```
unit Drzewo;

        interface

   type
        TDane = string[20];
     PDrzewo = ^TDrzewo;
     TDrzewo = record
                                                    dane       : TDane;
               krotnosc   : integer;
               wskaznik_l : PDrzewo;
               wskaznik_p : PDrzewo
                                    end;

          var
        skladnik_biezacy, pierwszy_skladnik, zmp2 : PDrzewo;

   procedure EnTree(dane:TDane; var skladnik_biezacy:PDrzewo);

   procedure SearchComponentTree(var skladnik_biezacy:PDrzewo;
pierwszy_skladnik:PDrzewo;

                                    dane:TDane; zmp:integer; var koniec:Boolean);

   procedure DeTree(var skladnik_biezacy:PDrzewo; var pierwszy_skladnik:PDrzewo;
                                                                    dane:TDane;
zmp:integer; tak_nie:Boolean; var koniec:Boolean);

   procedure DeleteTree(var skladnik_biezacy:PDrzewo; var pierwszy_skladnik:PDrzewo;

zmp:integer);

  implementation

              procedure EnTree(dane:TDane; var skladnik_biezacy:PDrzewo);
   begin
        if skladnik_biezacy <> nil then
        begin
       if skladnik_biezacy^.dane = dane then
        Inc(skladnik_biezacy^.krotnosc)
       else
        begin
              if dane <= skladnik_biezacy^.dane then
              EnTree(dane, skladnik_biezacy^.wskaznik_l)
         else
              EnTree(dane, skladnik_biezacy^.wskaznik_p)
        end
      end
```

```
    else
        begin
        New(skladnik_biezacy);
       skladnik_biezacy^.dane := dane;
       skladnik_biezacy^.krotnosc := 1;
       skladnik_biezacy^.wskaznik_l := nil;
                                          skladnik_biezacy^.wskaznik_p := nil
     end
  end;

  procedure Szukaj(skladnik_biezacy:PDrzewo; var pp:PDrzewo);
  begin
    if skladnik_biezacy^.wskaznik_l = nil then
      begin
        pp := skladnik_biezacy
      end
    else
        Szukaj(skladnik_biezacy^.wskaznik_l, pp)
  end;

  procedure DeTree(var skladnik_biezacy:PDrzewo; var pierwszy_skladnik:PDrzewo;
                                                                    dane:TDane;
zmp:integer; tak_nie:Boolean; var koniec:Boolean);
  var
        p, pp, z : PDrzewo;
    zmp2     : Boolean;
  begin
        Inc(zmp);
    koniec := false;
    if (pierwszy_skladnik <> nil) and (zmp = 1) then
                skladnik_biezacy := pierwszy_skladnik
    else
        if pierwszy_skladnik = nil then
       begin
                        WriteLn('Drzewo jest puste');
        WriteLn('Nie mozna usunac elementu');
        exit
                end;
    if skladnik_biezacy^.dane = dane then
        begin
        if (skladnik_biezacy^.wskaznik_l = nil) and (skladnik_biezacy^.wskaznik_p = nil) then
        begin
                if pierwszy_skladnik = skladnik_biezacy then
          tak_nie := true;
         Dispose(skladnik_biezacy);
         skladnik_biezacy := nil;
         if tak_nie then
                pierwszy_skladnik := nil
```

```
                                                    end
                        else
                            begin
                            if skladnik_biezacy^.wskaznik_l = nil then
                        begin
                     if pierwszy_skladnik = skladnik_biezacy then
                            tak_nie := true;
                        p := skladnik_biezacy;
                                    skladnik_biezacy := skladnik_biezacy^.wskaznik_p;
                        Dispose(p);
                     p := nil;
                     if tak_nie then
                        pierwszy_skladnik := skladnik_biezacy
                        end
                                else
                  begin
                                            if skladnik_biezacy^.wskaznik_p = nil then
                        begin
                        if pierwszy_skladnik = skladnik_biezacy then
                                            tak_nie := true;
                        p := skladnik_biezacy;
                                                    skladnik_biezacy :=
skladnik_biezacy^.wskaznik_l;
                       Dispose(p);
                       p := nil;
                       if tak_nie then
                        pierwszy_skladnik := skladnik_biezacy
                      end
                     else
                        begin
                        if skladnik_biezacy^.wskaznik_p^.wskaznik_l = nil then
                        begin
                            if pierwszy_skladnik = skladnik_biezacy then
                                                        tak_nie := true;
                            p := skladnik_biezacy;
                      skladnik_biezacy := p^.wskaznik_p;
                      skladnik_biezacy^.wskaznik_l := p^.wskaznik_l;
                      Dispose(p);
                      p := nil;
                      if tak_nie then
                            pierwszy_skladnik := skladnik_biezacy
                        end
                       else
                        begin
                            if pierwszy_skladnik = skladnik_biezacy then
                                                        tak_nie := true;
                            Szukaj(skladnik_biezacy^.wskaznik_p^.wskaznik_l, pp);
                      p := skladnik_biezacy;
```

```
                    z := pp;
                    pp := pp^.wskaznik_p;
                            z^.wskaznik_p := skladnik_biezacy^.wskaznik_p;
                    z^.wskaznik_l := skladnik_biezacy^.wskaznik_l;
                    skladnik_biezacy := z;
                    Dispose(p);
                    p := nil;
                    if tak_nie then
                            pierwszy_skladnik := skladnik_biezacy
                  end
                end
              end
                    end;
          WriteLn('Element usuniety');
         koniec := true;
          exit
       end
     else
          begin
         if (dane < skladnik_biezacy^.dane) and (skladnik_biezacy^.wskaznik_l <> nil) then
          DeTree(skladnik_biezacy^.wskaznik_l, pierwszy_skladnik, dane, zmp, false, koniec)
         else
          if skladnik_biezacy^.wskaznik_p <> nil then
                  DeTree(skladnik_biezacy^.wskaznik_p, pierwszy_skladnik, dane, zmp, false,
koniec)
       end;
     if (zmp = 1) and (koniec = false) then
          WriteLn('Nie ma takiego elementu')
   end;

   procedure DeleteTree(var skladnik_biezacy:PDrzewo; var pierwszy_skladnik:PDrzewo;

zmp:integer);
   begin
          Inc(zmp);
     if (pierwszy_skladnik <> nil) and (zmp = 1) then
       begin
                        skladnik_biezacy := pierwszy_skladnik
       end;
     if skladnik_biezacy <> nil then
                  begin
                                        DeleteTree(skladnik_biezacy^.wskaznik_l,
pierwszy_skladnik, zmp);
                                        DeleteTree(skladnik_biezacy^.wskaznik_p,
pierwszy_skladnik, zmp);
         Dispose(skladnik_biezacy);
         skladnik_biezacy := nil
               end
```

```
   end;

   procedure SearchComponentTree(var skladnik_biezacy:PDrzewo;
pierwszy_skladnik:PDrzewo;

                                          dane:TDane; zmp:integer; var koniec:Boolean);
   begin
     koniec := false;
          Inc(zmp);
          if (pierwszy_skladnik <> nil) and (zmp = 1) then
                    skladnik_biezacy := pierwszy_skladnik
     else
          if pierwszy_skladnik = nil then
        begin
                         WriteLn('Drzewo jest puste');
          exit
                    end;
     if skladnik_biezacy^.dane = dane then
       begin
          koniec := true;
        WriteLn('Element znaleziony');
                    exit
       end
     else
       begin
                    if (skladnik_biezacy^.wskaznik_l <> nil) and (dane <=
skladnik_biezacy^.dane) then
          begin
                    skladnik_biezacy := skladnik_biezacy^.wskaznik_l;
                    SearchComponentTree(skladnik_biezacy, pierwszy_skladnik, dane, zmp,
koniec)
          end
                                          else
          begin
                    if (skladnik_biezacy^.wskaznik_p <> nil) and (dane >=
skladnik_biezacy^.dane) then
              begin
                    skladnik_biezacy := skladnik_biezacy^.wskaznik_p;
                              SearchComponentTree(skladnik_biezacy, pierwszy_skladnik,
dane, zmp, koniec)
              end
          end
       end;
     if (zmp = 1) and (koniec = false) then
                         Writeln('Nie ma takiego elementu')
   end;
begin
          skladnik_biezacy := nil;
```

```
  pierwszy_skladnik := nil;
end.
```