```
{
Autor : Slawek Kolasinski
Data  : 03.2009
e-mail: skola@mimuw.edu.pl
}

Program Ex20090330;
type
   typ     = real;
   drzewo = ^wezel;
   wezel   = record
                wartosc : typ;
                lewy    : drzewo;
                prawy   : drzewo;
             end;

function max(x,y : integer) : integer;
begin
   if x > y then
      max := x
   else
      max := y;
end; { max }

procedure inicjuj(var t : drzewo);
begin
   t := nil;
end; { inicjuj }

function puste(t : drzewo) : boolean;
begin
   puste := (t = nil);
end; { puste }

procedure dodaj(var t : drzewo; x : typ);
begin
   if puste(t) then begin
      new(t);
      t^.wartosc := x;
      { zainicjuj lewe i prawe poddrzewo, _…eby byly puste }
      inicjuj(t^.lewy);
      inicjuj(t^.prawy);
   end else if x < t^.wartosc then
      dodaj(t^.lewy, x)
   else
      dodaj(t^.prawy, x);
end; { dodaj }

procedure kasuj(var t : drzewo);
begin
   if not puste(t) then begin
      kasuj(t^.lewy);
      kasuj(t^.prawy);
      dispose(t);
      t := nil;
   end;
end; { kasuj }

procedure predfs_print(t : drzewo);
begin
```



```
   if puste(t) then exit;
   write(t^.wartosc:2:2, ' ');
   predfs_print(t^.lewy);
   predfs_print(t^.prawy);
end; { predfs_print }

function suma(t : drzewo) : typ;
begin
   if puste(t) then suma := 0
   else suma := t^.wartosc + suma(t^.lewy) + suma(t^.prawy);
end; { suma }

function suma_ilosc(t : drzewo; var ilosc : integer) : typ;
var
   il1,il2 : integer;
begin
   if puste(t) then begin
      suma_ilosc := 0;
      ilosc := 0;
   end else begin
      suma_ilosc := suma_ilosc(t^.lewy, il1) + suma_ilosc(
                                          t^.prawy, il2) + t^.wartosc;
      ilosc := il1 + il2 + 1;
   end;
end; { srednia_pom }

function srednia(t : drzewo) : typ;
var
   s     : typ;
   ilosc : integer;
begin
   if puste(t) then srednia := 0
   else begin
      s := suma_ilosc(t, ilosc);
      srednia := s / ilosc;
   end;
end; { srednia }

function wysr_pom(t : drzewo; var suma : typ;
                                       var ilosc : integer) : typ;
var
   s1,s2,wyn : typ;
   il1,il2   : integer;
begin
   if puste(t) then begin
      wysr_pom := 0;
      suma := 0;
      ilosc := 0;
   end else begin
      wyn := wysr_pom(t^.lewy, s1, il1) + wysr_pom(
                                                t^.prawy, s2, il2);
      suma := s1 + s2 + t^.wartosc;
      ilosc := il1 + il2 + 1;
      wysr_pom := wyn + (t^.wartosc - (suma / ilosc));
   end;
end; { wysr_pom }

function wysrodkowanie(t : drzewo) : typ;
var
   s  : typ;
```

```
118     il : integer;
119  begin
120     if puste(t) then wysrodkowanie := 0
121     else wysrodkowanie := wysr_pom(t,s,il);
122  end; { wysrodkowanie }

124  function ciezar_pom(t : drzewo; poziom : integer) : typ;
125  begin
126     if puste(t) then ciezar_pom := 0
127     else begin
128        ciezar_pom := (t^.wartosc * poziom) + ciezar_pom(
             t^.lewy, poziom + 1) + ciezar_pom(t^.prawy, poziom + 1);
129     end;
130  end; { ciezar_pom }

132  function ciezar(t : drzewo) : typ;
133  begin
134     ciezar := ciezar_pom(t,1);
135  end; { ciezar }

137  var
138     t1 : drzewo;

140  begin
141     inicjuj(t1);

143     dodaj(t1, 6); dodaj(t1, 3); dodaj(t1, 10); dodaj(t1, 1);
                                                      dodaj(t1, 12);
144     dodaj(t1, 2); dodaj(t1, 11); dodaj(t1, 4); dodaj(t1, 8);
                                                       dodaj(t1, 5);
145     dodaj(t1, 9); dodaj(t1, 13); dodaj(t1, 0); dodaj(t1, 7);

147     write('t1: '); predfs_print(t1); writeln;

149     writeln('suma t1: ', suma(t1):2:2);
150     writeln('srednia t1: ', srednia(t1):2:2);
151     writeln('wysrodkowanie t1: ', wysrodkowanie(t1):2:2);
152     writeln('ciezar t1: ', ciezar(t1):2:2);

154     kasuj(t1);
155     write('po skasowaniu t1: '); predfs_print(t1); writeln;
156  end.
```