Continuação do Programa 6.9

```
    else begin { Aux = Pagina a esquerda de ApPag }
         Aux := ApPai^.p[PosPai-1];
         DispAux := (Aux^.n - M + 1) div 2;
         for j := ApPag^.n downto 1 do
           ApPag^.r[j+1] := ApPag^.r[j];
         ApPag^.r[1] := ApPai^.r[PosPai];
         for j := ApPag^.n downto 0 do
           ApPag^.p[j+1] := ApPag^.p[j];
         ApPag^.n := ApPag^.n + 1;
         if DispAux > 0
         then begin { Existe folga: transfere de Aux para ApPag }
              for j := 1 to DispAux - 1 do with Aux^ do
                InsereNaPagina (ApPag, r[Aux^.n+1-j], p[n+1-j]);
              ApPag^.p[0] := Aux^.p[Aux^.n+1-DispAux];
              ApPai^.r[PosPai] := Aux^.r[Aux^.n+1-DispAux];
              Aux^.n := Aux^.n - DispAux;
              Diminuiu := false
              end
         else begin { Fusao: intercala ApPag em Aux e libera ApPag }
              for j := 1 to M do
                InsereNaPagina (Aux, ApPag^.r[j], ApPag^.p[j]);
              dispose (ApPag);
              ApPai^.n := ApPai^.n - 1;
              if ApPai^.n >=M then Diminuiu := false;
              end;
         end;
    end;
end; { Reconstitui }

procedure Antecessor (Ap: TipoApontador; Ind: Integer;
                      ApPai: TipoApontador;
                      var Diminuiu: Boolean);
begin
with ApPai^ do
  begin
  if p[n] <> nil
  then begin
       Antecessor (Ap, Ind, p[n], Diminuiu);
       if Diminuiu then Reconstitui (p[n], ApPai, n, Diminuiu);
       end
  else begin
       Ap^.r[Ind] := r[n];  n := n - 1;
       Diminuiu := n < M;
       end;
  end
end; { Antecessor }
```


Continuação do Programa 6.9

```
begin { Ret }
  if Ap = nil
  then begin
       writeln ('Erro: registro nao esta na arvore');
       Diminuiu := false;
       end
  else with Ap^ do
       begin
       Ind := 1;
       while (Ind < n) and (Ch > r[Ind].Chave) do Ind := Ind + 1;
       if Ch = r[Ind].Chave
       then if p[Ind-1] = nil
            then begin { Pagina folha }
            n := n - 1; Diminuiu := n < M;
            for j := Ind to n do
              begin
              r[j] := r[j+1];
              p[j] := p[j+1];
              end;
            end
       else begin { Pagina nao e folha: trocar com antecessor }
            Antecessor (Ap, Ind, p[Ind-1], Diminuiu);
            if Diminuiu
            then Reconstitui (p[Ind-1], Ap, Ind-1, Diminuiu);
            end
       else begin
            if Ch > r[Ind].Chave then Ind := Ind + 1;
            Ret (Ch, p[Ind-1], Diminuiu);
            if Diminuiu
            then Reconstitui (p[Ind-1], Ap, Ind-1, Diminuiu);
            end
       end
  end; { Ret }

begin { Retira }
  Ret (Ch, Ap, Diminuiu);
  if Diminuiu and (Ap^.n = 0)
  then begin { Arvore diminui na altura }
       Aux := Ap; Ap := Aux^.p[0];
       dispose (Aux);
       end
end; { Retira }
```

A Figura 6.13 mostra o resultado obtido quando se retira a seguinte sequência de chaves da árvore B: 45 30 28; 50 8 10 4 20 40 55 17 33 11 36; 3 9 52. Cada ponto e vírgula corresponde a um salto de uma árvore para outra no desenho da Figura 6.13.