

```

1  {
2  Autor : Slawek Kolasinski
3  Data  : 04.2009
4  e-mail: skola@mimuw.edu.pl
5  }
6  Program Graph;
7  const
8    N = 26;
9  type
10 { typ danych w grafie }
11   gtyp = record
12     odwiedzony : boolean;
13     numer       : integer;
14     odl         : integer;
15   end;
16 { typ danych na liscie }
17   ltyp = integer;
18   lista = ^element;
19   wezel = record
20     dane      : gtyp;
21     sasiedzi : lista;
22   end;
23   element = record
24     glowa : ltyp;
25     ogon  : lista;
26   end;
27   graf = array[1..N] of wezel;
28   stos = lista;
29   kolejka = record
30     pierwszy : lista;
31     ostatni  : lista;
32   end;
33
34 (***** OBSLUGA OPERACJI NA LISTACH *****)
35 procedure l_inicjuj(var l : lista);
36 begin
37   l := nil;
38 end; { l_inicjuj }
39
40 function l_pusta(l : lista) : boolean;
41 begin
42   l_pusta := (l = nil);
43 end; { l_pusta }
44
45 function l_glowa(l : lista) : ltyp;
46 begin
47   if not l_pusta(l) then
48     l_glowa := l^.glowa
49   else begin
50     writeln('glowa: pusta lista!');
51     halt;
52   end;
53 end; { l_glowa }
54
55 function l_ogon(l : lista) : lista;
56 begin
57   if not l_pusta(l) then
58     l_ogon := l^.ogon
59   else begin
60     writeln('ogon: pusta lista!');

```

```

61   halt;
62   end;
63 end; { l_ogon }
64
65 function l_usun(var l : lista) : ltyp;
66 var
67   czubek : lista;
68 begin
69   l_usun := l_glowa(l);
70   czubek := l;
71   l := l_ogon(l);
72   dispose(czubek);
73 end; { l_usun }
74
75 procedure l_kasuj(var l : lista);
76 var
77   ignoruj : ltyp;
78 begin
79   while not l_pusta(l) do ignoruj := l_usun(l);
80 end; { l_kasuj }
81
82 procedure l_drukuj(l : lista);
83 begin
84   if not l_pusta(l) then begin
85     write(l_glowa(l));
86     l := l_ogon(l);
87   end;
88   while not l_pusta(l) do begin
89     write(' ', l_glowa(l));
90     l := l_ogon(l);
91   end;
92 end; { l_drukuj }
93
94 procedure l_dodaj_przed(var l : lista; x : ltyp);
95 var
96   elt : lista;
97 begin
98   new(elt);
99   elt^.glowa := x;
100  elt^.ogon := l;
101  l := elt;
102 end; { l_dodaj_przed }
103
104 procedure l_dodaj_za(var l : lista; x : ltyp);
105 var
106   elt : lista;
107 begin
108   if l_pusta(l) then l_dodaj_przed(l,x)
109   else begin
110     new(elt);
111     elt^.glowa := x;
112     elt^.ogon := l^.ogon;
113     l^.ogon := elt;
114   end;
115 end; { l_dodaj_za }
116
117 function l_znajdz(l : lista; x : ltyp) : boolean;
118 begin
119   while (not l_pusta(l)) and (l_glowa(l) <> x) do l := l_ogon(

```

```

120   l_znajdz := not l_pusta(l);
121   end; { l_znajdz }

123   function l_dlugosc(l : lista) : integer;
124   var
125     licznik : integer;
126   begin
127     licznik := 0;
128     while not l_pusta(l) do begin
129       licznik := licznik + 1;
130       l := l_ogon(l);
131     end;
132     l_dlugosc := licznik;
133   end; { l_dlugosc }

135   (***** OBSLUGA OPERACJI NA STOSIE *****)
136   procedure s_inicjuj(var s : stos);
137   begin
138     l_inicjuj(s);
139   end; { s_inicjuj }

141   procedure s_kasuj(var s :stos);
142   begin
143     l_kasuj(s);
144   end; { s_kasuj }

146   procedure s_push(var s : stos; x : ltyp);
147   begin
148     l_dodaj_przed(s,x);
149   end; { s_push }

151   function s_pop(var s : stos) : ltyp;
152   begin
153     s_pop := l_usun(s);
154   end; { s_pop }

156   function s_top(s : stos) : ltyp;
157   begin
158     s_top := l_glowa(s);
159   end; { s_top }

161   function s_pusty(s : stos) : boolean;
162   begin
163     s_pusty := l_pusta(s);
164   end; { s_pusty }

166   (***** OBSLUGA OPERACJI NA KOLEJCE *****)
167   procedure q_inicjuj(var q : kolejka);
168   begin
169     l_inicjuj(q.pierwszy);
170     q.ostatni := q.pierwszy;
171   end; { q_inicjuj }

173   procedure q_kasuj(var q : kolejka);
174   begin
175     l_kasuj(q.pierwszy);
176     q.ostatni := q.pierwszy;
177   end; { q_kasuj }

179   function q_pusta(q : kolejka) : boolean;

```

```

180   begin
181     q_pusta := l_pusta(q.pierwszy) and l_pusta(q.ostatni);
182   end; { q_pusta }

184   procedure q_enqueue(var q : kolejka; x : ltyp);
185   begin
186     if q_pusta(q) then begin
187       l_dodaj_przed(q.pierwszy,x);
188       q.ostatni := q.pierwszy;
189     end else begin
190       l_dodaj_za(q.ostatni,x);
191       q.ostatni := l_ogon(q.ostatni);
192     end;
193   end; { q_push }

195   function q_dequeue(var q : kolejka) : ltyp;
196   begin
197     q_dequeue := l_usun(q.pierwszy);
198     if l_pusta(q.pierwszy) then q.ostatni := q.pierwszy;
199   end; { q_pop }

201   function q_top(q : kolejka) : ltyp;
202   begin
203     q_top := l_glowa(q.pierwszy);
204   end; { q_top }

206   (***** OBSLUGA OPERACJI NA GRAFACH *****)
207   procedure g_inicjuj(var g : graf);
208   var
209     i : integer;
210   begin
211     for i := 1 to N do begin
212       l_inicjuj(g[i].sasiedzi);
213       g[i].dane.odwiedzony := false;
214       g[i].dane.odl := 0;
215       g[i].dane.numer := i;
216     end;
217   end; { g_inicjuj }

219   procedure g_zeruj_odwiedzone(var g : graf);
220   var
221     i : integer;
222   begin
223     for i := 1 to N do begin
224       g[i].dane.odwiedzony := false;
225     end;
226   end; { g_zeruj_odwiedzone }

228   procedure g_kasuj(var g : graf);
229   var
230     u : integer;
231   begin
232     for u := 1 to N do begin
233       l_kasuj(g[u].sasiedzi);
234     end;
235   end; { g_kasuj }

237   function g_jest_krawedz(var g : graf; u,v : integer) : boolean;
238   begin
239     g_jest_krawedz := l_znajdz(g[u].sasiedzi, v);

```

```

240 end; { g_jest_krawedz }
242 procedure g_dodaj_krawedz(var g : graf; u,v : integer);
243 begin
244     l_dodaj_przed(g[u].sasiedzi, v);
245 end; { g_dodaj_krawedz }
247 procedure g_dodaj_krawedz2(var g : graf; u,v : integer);
248 begin
249     g_dodaj_krawedz(g,u,v);
250     if u <> v then g_dodaj_krawedz(g,v,u);
251 end; { g_dodaj_krawedz2 }
253 function g_stopien(var g : graf; u : integer) : integer;
254 begin
255     g_stopien := l_dlugosc(g[u].sasiedzi);
256 end; { g_stopien }
258 procedure g_drukuj(var g : graf);
259 var
260     u : integer;
261 begin
262     for u := 1 to N do begin
263         write(u, '(' ,g_stopien(g,u), '): ');
264         l_drukuj(g[u].sasiedzi);
265         writeln;
266     end;
267 end; { g_drukuj }
269 procedure g_z_pliku2(nazwa : string; var g : graf);
270 var
271     plik : text;
272     u,v : integer;
273 begin
274     assign(plik,nazwa);
275     reset(plik);
276     g_inicjuj(g);
277     while not eof(plik) do begin
278         readln(plik, u, v);
279         if (1 <= u) and (1 <= v) and (u <= N) and (
                v <= N) then begin
280             if (not g_jest_krawedz(g,u,v)) then g_dodaj_krawedz2(
                    g,u,v);
281             end else begin
282                 writeln(
                    'g_z_pliku2: Indeks wierzcholka poza zakresem!');
283                 halt;
284             end;
285         end;
286         close(plik);
287     end; { g_z_pliku2 }
289 function max(a,b : integer) : integer;
290 begin
291     if a < b then max := b else max := a;
292 end; { max }
294 procedure g_srednica_pom(var g : graf; u : integer;
295     var max1,max2 : integer);

```

```

296     l : lista;
297     v : integer;
298     m1,m2 : integer;
299 begin
300     g[u].dane.odwiedzony := true;
301     max1 := 0;
302     max2 := 0;
303     l := g[u].sasiedzi;
304     while not l_pusta(l) do begin
305         v := l_glowa(l);
306         if not g[v].dane.odwiedzony then begin
307             g_srednica_pom(g,v,m1,m2);
308             m1 := m1 + 1;
309             if m1 >= max1 then begin
310                 max2 := max1;
311                 max1 := m1;
312             end else if m1 > max2 then begin
313                 max2 := m1
314             end;
315             end;
316             l := l_ogon(l);
317         end;
318     end; { g_srednica_pom }
320 function g_srednica(var g : graf) : integer;
321 var
322     m1,m2 : integer;
323 begin
324     g_zeruj_odwiedzone(g);
325     g_srednica_pom(g,1,m1,m2);
326     g_srednica := m1+m2;
327 end; { g_srednica }
329 var
330     d : graf;
331 begin
332     g_z_pliku2('drzewo.txt',d);
334     writeln('drzewo');
335     g_drukuj(d);
337     writeln('srednica d = ', g_srednica(d));
339     g_kasuj(d);
340 end.

```