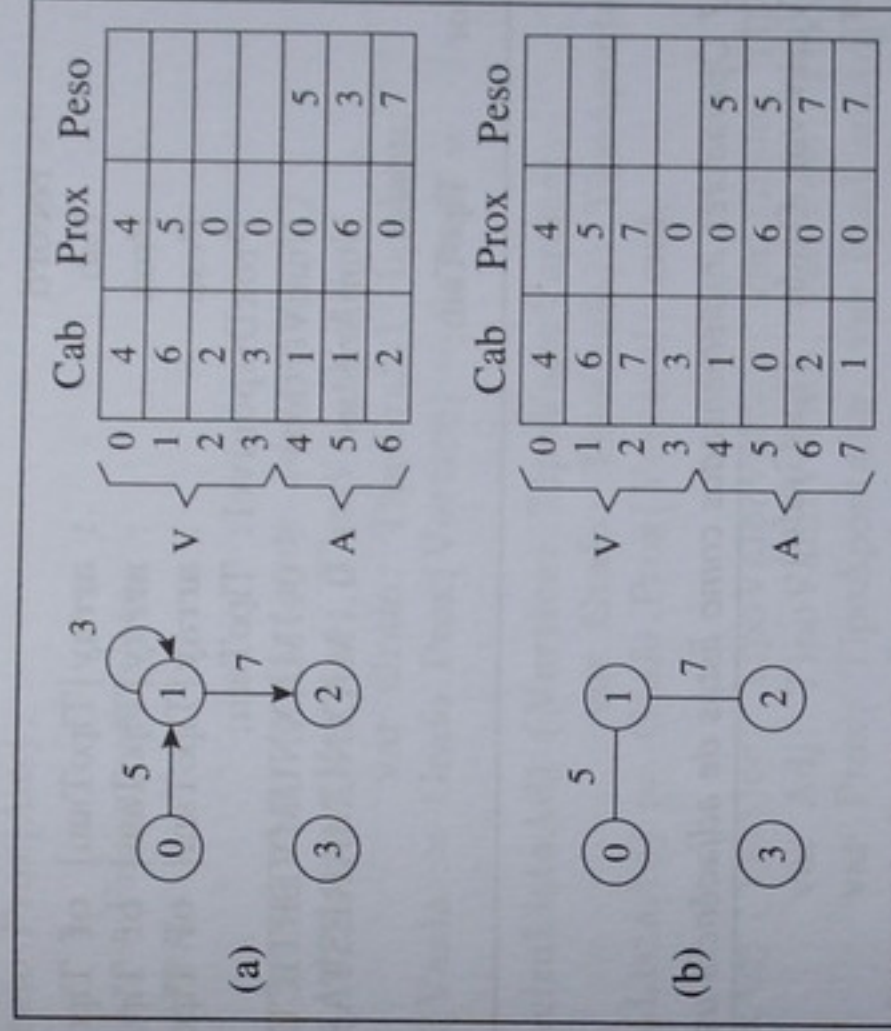


Figura 7.9 Representação para listas de adjacência usando arranjos. (a) Grafo direcionado; (b) Grafo não direcionado.

O Programa 7.6 mostra a estrutura de dados usada. O registro TipoGrafo contém três arranjos de dimensões entre 0 e $|V| + 2 \times |A|$ cada um: (i) o arranjo Cab, cujas $|V|$ primeiras posições contêm os endereços do último item da lista de adjacentes de cada vértice e as $|A|$ últimas posições contêm os vértices propriamente ditos, (ii) o arranjo Prox, que contém o endereço do próximo item da lista de adjacentes e (iii) o arranjo Peso, o qual contém, nas últimas $|A|$ posições, o valor do peso de cada aresta do grafo. A variável ProxDisponível contém a próxima posição disponível para inserção de uma nova aresta. As duas variáveis seguintes, NumVertices e NumArestas, contêm o número de vértices e o número de arestas do grafo, respectivamente.

Uma possível implementação para as operações definidas anteriormente é mostrada no Programa 7.7.