```
   unit moretext;


   interface
       uses
MemTypes,QuickDraw,OSIntf,ToolIntf,PackIntf,PrintTraps,PasLibIntf
,
       globals,help;


    procedure text150(i,full:longint);
    procedure text300(i,full:longint);

implementation


procedure text75(i,full:longint);
begin
  case i of
1: write('edges <= (nodes-1)*(nodes-2)/2+nconn');
2: write('chr <= (ncov+clique+1)/2');
3: write('spectr >= 2*edges/nodes');
4: write('spectr <= SQRT[2*edges*ncov/(ncov+1)]');
5: write('clique >= nodes**2/(nodes**2-2*edges)');
6: write('spectr <= maxdeg');
7: if full = 1 then write('nodes >= ...more...')
   else
    begin
     writeln('nodes >= nconn*(diam-3)+2*mindeg+2, always and');
     if full <> 0 then write('         ');
     write('      ');
     writeln('        >= 1+mindeg+diam*nconn+FL[diam/3]*(mindeg-3
*nconn+1)');
     if full <> 0 then write('         ');
     write('                  when mindeg > 3*nconn-1');
   end;
8: write('nodes >= maxdeg+1+(mindeg+1)*(ncomp-1)');
9: write('eccov <= nodes**2/4');
10: write('diam <= 2*radius');
11: write('eind <= nodes/2');
12: write('eind >= nodes/(maxdeg+1)');
13: write('if girth = 2*diam+1 then regular      { retired }');
14: write('chr >= nodes/(nodes-spectr)           { retired }');
15: write('if mindeg >= 3 then girth <= 2*LOG2[edges-
nodes+ncomp]');
16: write('nconn=0 iff econn=0');
17: write('edges <= eccov*clique(clique-1)/2');
18: write('chr <= (7+SQRT[48*genus+1])/2');
19: write('if clique = 2 then maxdeg <= nind and edges <=
```

```
ncov*nind');
20: if full = 1 then write('if chr = 2 then ...more...')
      else
    begin
      writeln('if chr = 2 then eind = ncov, nind = nccov, echr =
maxdeg,');
      if full <> 0 then write('         ');
      write('   ');
      write('         girth and circ are even, and, if nodes > 2,
not complete');
    end;
21: write('genus <= CL[(nodes-3)*(nodes-4)/12]');
22: write('if edges > maxdeg*eind then echr = maxdeg+1');
23: write('eccov <= ecov*eind');
24: if full = 1 then write('if mindeg >= 3 then ...more...')
    else
    begin
     writeln('if mindeg >= 3 then nodes');
     if full <> 0 then write('         ');
     write('    ');
     writeln('       >= (mindeg*(mindeg-1)**k-2)/(mindeg-2), if
girth = 2*k+1');
     if full <> 0 then write('         ');
     write('   ');
     write('        >= (2*(mindeg-1)**k-2)/(mindeg-2),      if
girth = 2*k');
   end;
25: write('ecov >= nodes/2');
26: write('edges <= (nodes-1)*(nodes-2)/2+nodes/dom-1');
27: write('ecov <= nodes*maxdeg/(maxdeg+1)');
28: if full = 1 then write('maxdeg <= ...more...')
     else
      begin
      writeln('maxdeg <= nodes-nconn*(diam-4)-3,   if diam >=
4');
      if full <> 0 then write('         ');
      write('               <= nodes-diam+1,
otherwise');
    end;
29: write('eccov <= edges-clique*(clique-1)/2+1');
30: if full = 1 then write('if connct then edges <= ...more...')
     else
       begin
        writeln('if connct then edges');
        if full <> 0 then write('         ');
        writeln('          <= nodes*(nodes-radius)/2, for 1 <=
radius <= MIN[2, nodes/2]');
        if full <> 0 then write('         ');
        writeln('          <= (nodes**2+5*nodes-4*radius*nodes+4
*radius**2-6*radius)/2,');
        if full <> 0 then write('         ');
        write('                             for 3 <= radius <=
```

```
nodes/2');
      end;
31: write('chr <= (nodes+1)**2/(4*nccov)');
32: write('chr >= 2*SQRT[nodes]-nccov');
33: write('dom <= nodes+1-SQRT[2*edges+1]');
34: write('if connct and not a tree then girth <= 2*diam+1');
35: write('if planar and clique = 2 then nind >= (nodes+1)/3');
36: if full = 1 then write('if not planar then ...more...')
     else
     begin
      writeln('if not planar then maxdeg >= 3, nodes >= 5,
edges >= 9,');
      if full <> 0 then write('         ');
      write('   ');
      write('           eind >= 2, ncov >= 3, ecov >= 3,
bwidth >= 4');
    end;
37: write('edges <= ncomp-1+(nodes+2-2*ncomp)*(nodes+1-2
*ncomp)/2');
38: write('dom >= nodes/(maxdeg+1)');
39: write('clique <= 2 iff girth >= 4');
40: if full = 1 then write('complete iff ...more...')
    else
      begin
      writeln('complete iff mindeg = nodes-1 iff nind = 1');
      if full <> 0 then write('         ');
      write('    ');
      write('             iff nccov = 1 iff eccov = 1 iff diam =
1');
    end;
41: write('bipartite iff chr <= 2');
42: write('radius = 1 iff maxdeg = nodes-1');
43: write('tree iff forest and connected');
44: if full = 1 then write('connct iff...more...')
     else
      begin
      writeln('connct iff nconn > 0 iff ncomp = 1');
      if full <> 0 then write('         ');
      write('     ');
      write('          iff radius <= nodes/2 iff diam <=
nodes-1');
    end;
45: write('cycle iff maxdeg = mindeg = 2 and connected');
46: write('regular iff maxdeg = mindeg');
47: write('planar iff genus = 0 iff thick = 1');
48: write('if forest then planar, chr = 2, mindeg = 1');
49: if full = 1 then write('if cycle then ...more...')
    else
      begin
        writeln('if cycle then planar, regular, not forest,
cross# = 0,');
        if full <> 0 then write('         ');
```

```
        writeln('            nodes = edges = girth = circ > 2 =
nconn = bwidth = arb,');
        if full <> 0 then write('         ');
        writeln('            ncov = ecov = FL[(nodes+1)/2],');
        if full <> 0 then write('         ');
        writeln('            radius = nind = eind =
FL[nodes/2],');
        if full <> 0 then write('         ');
        writeln('            if nodes = 3 then clique = 3 and
nccov = 1,');
        if full <> 0 then write('         ');
        writeln('                         else clique = 2 and
nccov = ncov,');
        if full <> 0 then write('         ');
        write('            if nodes even then chr = echr = 2,
else chr = echr = 3');
      end;
50: if full = 1 then write('forest iff edges = nodes-ncomp
iff...more...')
      else
      begin
      writeln('forest iff edges = nodes-ncomp iff arb = 1 ');
      if full <> 0 then write('         ');
      write('   ');
      write('            iff girth and circ are undefined');
    end;
51: write('arb >= chr/2');
52: if full = 1 then write('arb <= ...more...')
      else
      begin
      writeln('arb <= 1+spectr/2,   always, and ');
      if full <> 0 then write('         ');
      writeln('           <= (1+maxdeg)/2, if (connct, not a
cycle, and not complete');
      if full <> 0 then write('         ');
      writeln('                                  with nodes odd)
or (clique <= maxdeg,');
      if full <> 0 then write('         ');
      write('                                    maxdeg >= 4 and
regular)');
    end;
53: if full = 1 then write('arb <= ...more...')
      else
      begin
      writeln('arb <= chr-FL[chr/(1+nodes/(s*chr))]');
      if full <> 0 then write('         ');
      write('    ');
      write('             where  s = FL[(girth-1)/2]');
    end;
54: write('if planar then mindeg <= 5, clique <= 4, arb <= 3,
cross# = 0');
55: write('edges >= (maxdeg+(nodes-1)*mindeg)/2');
```

```
56: write('edges <= ((nodes-1)*maxdeg+mindeg)/2');
57: write('if regular and mindeg is odd then nodes is even.');
58: write('clique >= nodes/(nodes-spectr)-1/3');
59:  if full = 1 then write('cross# <= ...more...')
      else
      begin
      writeln('cross# <= (FL[nodes/2]*Fl[(nodes-1)/2]');
      if full <> 0 then write('         ');
      write('
*Fl[(nodes-2)/2]*Fl[(nodes-3)/2])/4');
     end;
60: if full = 1 then write('if girth exists ...more...')
      else
      begin
      writeln('if girth exists and (nconn > 0 or mindeg > 1)
then');
      if full <> 0 then write('         ');
      write('                 genus >= edges*(1-2/girth)/2-
nodes/2+ncomp');
    end;
61: write('if genus <= 1 then eccov <= ncov*nind');
62: write('if mindeg >= nodes/2 then nconn >= nind');
63: if full = 1 then write('if connct then ...more...')
      else
      begin
      writeln('if connct then 2*diam-3-(diam**2-diam-4)/nodes');
      if full <> 0 then write('         ');
      write('                          <= (nodes**2-2
*edges)/nodes');
    end;
64: write('if nodes >= 3 and nconn >= nind then Hamiltonian');
65: write('if edges >= (nodes**2-3*nodes+6)/2 then Hamiltonian');
66: write('if planar and nconn >= 4 then Hamiltonian');
67: write('if connct and not (complete or odd cycle) then chr <=
maxdeg');
68: write('if complete then regular and echr = nodes  {-1, if
nodes even}');
69: write('chr >= 2*edges/(2*edges-spectr**2)');
70: if full = 1 then write('if genus <= ...more...')
      else
       begin
      writeln('if genus <= 1 then chr <= 7    if girth =  3');
      if full <> 0 then write('         ');
      write('                              <= 4    if girth >=
4');
    end;
71: if full = 1 then write('if girth exists then ...more...')
      else
       begin
      writeln('if girth exists then circ <= MIN[nodes-
(ncomp-1)*(mindeg+1),');
      if full <> 0 then write('         ');
```

```
      write('                edges-(ncomp-1)*mindeg], and
maxdeg >= 2');
    end;
72: write('Hamiltonian iff circ = nodes');
73: if full = 1 then write('if Hamiltonian then ...more...')
      else
       begin
      writeln('if Hamiltonian then arb >= 2, nconn >= 2, nind <=
nodes/2,');
      if full <> 0 then write('         ');
      write('   ');
      writeln('         ncov >= nodes/2, ecov <= (nodes+1)/2,');
      if full <> 0 then write('         ');
      write('            nccov <= (nodes+1)/2, eind >=
(nodes-1)/2');
     end;
74: write('bwidth <= nodes-FL[nind/2]-1       { retired }');
75: write('edges >= s*(nodes-nind*(s+1)/2)     where  s =
FL[nodes/nind]');
  end;
end;


procedure text150(i,full:longint);
begin
  if i <= 75 then text75(i,full)
  else
  case i of
76: write('eccov <= nccov+nodes*(nccov-1)/2');
77: if full = 1 then write('if nodes >= 6*mindeg and ...more...')
      else
       begin
      writeln('if nodes >= 6*mindeg and ');
      if full <> 0 then write('         ');
      writeln('           edges > (nodes-mindeg)*(nodes-
mindeg-1)/2+mindeg**2');
      if full <> 0 then write('         ');
      write('           then Hamilitonian');
    end;
78: write('if nodes >= 4 and edges >= 2*nodes-3 then girth <=
(circ+2)/2');
79: if full = 1 then write('if not a forest then ...more...')
      else
       begin
      writeln('if not a forest then nind >= (girth-1)/2,');
      if full <> 0 then write('         ');
      write('               radius >= (girth-1)/2 and eind >=
(circ-1)/2');
    end;
80: if full = 1 then write('if (girth exists and >=
4) ...more...')
      else
```

```
       begin
      writeln('if (girth exists and >= 4) or (forest and nodes >
2)');
      if full <> 0 then write('         ');
      write('           then not complete');
    end;
81: write('if regular then nind <= nodes/2-
(clique-1)*(clique-2)/(2*mindeg)');
82: write('if mindeg >= (nodes-1)/2 then econn = mindeg');
83: write('if genus > 0 then arb <= (9+SQRT[48*genus+1])/4');
84: write('if clique = 2 then arb <= 2+SQRT[genus]');
85: write('if clique = 2 then chr <= 3+2*SQRT[genus]');
86: if full = 1 then write('eind >= ...more...')
    else
     begin
       writeln('eind');
       if full <> 0 then write('         ');
       writeln('              >= MIN[FL[nodes/2], mindeg]');
       if full <> 0 then write('         ');
       writeln('                              if nodes-maxdeg
< mindeg <= maxdeg-2');
       if full <> 0 then write('         ');
       writeln('              >= nodes*mindeg/(mindeg+maxdeg)');
       if full <> 0 then write('         ');
       writeln('                              if mindeg <=
MIN[maxdeg-2, nodes-maxdeg]');
       if full <> 0 then write('         ');
       writeln('              >= nodes*maxdeg/(2*maxdeg+2)');
       if full <> 0 then write('         ');
       writeln('                              if maxdeg is even
and mindeg >= maxdeg-1');
       if full <> 0 then write('         ');
       writeln('              >= (nodes*mindeg+1)/(2*mindeg+2)');
       if full <> 0 then write('         ');
       writeln('                              if maxdeg is odd and
mindeg = maxdeg-1');
       if full <> 0 then write('         ');
       writeln('              >= nodes/2');
       if full <> 0 then write('         ');
       writeln('                              if complete and
nodes even');
       if full <> 0 then write('         ');
       writeln('              >= ((nodes-u*(mindeg-1))/2-k');
       if full <> 0 then write('         ');
       writeln('                              if maxdeg = mindeg
is odd and < nodes-1');
       if full <> 0 then write('         ');
       writeln('                 where u = FL[(nodes-
mindeg-3)/(mindeg+1)**2] and');
       if full <> 0 then write('         ');
       write('                   k = FL[(FL[(nodes-u*(mindeg+1)**
2)/(mindeg+2)]-1)/2]');
```

```
     end;
87: write('if genus > 0 then mindeg <= (5+SQRT[48*genus+1])/2');
88: write('if genus > 0 and clique = 2 then econn <= 2+2
*SQRT[genus]');
89: if full = 1 then write('if genus = 0 then ...more...')
      else
       begin
      writeln('if genus = 0 then econn <= 5, 3, 3, 2 ');
      if full <> 0 then write('         ');
      write('                       when girth = 3, 4, 5, >= 6,
respectively');
    end;
90: if full = 1 then write('if genus <= 1 then ...more...')
      else
       begin
      writeln('if genus <= 1 then econn <= 6, 4, 3, 3, 2 .');
      if full <> 0 then write('         ');
      write('    ');
      write('                   when girth = 3, 4, 5, 6, >= 7,
respectively');
    end;
91: if full = 1 then write('if mindeg >= 3 ...more...')
      else
       begin
      writeln('if mindeg >= 3 and 1 <= s = FL[(girth-1)/4] ');
      if full <> 0 then write('         ');
      write('           then nodes >= girth*(mindeg-1)**s');
    end;
92: write('if nconn >= 2 then circ >= MIN[nodes, 2*mindeg]');
93: if full = 1 then write('if...more...then edges >=
(nodes-1)*(mindeg+1)/2')
   else
     begin
      writeln('if diam = 2 and ((nodes > mindeg**3+mindeg+1)
or');
      if full <> 0 then write('         ');
      writeln('                  (nodes or mindeg odd and nodes >
mindeg**3+1))');
      if full <> 0 then write('         ');
      write('               then edges >= (nodes-1)*(mindeg+
1)/2');
     end;
94: if full = 1 then write('if...more...then edges >=
(nodes-1)*(nconn+1)/2')
   else
     begin
       writeln('if diam = 2 and ((nodes > nconn**3+nconn+1) or');
       if full <> 0 then write('          ');
       writeln('                  (nodes or nconn odd and nodes >
nconn**3+1))');
       if full <> 0 then write('          ');
       write('               then edges >= (nodes-1)*(nconn+
```

```
1)/2');
     end;
95: if full = 1 then write('if...more...then edges >=
(nodes-1)*(econn+1)/2')
   else
     begin
       writeln('if diam = 2 and ((nodes > econn**3+econn+1) or');
       if full <> 0 then write('          ');
       writeln('                 (nodes or econn odd and nodes >
econn**3+1))');
       if full <> 0 then write('          ');
       write('               then edges >= (nodes-1)*(econn+1)/2
{ retired }');
     end;
96: write('if girth exists then nodes >= (girth-1)*(arb-1)+1');
97: write('if clique = 2 and chr >= 4 then nodes >= 11');
98: if full = 1 then write('if girth > 3, genus > 1,
and ...more...')
   else
    begin
      writeln('if girth > 3, genus > 1, and chr >= (3
*girth-2)/(girth-2) then');
      if full <> 0 then write('        ');
      writeln('             chr <= (3+6*x+SQRT[57-60*x+36*x**2+48
*x*genus])/6');
      if full <> 0 then write('        ');
      write('                        where x = girth/(girth-2)');
    end;
99: write('if diam <= 2 then econn = mindeg');
100: write('if nind >= eind then eccov <= ncov*nind');
101: if full = 1 then write('ncov*ecov <= ...more...')
      else
       begin
       writeln('ncov*ecov <= (nodes-1)*FL[(nodes+1)/2],');
       if full <> 0 then write('         ');
       writeln('             if connct or nodes odd, otherwise');
       if full <> 0 then write('         ');
       write('                      <= (nodes**2-4)/2');
     end;
102: write('echr <= 2*bwidth');
103: write('circ >= clique*mindeg/(clique-1)');
104: write('circ >= clique*(chr-1)/(clique-1)');
105: write('if clique = 2 and chr >= 3 then circ >= 2*chr-1');
106: write('edges <= ncov*(nind+ncov*(chr-1)/(2*chr))');
107: write('mindeg <= maxdeg');
108: write('nccov <= (nodes+nind-clique+1)/2');
109: write('eind >= ncov/2');
110: if full = 1 then write('if mindeg >= 4 ...more...')
      else
       begin
       writeln('if mindeg >= 4 and girth >= 5 ');
       if full <> 0 then write('         ');
```

```
       write('               then circ >= (girth-2)*(mindeg-2)+
5');
     end;
111: write('if connct then diam <= 2*nind-1');
112: write('if connct, mindeg >= MAX[nind, (nodes+2)/3] then
Hamiltonian');
113: write('edges >= nind*mindeg+(clique-1)*(clique-2)/2');
114: write('edges >= ncov+(clique-1)*(clique-2)/2');
115: write('edges >= chr*(chr-3)/2+nodes-ncomp+1');
116: if full = 1 then write('if bipartite then ...more...')
      else
       begin
       writeln('if bipartite then genus <= CL[(nodes-4)**2/16],
if nodes even,');
       if full <> 0 then write('         ');
       write('                  ');
       write('  else genus <= CL[(nodes-3)*(nodes-5)/16]');
     end;
117: write('if not complete then nconn >= 2*mindeg-nodes+2');
118: if full = 1 then write('if nodes = ...more...')
      else
       begin
       writeln('if nodes = 2*k or 2*k+1 >= 6, and edges >=');
       if full <> 0 then write('         ');
       write('             ');
       write('    k**2+1 {+mindeg, when nodes = 2*k+1}, then
circ >= 5');
     end;
119: write('if chr > clique then mindeg <= (3*clique-4)*nodes/(3
*clique-1)');
120: if full = 1 then write('if Hamiltonian then ...more...')
      else
       begin
       writeln('if Hamiltonian then edges > nodes, when nodes
even and chr = 3,');
       if full <> 0 then write('         ');
       write('                  ');
       write('and >= nodes+(chr-1)*(chr-2)/2, when 4 <= chr
< nodes');
     end;
121: write('chr <= nodes-nconn*(diam-3)-2');
122: if full = 1 then write('if edges > nodes**2/4
then ...more...')
      else
       begin
       writeln('if edges > nodes**2/4 then eccov <= k+(1+SQRT[1+4
*k])/2');
       if full <> 0 then write('         ');
       write('               where k = nodes*(nodes-1)/2-edges');
     end;
123: write('if nodes <= 2*econn+3 then mindeg = econn');
124: write('if connct then bwidth >= (nodes-1)/diam');
```

```
125: if full =1 then write('if genus > 0 then
nodes >= ...more...')
      else
      begin
        writeln('if genus > 0 then nodes >= FL[9*(girth-1)/4]');
        if full <> 0 then write('         ');
        write('                                    {+1, if girth
= 3 MOD 4}');
      end;
126: write('if mindeg >= (nodes-1)/2 then nodes <= 2*econn+3');
127: write('if Hamiltonian, nodes even, and maxdeg = 3 then echr
= maxdeg');
128: if full = 1 then write('if nodes >= 2*eind+1 then edges
<= ...more...')
     else
      begin
        writeln('if nodes >= 2*eind+1 then edges <=');
        if full <> 0 then write('         ');
        writeln('         (i) MIN[nodes*maxdeg/2, eind*maxdeg');
        if full <> 0 then write('         ');
        writeln('             +FL[(2*nodes-2*eind)/(maxdeg+
3)]*(maxdeg-1)/2]');
        if full <> 0 then write('         ');
        writeln('                when maxdeg odd and maxdeg <= 2
*eind,');
        if full <> 0 then write('         ');
        writeln('                  and nodes<= 2
*eind+eind/FL[(maxdeg+1)/2]');
        if full <> 0 then write('         ');
        writeln('        (ii) nodes*maxdeg/2');
        if full <> 0 then write('         ');
        writeln('                when maxdeg even and maxdeg <= 2
*eind,');
        if full <> 0 then write('         ');
        writeln('                 and nodes <= 2
*eind+eind/FL[(maxdeg+1)/2]');
        if full <> 0 then write('         ');
        writeln('       (iii) eind*maxdeg+FL[eind/FL[(maxdeg+
1)/2]]*FL[maxdeg/2]');
        if full <> 0 then write('         ');
        writeln('               when maxdeg <= 2*eind < nodes-
FL[eind/FL(maxdeg+1)/2]]');
        if full <> 0 then write('         ');
        writeln('        (iv) eind*maxdeg');
        if full <> 0 then write('         ');
        writeln('                when 2*eind+1 <= maxdeg <=
nodes-eind');
        if full <> 0 then write('         ');
        writeln('         (v) MAX[eind*(2*eind+1),
eind*(nodes+maxdeg-eind)/2]');
        if full <> 0 then write('         ');
        write('               when nodes-maxdeg < eind <=
```

```
(maxdeg-1)/2');
      end;
129: if full = 1 then write('if 3 < diam < infinity
then ...more...')
      else
       begin
       writeln('if 3 < diam < infinity then ');
       if full <> 0 then write('         ');
       write('                  mindeg <= (nodes-
diam-3)/(FL[diam/3]+1)+2');
     end;
130: if full = 1 then write('if diam = 2 then ...more...')
      else
       begin
       writeln('if diam = 2 then edges >= nodes*(nodes-1)/(2
*maxdeg),');
       if full <> 0 then write('         ');
       writeln('                                when maxdeg**2 <=
8*nodes,');
       if full <> 0 then write('         ');
       write('               ');
       write('else edges >= maxdeg*nodes*(nodes-1)/(maxdeg**2+8
*nodes)');
     end;
131: write('chr <= maxdeg+1-FL[(maxdeg+1)/MAX[4, clique+1]]');
132: if full = 1 then write('mindeg <= ...more...')
      else
       begin
       writeln('mindeg <= (nodes-1)**2/(4*(nodes-1-maxdeg)),');
       if full <> 0 then write('         ');
       writeln('                                when nodes <> 3
MOD 4,');
       if full <> 0 then write('         ');
       write('               ');
       write('else mindeg <= (nodes-3)*(nodes+1)/(4*(nodes-1-
maxdeg))');
     end;
133: if full = 1 then write('nind >= ...more...')
      else
       begin
       writeln('nind >= CL[nodes-2*edges/x]+');
       if full <> 0 then write('         ');
       writeln('                   CL[(nodes-CL[nodes-2
*edges/x]*x)/(x+1)]');
       if full <> 0 then write('         ');
       write('                     where x = 1+FL[2
*edges/nodes]');
      end;
134: if full = 1 then write('if radius = diam = 2
then ...more...')
      else
       begin
```

```
       writeln('if radius = diam = 2 then edges >= 4,  if nodes =
4,');
       if full <> 0 then write('         ');
       write('                            else edges >= 2
*nodes-5');
     end;
135: write('edges >= 2*ncov-ncomp');
136: if full = 1 then write('if maxdeg <= 2*eind and is odd
then ...more...')
      else
       begin
       writeln('if maxdeg <= 2*eind and maxdeg is odd then ');
       if full <> 0 then write('         ');
       write('          ');
       write('  edges <= eind*maxdeg+((maxdeg-1)/2)*FL[2
*eind/(maxdeg+1)]');
     end;
137: if full = 1 then write('edges >= ...more...  { retired }')
     else
       begin
         writeln('edges >=nodes*(nodes-1)*(maxdeg-2)/');
         if full <> 0 then write('         ');
         write('                   (2*((maxdeg-1)**diam-1))
{ retired }');
       end;
138: write('if edges >= (nodes**2-5*nodes+14)/2 then circ >=
nodes-1');
139: write('if edges >= (circ*(2*nodes-circ)+1)/4 then girth =
3');
140: write('if edges >=4*nodes then cross# > edges**3/(100
*nodes**2)');
141: write('circ >= 2*edges/(nodes-1)');
142: write('if edges > 1+nodes+FL[nodes**(3/2)/2] then girth < 5
{ retired }');
143: write('nccov <= .5+SQRT[nodes**2-nodes-2*edges+.25]');
144: write('chr <= .5+SQRT[2*edges+.25]');
145: write('if clique = 2 then nind >= (SQRT[8*nodes+9]-3)/2');
146: write('if connct then bwidth <= nodes-diam');
147: if full = 1 then write('if clique = 2 then ...more...')
      else
       begin
       writeln('if clique = 2 then chr <= 2, if nodes <= 4 or
maxdeg >= nodes-2');
       if full <> 0 then write('         ');
       writeln('                              <= 3, if 4 < nodes
< 11');
       if full <> 0 then write('         ');
       write('      ');
       write('                         <= (nodes-maxdeg+10)/4,
otherwise');
     end;
148: write('if cubic, planar and econn > 1 then echr = maxdeg');
```

```
149: write('if planar and maxdeg >= 8 then echr = maxdeg');
150: write('if spectr <= maxdeg/2 then echr = maxdeg');
end;
end;

procedure text225(i,full:longint);
begin
  case i of
 151: write('if nodes > 2, regular, and nconn = 1 then echr =
maxdeg+1');
152: if full = 1 then write('if clique=2 and ...more...')
      else
       begin
       writeln('if clique = 2 and 2*nodes/5 <= nind < nodes/2
then');
       if full <> 0 then write('         ');
       write('                   edges <= nind**2+4*(nodes/2-
nind)**2');
     end;
153: write('bwidth <= maxdeg*(maxdeg-1)**(radius-1)');
154: if full = 1 then write('if maxdeg >= 3 then ...more...
{ retired }')
      else
       begin
       writeln('if maxdeg >= 3 then              { retired }');
       if full <> 0 then write('         ');
       write('           nodes <= 1
+maxdeg*((maxdeg-1)**diam-1)/(maxdeg-2)');
     end;
155: write('if mindeg = 2 then nodes <= (2+MAX[4, maxdeg])
eind/2');
156: write('if mindeg <= nodes/2 then eind >= mindeg');
157: write('if clique <= (maxdeg-1)/2 then chr <= maxdeg-1');
158: if full = 1 then write('if connct then ...more...')
      else
       begin
       writeln('if connct then genus <= 0, 1, 2  when,
respectively,');
       if full <> 0 then write('         ');
       write('              ');
       write('edges-nodes+1 is <= 3, in {4,5,6,7}, in {8,9,10}');
     end;
159: write('nind >= (nodes-1)/(maxdeg+1)+1/(mindeg+1)');
160: if full = 1 then write('if clique = 2 then ...more...')
      else
       begin
       writeln('if clique = 2 then nind >= nodes/(maxdeg-.2),
when maxdeg > 2,');
       if full <> 0 then write('         ');
       writeln('            and nind >= nodes/maxdeg-1/(maxdeg+
1)+1/(mindeg+1),');
       if full <> 0 then write('         ');
```

```
       write('               ');
       write('when nodes >= 3, connct, and not (odd cycle or even
path)');
      end;
161: if full = 1 then write('if genus > 1 ...more...')
      else
       begin
       writeln('if genus > 1, girth > 3, and mindeg >=');
       if full <> 0 then write('         ');
       writeln('           FL[(5+SQRT[16*genus+1])/2], then
regular, Hamiltonian and');
       if full <> 0 then write('         ');
       write('           ');
       write('nodes = 2*mindeg {+2, when mindeg = (3+SQRT[16
*genus+9])/2}');
     end;
162: if full = 1 then write('if nconn >= 2 ...more...')
      else
       begin
       writeln('if nconn >= 2, regular, and (mindeg >= (nodes-
SQRT[nodes])/2');
       if full <> 0 then write('         ');
       write('              ');
       writeln(' or (nodes even and mindeg >= (nodes-SQRT[2
*nodes])/2))');
       if full <> 0 then write('         ');
       write('                   then Hamiltonian');
     end;
163: write('if regular, nconn >= 2 and nodes <= 3*mindeg then
Hamiltonian');
164: write('if spectr > SQRT[edges] then girth = 3');
165: write('spectr >= SQRT[maxdeg]');
166: if full = 1 then write('if diam > 1 ...more...')
      else
       begin
       writeln('if diam > 1 and nconn > 1 then ');
       if full <> 0 then write('         ');
       write('             edges >= (nodes*diam-2
*diam-1)/(diam-1)');
     end;
167: write('if girth >= 5 then chr <= 2*(maxdeg+3)/3');
168: write('if girth >= 2*maxdeg**2 then chr <= (maxdeg+4)/2');
169: write('nind >= nodes**2/(2*edges+nodes)');
170: if full = 1 then write('if connct and not complete
then ...more...')
      else
       begin
       writeln('if connct and not complete then');
       if full <> 0 then write('         ');
       write('   ');
       write('          nind >= (nodes**3+3*nodes+1)/(nodes*(2
*edges+nodes))');
```

```
     end;
171: write('if genus > 1 and girth >= 4 then mindeg <= 2+2
*SQRT[genus]');
172: write('if connct then diam <= 2*ncov');
173: write('nind >= 2*nodes/(maxdeg+clique+1)');
174: write('nind >= (nodes+2*maxdeg+1-clique-mindeg)/(maxdeg+
1)');
175: write('bwidth >= (2*nodes-1-SQRT[(2*nodes-1)**2-8
*edges])/2');
176: write('if clique = 2 then bwidth >= (3*mindeg-2)/2');
177: write('if tree then bwidth <= nodes/2
{ retired }');
178: write('nccov <= nodes-mindeg-FL[(nodes-mindeg)/MAX[4, nind+
1]]');
179: write('if dom >= 2 then edges <= (nodes-nind)*(nodes+nind-2
*dom+2)/2');
180: if full = 1 then write('if regular and ...more...')
      else
       begin
       writeln('if regular and 2 < (nodes-1)/2 <= maxdeg <=
nodes-2 then');
       if full <> 0 then write('         ');
       write('            ');
       writeln('chr <= MIN[maxdeg, 3*nodes/5], if maxdeg+nodes is
odd, else');
       if full <> 0 then write('         ');
       write('                ');
       writeln('<= MIN[maxdeg,');
       if full <> 0 then write('         ');
       write('                       (2*(nodes-
maxdeg)-3)*nodes/(3*(nodes-maxdeg)-4)]');
     end;
181: write('if nccov > nind then maxdeg >= 3*nodes/(3
*nind-1)-1');
182: write('clique >= 2*nodes/(nodes-mindeg+nind)');
183: write('if nind <= 2 then clique >= (SQRT[8*nodes+9]-3)/2');
184: write('edges >= nodes*(nodes-1)/2-(nodes-clique)*(nodes-
mindeg-1)');
185: write('if nccov <= 2 then clique = chr');
186: write('if nind = 2 and nccov >= 4 then nodes >= 11');
187: if full = 1 then write('if regular and maxdeg <= nodes-2
then ...more...')
      else
       begin
       writeln('if regular and maxdeg <= nodes-2 then ');
       if full <> 0 then write('         ');
       write('          ');
       write('   clique <= nodes/2-(nind-1)*(nind-2)/(2*(nodes-
maxdeg-1))');
     end;
188: if full = 1 then write('if girth is undefined
then ...more...')
```

```
      else
       begin
       writeln('if girth is undefined then thick = 1,');
       if full <> 0 then write('         ');
       write('               ');
       write('              else thick >=
edges*(1-2/girth)/(nodes-2)');
     end;
189: write('thick <= (nodes+7)/6, if nodes <> 9 or 10. Then <=
3.');
190: write('thick <= (echr+1)/2');
191: write('thick <= MAX[bwidth/2, 1]');
192: write('if clique = 2 then nind >= mindeg*CL[(diam+1)/4]');
193: write('thick <= (ncov+1)/2');
194: write('thick >= 3, if clique = 9 or 10, else >= (clique+
2)/6');
195: write('edges <= nodes*(nodes-1)/2-nodes+clique-
(nind-1)*(nind-2)/2');
196: if full = 1 then write('edges <= ...more...')
      else
        begin
          writeln('edges <= nodes*(nodes-1)/2-clique*(nodes-
maxdeg-1)-');
          if full <> 0 then write('         ');
          write('
(nind-1)*(nind-2)/2');
        end;
197: write('if nodes >= 3 then eccov <= thick*(2*nodes-
ncomp-3)');
198: write('ncov <= nodes-nodes/chr');
199: write('bwidth <= nodes-1-FL[(nodes-ncov)/2]');
200: write('if nodes > 2*eind+1 then ncov <= 2*eind-nconn');
201: if full = 1 then write('if nodes >= MAX[6, 3*eind-1]
and ...more...')
      else
       begin
       writeln('if nodes >= 6, connct and nodes > 3*eind-2
then');
       if full <> 0 then write('         ');
       write('             ncov <= 2*eind-mindeg');
     end;
202: write('if regular then 2*ncov >=
nodes+(clique-1)*(clique-2)/mindeg');
203: write('if clique = 2 then ncov <= (2*nodes+3-SQRT[8*nodes+
9])/2');
204: if full = 1 then write('if clique = 2 and ...more...')
      else
       begin
       writeln('if clique = 2 and nodes/2 < ncov <= 3*nodes/5
then');
       if full <> 0 then write('         ');
       write('            ncov <= (3*nodes-SQRT[5*edges-nodes**
```

```
2])/5');
     end;
205: if full = 1 then write('if mindeg = 2 then ecov
<= ...more...')
      else
        begin
          writeln('if mindeg = 2 then');
          if full <> 0 then write('         ');
          write('             ecov <= nodes*MAX[4, maxdeg]/CL[2
+MAX[4, maxdeg]]');
        end;
206: write('ncov <= (nodes*maxdeg+1)/(maxdeg+1)-1/(mindeg+1)');
207: if full = 1 then write('if clique = 2 then ncov
<= ...more...')
     else
      begin
        writeln('if clique = 2 then ncov <=
nodes*(maxdeg-1.2)/(maxdeg-.2)');
        if full <> 0 then write('         ');
        writeln('                           when maxdeg >= 3,
and');
        if full <> 0 then write('         ');
        writeln('                  <=
nodes*(maxdeg-1)/maxdeg-1/(maxdeg+1)-1/(mindeg+1)');
        if full <> 0 then write('         ');
        writeln('                           when nodes >= 3,
connct and ');
        if full <> 0 then write('         ');
        write('                           not (odd cycle or even
path)');
      end;
208: if full = 1 then write('if connct and not complete
then ...more...')
      else
       begin
       writeln('if connct and not complete then ncov <=');
       if full <> 0 then write('         ');
       write('        ');
       write('       (2*edges*nodes**2-3*nodes-1)/(2
*edges*nodes+nodes**2)');
     end;
209: write('ncov <= nodes(1-2/(maxdeg+clique+1))');
210: write('ncov <= ((nodes-2)maxdeg+clique+mindeg-1)/(maxdeg+
1)');
211: if full = 1 then write('if ncov > nodes-nccov
then ...more...')
       else
         begin
           writeln('if ncov > nodes-nccov then');
           if full <> 0 then write('         ');
           write('               ncov <= nodes*maxdeg/(maxdeg+
1)-1/3');
```

```
        end;
212: write('nccov <= ecov');
213: write('dom <= eind');
214: write('dom <= nind');
215: write('ncomp <= dom');
216: write('maxdeg <= echr');
217: write('echr <= maxdeg+1');
218: write('mindeg <= ncov');
219: write('econn <= mindeg');
220: write('clique <= chr');
221: write('chr <= ncov+1');
222: write('eind <= ncov');
223: write('nind <= nccov');
224: write('nccov <= eccov');
225: write('radius <= diam');
  end;
end;


procedure text300(i,full:longint);
begin
  if i < 226 then text225(i,full)
    else
  case i of
226: write('nconn <= econn');
227: write('girth <= circ');
228: write('chr <= circ');
229: write('genus <= cross#');
230: write('mindeg <= circ-1');
231: write('chr <= bwidth+1');
232: write('mindeg <= bwidth');
233: write('nodes <= nind*chr');
234: write('nodes <= nccov*clique');
235: write('edges <= echr*eind');
236: write('edges <= ncov*maxdeg');
237: write('ncov <= bwidth*nind');
238: write('chr <= spectr+1');
239: write('nodes = ncov+nind');
240: write('nodes = ecov+eind');
241: write('if  clique=2 then chr <= 3*(nodes+12)/16');
242: write('nodes >= nccov+chr-1');
243: write('nodes >= dom+maxdeg');
244: write('if nind = 2 then nccov <= 3*(nodes+12)/16');
245: write('if mindeg >= 2 then eccov <= 2*(nodes-2+2*genus)-4
*(ncomp-1)');
246: write('if nodes >= 3 then eccov <= 2*(nodes-2+2*genus)-
ncomp+1');
247: write('nind <= nodes/(1+mindeg/maxdeg)');
248: write('ncov >= nodes/(1+maxdeg/mindeg)');
249: write('nind >= nodes/(bwidth+1)');
250: write('ncov <= nodes/(1+1/bwidth)');
251: write('if girth is defined then edges >= (girth-1)*(arb-1)**
```

```
2+arb-1');
252: if full = 1 then write('if nconn >= 2 and girth >= 4
then ...more...')
      else
       begin
       writeln('if nconn >= 2 and girth >= 4 then');
       if full <> 0 then write('         ');
       write('           ');
       writeln('eind >= maxdeg*CL[(girth-4)/4] {+1 or 2, when,
respectively,');
       writeln('            (girth = 2 MOD 4 or (maxdeg=2 and
girth =3 MOD 4)) or');
       write('            (girth = 0 MOD 4 or (maxdeg >= 3 and
girth = 3 MOD 4))}');
     end;
253: if full = 1 then write('if nconn >= 1 then maxdeg
<= ...more...')
       else
         begin
           writeln('if nconn >= 1 then maxdeg <=');
           if full <> 0 then write('         ');
           write('
(nodes-1)/(CL[(girth-4)/4]*(nconn-1)+1)');
         end;
254: if full = 1 then write('if nind = 2 then ...more...')
      else
       begin
       writeln('if nind = 2 then nccov <= 2,      if mindeg = 1
or nodes <= 4');
       if full <> 0 then write('         ');
       writeln('                                  <= 3,      if 5
<= nodes <= 10');
       if full <> 0 then write('         ');
       write('                                 <= (mindeg+11)/4,
otherwise');
     end;
255: write('if connct then edges >= nodes+8*thick-13');
256: if full = 1 then write('if maxdeg >= 3 then ...more...')
      else
       begin
       writeln('if maxdeg >= 3 then diam >=');
       if full <> 0 then write('         ');
       write('               MIN[k: (maxdeg-1)**k >=
(nodes*(maxdeg-2)+2)/maxdeg]');
     end;
257: if full = 1 then write('if diam >= 3 then ...more...')
      else
       begin
       writeln('if diam >= 3 then eind >= ');
       if full <> 0 then write('         ');
       write('            nconn*CL[(diam-2)/2]+1 {+1, when diam
even and nconn >= 2}');
```

```
     end;
258: if full = 1 then write('genus >= thick+(edges-4*nodes-1)/6,
when ...more...')
     else
       begin
         writeln('genus >= thick+(edges-4*nodes-1)/6, when
((nconn > 0 and');
         if full <> 0 then write('         ');
         write('           nodes > 2) or mindeg > 1) and (thick
<> 3 or nodes <> 9, 10)');
       end;
259: write('if girth > 1+2*LOG2[nodes] then chr <= 3');
260: write('genus <= k/2-k/(4*LOG2[k]),  where  k=edges-
nodes+ncomp');
261: write('if genus <= 2 and girth >= 5 then chr <= 4');
262: if full = 1 then write('if genus = 0 and girth >=
4 ...more...')
      else
       begin
       writeln('if (genus = 0, girth >= 4) or (genus <= 1,
girth >= 6)');
       if full <> 0 then write('         ');
       write('           ');
       write('  or (genus <= 2, girth >= 7) or (girth >= 9) then
chr <= 3');
     end;
263: write('maxdeg >= (nodes-1)**(1/radius)');
264: write('if connct then diam <= 3*dom-1');
265: if full = 1 then write('if connected and maxdeg >= 3
then ...more...')
      else
       begin
       writeln('if connected and maxdeg >= 3 then nodes <=');
       if full <> 0 then write('         ');
       write('              1
+mindeg*((maxdeg-1)**diam-1)/(maxdeg-2)');
     end;
266: if full = 1 then write('if diam >= 2 and maxdeg >= 3
then ...more...')
      else
       begin
       writeln('if diam >= 2 and maxdeg >= 3 then');
       if full <> 0 then write('         ');
       write('             maxdeg >=
CL[(nodes-1)/mindeg]**(1/(diam-1))');
     end;
267: write('if diam = radius = 2 then mindeg >= 2');
268: write('if not forest then bwidth >= (girth-1)*(arb-2)+2');
269: write('if not forest then bwidth >= (girth-1)*nodes/(2
*nind)-girth+2');
270: write('if not a forest then dom <= nodes-Fl(2*circ/3)');
271: write('circ >= FL[nodes/(k-1)],     where
```

```
k=CL[nodes/mindeg]');
272: write('if diam <= 2 then dom <= nconn');
273: write('if tree then radius = CL[diam/2]');
274: write('if Hamiltonian then nodes >= (maxdeg-1)*(girth-2)+
2');
275: write('nind >= maxdeg/(chr-1)');
276: write('clique >= (nodes-mindeg-1)/(nccov-1)');
277: write('if nconn <= 1 and nodes >= 3 then econn <=
maxdeg/2');
278: if full = 1 then write('if connct and maxdeg >= 3
then ...more...')
      else
       begin
       writeln('if connct and maxdeg >= 3 then');
       if full <> 0 then write('         ');
       write('            nodes <= 1
+maxdeg*((maxdeg-1)**radius-1)/(maxdeg-2)');
     end;
279: write('if connct then nodes >= maxdeg+2*radius-2');
280: write('if forest then bwidth <= nodes/2-ncomp+1');
281: if full = 1 then write('if nconn >= 2 ...more...')
      else
       begin
       writeln('if nconn >= 2, nodes <= 3*mindeg and ');
       if full <> 0 then write('         ');
       write('           edges <= ((nodes+1)*mindeg-1)/2 then
Hamiltonian');
     end;
282: write('if nodes even and not bipartite then not a cycle');
283: write('if cubic and nconn = 3 then circ >= nodes**(2/3)+1');
284: if full = 1 then write('if ..... then nind >= ...more...')
      else
       begin
       writeln('if connct, clique = 2, and not (odd cycle or even
path) then ');
       if full <> 0 then write('         ');
       write('             ');
       write('nind >= nodes/(maxdeg*(maxdeg+1))+(nodes**
2)/(nodes+2*edges)');
     end;
285: write('eccov <= nccov+nodes*(maxdeg+1-nodes/nccov)/2');
286: write('edges >= (maxdeg+(chr-1)**2+(nodes-chr)*mindeg)/2');
287: write('if planar then mindeg <= nind+2');
288: if full = 1 then write('if diam = clique = 2 ...more...')
      else
       begin
       writeln('if diam = clique = 2, regular, not bipartite,
and');
       if full <> 0 then write('         ');
       write('            maxdeg = 2*k or 3*k, then nodes >= 3
*maxdeg-k');
     end;
```

```
289: if full = 1 then write('if connct, not
Hamiltonian ...more...')
      else
       begin
       writeln('if connct, not Hamiltonian, not a tree and
circ >= (nodes-2)/2');
       if full <> 0 then write('         ');
       write('            then dom <= (2*nodes-circ)/3');
     end;
290: write('if connct, dom >= 3 then edges <= (nodes-dom+
1)*(nodes-dom)/2');
291: write('dom <= nodes*(1+LN[2]*LOG2[mindeg])/(mindeg+1)');
292: if full = 1 then write('if connected and not a tree then
girth <= 2*diam+1 and...more...')
    else
     begin
      writeln('if connected and not a tree then girth <= 2*diam+1
and equality');
      if full <> 0 then write('         ');
      writeln('             holds iff complete with nodes >= 3, a
cycle,');
      if full <> 0 then write('         ');
      write('             or a degree 3, 7, or 57 Moore graph');
     end;
293: if full = 1 then write('if connected and maxdeg = 2
then ...more...')
else
     begin
      writeln('if connected and maxdeg = 2 then ');
       if full <> 0 then write('         ');
       writeln('              nodes=diam+1,                if
mindeg=1');
       if full <> 0 then write('         ');
       write('              2*diam <= nodes <= 2*diam +1, if
mindeg=2');
      end;
294: write('eccov <= ((nodes-clique+2)**2)/4');
295: write('edges <= MAX[eind*(2*eind+1), eind*(2*nodes-
eind-1)/2]');
296: write('if cubic then nconn = econn');
297: write('if nconn > 2, genus = 0, and not Hamiltonian then
nodes >= 11');
298: write('if cubic, nconn > 1, not planar, not bipartite then
nodes >= 8');
299: write('if cubic and nconn = 1 then nodes >= 10');
300: write('if cubic, nconn = 1, and not planar then nodes >=
12');
 end;
end;

end.
```