```
unit rules150;

     interface
          uses
           globals,cmmnds1,pusherr,pushStack,ruleAtoF;

         procedure r101; procedure r102; procedure r103;
procedure r104; procedure r105;
           procedure r106; procedure r107; procedure r108;
procedure r109; procedure r110;
           procedure r111; procedure r112; procedure r113;
procedure r114; procedure r115;
           procedure r116; procedure r117; procedure r118;
procedure r119; procedure r120;
           procedure r121; procedure r122; procedure r123;
procedure r124; procedure r125;
           procedure r126; procedure r127; procedure r128;
procedure r129; procedure r130;
           procedure r131; procedure r132; procedure r133;
procedure r134; procedure r135;
           procedure r136; procedure r138;
procedure r139; procedure r140;
           procedure r141; procedure r143; procedure r144;
procedure r145; procedure r146;
           procedure r147; procedure r148; procedure r149;
procedure r150;

implementation



procedure r101;
(*************************************************)
(*                                               *)
(* if not connected and P even then              *)
(*          ncov*ecov <= (P-2)*(P+2)/2           *)
(*    else  ncov*ecov <= (P-1)*trunc((P+1)/2)    *)
(*                                               *)
(*************************************************)
begin
  if activerule[101] then
     begin
       rule:='101/ ';
       if (max[connct] = 0) and ((pReven = eq) or
          (not((odd(max[nodes])))))) and (max[nodes] < infinity)
then
          begin
            z1:=(max[nodes]*max[nodes]-4) div 2;
            z:=z1 div min[ecov];
            if z < max[ncov] then pushmax(ncov);
            z:=z1 div min[ncov];
```

```
            if z < max[ecov] then pushmax(ecov);
            z:=round(sqrt(2*min[ncov]*min[ecov]+4)+hf);
            if z > min[nodes] then pushmin(nodes);
          end
         else
           begin
             if max[nodes] < infinity then
               begin
                 z1:=(max[nodes]-1)*((max[nodes]+1) div 2);
                 z:= z1 div min[ecov];
                 if z < max[ncov] then pushmax(ncov);
                 z:=z1 div min[ncov];
                 if z < max[ecov] then pushmax(ecov);
               end;
             z:=round(sqrt(2*min[ncov]*min[ecov]+1)+hf);
             if z > min[nodes] then pushmin(nodes);
           end;
     end;
end;

procedure r102;
(*********************************)
(*                               *)
(*        echr <= 2*bandwith     *)
(*                               *)
(*********************************)
begin
  if activerule[102] then
    begin
      rule:='102/ ';
      if max[bwidth] < infinity then
           begin
             z:=2*max[bwidth];
             if z < max[echr] then pushmax(echr);
           end;
      z:=(min[echr]+1) div 2;
      if z > min[bwidth] then pushmin(bwidth);
    end;
end;

procedure r103;
(*************************************)
(*                                   *)
(*  circ >= clique*mindeg/(clique-1)    *)
(*                                   *)
(*************************************)
begin
  if activerule[103] then
    begin
      rule:='103/ ';
      z:=max[clique];
      if z < infinity then
```

```
              begin
                z:=(z*min[mindeg]-1) div (z-1)+1;
                if z > min[circ] then pushmin(circ);
              end;
      z:=max[clique];
      if (z < infinity) and (max[circ] < infinity) then
           begin
             z:=(max[circ]*(z-1)) div z;
             if z < max[mindeg] then pushmax(mindeg);
           end;
      z:=max[circ];
      if (z < infinity) and (z > min[mindeg]) then
           begin
             z:=(z-1) div (z-min[mindeg])+1;
             if z > min[clique] then pushmin(clique);
           end;
    end;
end;

procedure r104;
(*****************************************)
(*                                       *)
(*   circ >= clique*(chr-1)/(clique-1)   *)
(*                                       *)
(*****************************************)
begin
  if activerule[104] then
    begin
      rule:='104/ ';
      z:=max[clique];
      if z < infinity then
             begin
               z:=(z*(min[chr]-1)-1) div (z-1)+1;
               if z > min[circ] then pushmin(circ);
             end;
      z:=max[clique];
      if (z < infinity) and (max[circ] < infinity) then
           begin
             z:=(max[circ]*(z-1)) div z + 1;
             if z < max[chr] then pushmax(chr);
           end;
      z:=max[circ];
      if (z < infinity) and (z > min[chr]-1) then
           begin
             z:=(z-1) div (z-min[chr]+1)+1;
             if z > min[clique] then pushmin(clique);
           end;
   end;
end;

procedure r105;
(*********************************)
```

```
(*                                    *)
(*  if clique=2 and chr >= 3          *)
(*        then  circ >= 2*chr-1       *)
(*                                    *)
(************************************)
begin
  if (activerule[105]) and (min[clique] = 2) and (max[chr] >= 3)
then
   begin
     rule:='105/ ';
     if max[clique] = 2 then
         begin
           if max[circ] < infinity then z:=(max[circ]+1) div 2
              else if min[circ] = infinity then z:=2
                       else z:=infinity;
           if z < 2 then z:=2;
           if z < max[chr] then pushmax(chr);
         end;
     z:=min[chr];
     if (max[clique] = 2) and (z >= 3) then
          begin
            z:=2*z-1;
            if z > min[circ] then pushmin(circ);
          end
       else
         if (max[circ] < 2*min[chr]-1) and (min[chr] >= 3) then
                     begin
                       z:=3;
                       pushmin(clique);
                     end
    end;
end;

procedure r106;
(*****************************************)
(*                                       *)
(*  E <= ncov(nind + ncov(chr-1)/(2chr))    *)
(*                                       *)
(*****************************************)
begin
  if activerule[106] then
     begin
       rule:='106/ ';
       if (max[ncov] < infinity) and (max[nind] < infinity) and
          (max[chr] < infinity) then
             begin
               rz:=max[ncov]*(max[chr]-1)/(2*max[chr]);
               z:=trunk(max[ncov]*(max[nind]+rz));
               if z < max[edges] then pushmax(edges);
              end;
        if (max[ncov] < infinity) and (max[chr] < infinity) then
             begin
```

```
               rz:=max[ncov]*(max[ncov]*((max[chr]-1)/(2
*max[chr])));
               z:=round((min[edges]-rz)/max[ncov]+hf);
               if z > min[nind] then pushmin(nind);
             end;
        if (max[nind] < infinity) and (max[chr] < infinity) then
             begin
               rz:=(max[chr]-1)/max[chr];
               z:=max[nind];
               z:=round((-z+sqrt(z*z+2*min[edges]*rz))/rz+hf);
               if z > min[ncov] then pushmin(ncov);
             end;
        if (max[ncov] < infinity) and (max[nind] < infinity) then
             begin
               z:=max[ncov];
               rz:=2*(z*max[nind]+z*z/2-min[edges]);
               if rz >= 0 then
                   begin
                     z:=round(z*z/rz+hf);
                     if z > min[chr] then pushmin(chr);
                   end;
             end;
    end;
end;

procedure r107;
(*******************************)
(*                             *)
(*       mindeg <= maxdeg      *)
(*                             *)
(*******************************)
begin
  if activerule[107] then
    begin
      rule:='107/ ';
      rulea(mindeg,maxdeg,0);
    end;
end;

procedure r108;
(*********************************)
(*                               *)
(*  nccov <= (P+nind-clique+1)/2 *)
(*                               *)
(*********************************)
begin
  if activerule[108] then
     begin
       rule:='108/ ';
       if (max[nodes] < infinity) and (max[nind] < infinity) then
           begin
             z:=(max[nodes]+max[nind]-min[clique]+1) div 2;
```

```
            if z < max[nccov] then pushmax(nccov);
            z:=max[nodes]+max[nind]-2*min[nccov]+1;
            if z < max[clique] then pushmax(clique);
          end;
       if max[nind] < infinity then
          begin
            z:=2*min[nccov]-max[nind]+min[clique]-1;
            if z > min[nodes] then pushmin(nodes);
          end;
       if max[nodes] < infinity then
          begin
            z:=2*min[nccov]-max[nodes]+min[clique]-1;
            if z > min[nind] then pushmin(nind);
          end;
     end;
end;

procedure r109;
(*********************)
(*                   *)
(*  eind >= ncov/2   *)
(*                   *)
(*********************)
begin
  if activerule[109] then
    begin
      rule:='109/ ';
      z:=(min[ncov]+1) div 2;
      if z > min[eind] then pushmin(eind);
      if max[eind] < infinity then
        begin
          z:=2*max[eind];
          if z < max[ncov] then pushmax(ncov);
        end;
     end;
end;

procedure r110;
(*********************************************)
(*                                           *)
(*    if mindeg>=4 and girth>=5 then         *)
(*     circ>=(girth-2)*(mindeg-2)+5          *)
(*                                           *)
(*********************************************)
begin
  if (activerule[110]) and (min[girth] < infinity)
     and (max[mindeg] >= 4) and (max[girth] >= 5) then
       begin
         rule:='110/ ';
         if max[circ] < infinity then
              begin
                if min[mindeg] >= 4 then
```

```
                  begin
                    z:=2+(max[circ]-5) div (min[mindeg]-2);
                    if z < 4 then z:=4;
                    if z < max[girth] then pushmax(girth);
                  end;
                if min[girth] >= 5 then
                  begin
                    z:=2+(max[circ]-5) div (min[girth]-2);
                    if z < 3 then z:=3;
                    if z < max[mindeg] then pushmax(mindeg);
                  end;
              end;
          z:=3;
          if (min[mindeg] > 3) and (min[girth] > 4) then
              begin
                z:=(min[girth]-2)*(min[mindeg]-2)+5;
                if z > min[circ] then pushmin(circ);
              end;
       end;
end;

procedure r111;
(*************************************)
(*                                   *)
(* if connected then diam<=2*nind-1  *)
(*                                   *)
(*************************************)
begin
  rule:='111/ ';
  z:=0;
  if (activerule[111]) and (max[connct] = 1) then
       if min[connct] = 1 then
            begin
              if max[nind] < infinity then
                 begin
                   z:=2*max[nind]-1;
                   if z < max[diam] then pushmax(diam);
                 end;
              z:=(min[diam]+2) div 2;
              if z > min[nind] then pushmin(nind);
            end
          else
            if min[diam] > 2*max[nind]-1 then pushmax(connct);
end;

procedure r112;
(*********************************************************)
(*                                                       *)
(*    if nind <= mindeg, mindeg>=(P+2)/3 and conn then   *)
(*                              hamiltonian              *)
(*                                                       *)
(*********************************************************)
```

```
begin
  if (activerule[112]) and (min[hamil]=0) and (max[nodes]
< infinity) then
        begin
          rule:='112/ ';
          if (max[nind] <= min[mindeg]) and
             (min[mindeg] >= (max[nodes]+2)/3)
             and (min[connct] = 1) then
               begin
                 z:=1;
                 pushmin(hamil);
               end
             else
               if max[hamil]=0 then
                  if (max[nind] <= min[mindeg]) and (min[connct]
= 1) then
                    begin
                      z:=(max[nodes]+1) div 3;
                      if (max[nodes] < infinity) and
                         (z < max[mindeg]) then pushmax(mindeg);
                      z:=3*min[mindeg]-1;
                      if z > min[nodes] then pushmin(nodes);
                    end
                  else
                    if (min[mindeg] >= (max[nodes]+2)/3) and
                       (min[connct] = 1) then
                       begin
                         z:=min[mindeg]+1;
                         if z > min[nind] then pushmin(nind);
                         z:=max[nind]-1;
                         if z < max[mindeg] then
pushmax(mindeg);
                       end
                     else
                       if (max[nind] <= min[mindeg]) and
                          (min[mindeg] >= (max[nodes]+2)/3) then
                            begin
                              z:=0;
                              pushmax(connct);
                            end;
          end;
end;

procedure r113;
(*********************************************)
(*                                           *)
(*  E >= nind*mindeg+(clique-1)*(clique-2)/2 *)
(*                                           *)
(*********************************************)
begin
  if activerule[113] then
    begin
```

```
      rule:='113/ ';
      z1:=((min[clique]-1)*(min[clique]-2)) div 2;
      if max[edges] < infinity then
           begin
              z:=(max[edges]-z1) div min[mindeg];
              if z < max[nind] then pushmax(nind);
              z:=(max[edges]-z1) div min[nind];
              if z < max[mindeg] then pushmax(mindeg);
              z:=1+8*(max[edges]-min[nind]*min[mindeg]);
              z:=trunk((3+sqrt(z))/2);
              if z < max[clique] then pushmax(clique);
            end;
       z:=min[nind]*min[mindeg]+z1;
       if z > min[edges] then pushmin(edges);
     end;
end;

procedure r114;
(*****************************************)
(*                                       *)
(*  E >= ncov + (clique-1)*(clique-2)/2  *)
(*                                       *)
(*****************************************)
begin
  if activerule[114] then
     begin
       rule:='114/ ';
       z1:=(min[clique]-1)*(min[clique]-2) div 2;
       z:=min[ncov]+z1;
       if z > min[edges] then pushmin(edges);
       if max[edges] < infinity then
         begin
           z:=max[edges]-z1;
           if z < max[ncov] then pushmax(ncov);
           z:=1+8*(max[edges]-min[ncov]);
           z:=trunk((3+sqrt(z))/2);
           if z < max[clique] then pushmax(clique);
         end;
     end;
end;

procedure r115;
(************************************************)
(*                                              *)
(*   E >= chr*(chr-3)/2 + P - Ncomp+1           *)
(*                                              *)
(************************************************)
begin
  if activerule[115] then
    begin
      rule:='115/ ';
      z:=(min[chr]*(min[chr]-3)) div 2 +min[nodes]-max[ncomp]+1;
```

```
      if z > min[edges] then pushmin(edges);
      if max[edges] < infinity then
              begin
                z:=max[edges]-z+min[nodes];
                if z < max[nodes] then pushmax(nodes);
                z:=min[nodes]-z+max[ncomp];
                if z > min[ncomp] then pushmin(ncomp);
                z:=1+8*(max[edges]-min[nodes]+max[ncomp]);
                z:=trunk((3+sqrt(z))/2);
                if z < max[chr] then pushmax(chr);
              end;
      end;
end;

procedure r116;
(*********************************************)
(*                                           *)
(*      if bipartite then genus <=           *)
(*        upper((P-4)**2/16)    P even       *)
(*        upper((P-3)(P-5)/16)  P odd        *)
(*                                           *)
(*********************************************)
begin if (activerule[116]) and (max[bipart]=1) then
  begin
    rule:='116/ ';
    z:=max[nodes];
    if min[bipart] = 1 then
      begin
        if z < infinity then
           begin
             if (odd(z)) or (pRodd = eq) then z:=((z-3)*(z-5)+15)
div 16
                   else z:=((z-4)*(z-4)+15) div 16;
             if z < max[genus] then pushmax(genus);
           end;
         if min[genus] > 0 then
            begin
              z:=round(4+sqrt(16*min[genus]-15)+hf);
              if z > min[nodes] then pushmin(nodes);
            end;
       end
        else if z < infinity then
          begin
             if (odd(z)) or (pRodd = eq) then z:=((z-3)*(z-5)+15)
div 16
                   else z:=((z-4)*(z-4)+15) div 16;
             if min[genus] > z then
                     begin
                       z:=0;
                       pushmax(bipart);
                     end;
          end;
```

```
   end;
end;

procedure r117;
(*********************************************)
(*                                           *)
(*   if not complete  then                   *)
(*               Nconn>=2*mindeg-P+2         *)
(*                                           *)
(*********************************************)
begin
  if (activerule[117]) and (min[compl]= 0) then
    begin
      rule:='117/ ';
      z:=2*min[mindeg]-max[nodes]+2;
      if max[compl]=0 then
          begin
            if (max[nodes] < infinity) and (z > min[nconn]) then
pushmin(nconn);
            if max[nconn] < infinity then
                   begin
                     z:=2*min[mindeg]+2-max[nconn];
                     if z > min[nodes] then pushmin(nodes);
                     if max[nodes] < infinity then
                        begin
                          z:=(max[nconn]+max[nodes]-2) div 2;
                          if z < max[mindeg] then
pushmax(mindeg);
                        end;
                   end;
               end
            else
              if (max[nconn] < z) and (max[nodes] < infinity)
then
                          begin
                            z:=1;
                            pushmin(compl);
                          end;
    end;
end;

procedure r118;
(*************************************************)
(*                                               *)
(*      if P>=6, even, and E>P**2/4    or        *)
(*         P>=7,odd, and E>(P-1)**2/4+mindeg then *)
(*               circ>=5                         *)
(*                                               *)
(*************************************************)
begin
  if (activerule[118]) and (max[nodes] < infinity) and
(min[circ] > 5)
```

```
     and (min[circ] < infinity) then
    begin
      rule:='118/ ';
      rlb:=max[nodes]*(max[nodes]/4);
      rhb:=(max[nodes]-1)*((max[nodes]-1)/4)+max[mindeg];
      if ((not(odd(max[nodes]))) and (min[edges] > rlb)) or
         ((odd(max[nodes])) and (min[edges] > rhb)) then
            begin
              z:=5;
              pushmin(circ);
            end;
     end;
end;

procedure r119;
(************************************************)
(*                                              *)
(*  if  chr > clique  then                      *)
(*       mindeg <= (3*clique-4)*P/(3*clique-1)  *)
(*                                              *)
(************************************************)
begin
  if activerule[119] then
   begin
     rule:='119/ ';
     if max[nodes] < infinity then
        begin
          z:=max[nodes]-min[mindeg];
          z:=(4*max[nodes]-min[mindeg]-1) div (3*z)+1;
          if min[chr] < z then z:=min[chr];
          if z > min[clique] then pushmin(clique);
        end;
     if min[chr] > max[clique] then
       begin
         if max[nodes] < infinity then
           begin
             z:=(3*max[clique]-4)*max[nodes] div (3
*max[clique]-1);
             if z < max[mindeg] then pushmax(mindeg);
           end;
         z:=((3*max[clique]-1)*min[mindeg]-1) div (3
*max[clique]-4)+1;
         if z > min[nodes] then pushmin(nodes);
       end
      else
       begin
         rz:=max[clique];
         rz:=(3*rz-4)*max[nodes];
         if rz < infinity then
           begin
             z:=(3*max[clique]-4)*max[nodes];
             if min[mindeg] > z div (3*max[clique]-1) then
```

```
rulea(chr,clique,0);
            end;
        end;
   end;
end;

procedure r120;
(*********************************************************)
(*                                                       *)
(*   if Hamiltonian then                                 *)
(*       if P > chr >=4  then                            *)
(*             E >= P + (chr-1)*(chr-2)/2                *)
(*       else                                            *)
(*         if chr = 3 and P is even then E >= P+1        *)
(*                                                       *)
(*********************************************************)
begin
  if (activerule[120]) and (min[hamil] = 1) then
    begin
      rule:='120/ ';
      if (min[nodes] > max[chr]) and (min[chr] >= 4) then
           begin
             z:=min[nodes]+(min[chr]-1)*(min[chr]-2) div 2;
             if z > min[edges] then pushmin(edges);
             if max[edges] < infinity then
              begin
                z:=max[edges]-(min[chr]-1)*(min[chr]-2) div 2;
                if z < max[nodes] then pushmax(nodes);
                z:=1-8*(min[nodes]-max[edges]);
                z:=trunk((3+sqrt(z))/2);
                if z < max[chr] then pushmax(chr);
              end;
           end
         else
           if (min[chr] = 3) and
              ((min[nodes]=max[nodes]) and (not(odd(min[nodes])))
or
               (pReven=eq)) then
                begin
                  pReven:=eq;
                  pRodd:=ne;
                  rulea(nodes,edges,-1);
                end;
     end;
end;

procedure r121;
(*****************************)
(*                           *)
(*  chr <= P-Nconn*(diam-3)-2  *)
(*                           *)
(*****************************)
```

```
begin
  if (activerule[121]) and (max[diam] < infinity) then
    begin
      rule:='121/ ';
      if min[diam] < 3 then z1:=max[nconn]
                       else z1:=min[nconn];
      z:=min[chr]+z1*(min[diam]-3)+2;
      if z > min[nodes] then pushmin(nodes);
      if max[nodes] < infinity then
              begin
                z:=max[nodes]-z1*(min[diam]-3)-2;
                if z < max[chr] then pushmax(chr);
                if min[diam] > 3 then
                        begin
                          z:=(max[nodes]-min[chr]-2) div
(min[diam]-3);
                          if z < max[nconn] then pushmax(nconn);
                        end
                    else
                      if max[diam] < 3 then
                            begin
                              z:=(min[chr]-max[nodes]+1) div (3-
min[diam])+1;
                              if z > min[nconn] then
pushmin(nconn);
                            end;
                if min[nconn] > 0 then
                  begin
                    z:=(max[nodes]-min[chr]-2) div min[nconn] +3;
                    if z < max[diam] then pushmax(diam);
                  end;
              end;
    end;
end;

procedure r122;
(*****************************************)
(*                                       *)
(*  let k=P(P-1)/2-E                     *)
(*    if E > trunc(P**2/4) then          *)
(*         eccov<=k+(1+sqrt(1+4k))/2      *)
(*                                       *)
(*****************************************)
begin
  if (activerule[122]) and (max[nodes] < infinity) then
    begin
      rule:='122/ ';
      z:=max[nodes];
      k:=z*(z-1) div 2 - min[edges];
      z1:=trunk(k+(1+sqrt(1+4*k))/2);
      if min[edges] > z*z div 4 then
           begin
```

```
             z:=z1;
             if z < max[eccov] then pushmax(eccov);
            end
          else
            if min[eccov] > z1 then
               begin
                 z:=z*z div 4;
                 if z < max[edges] then pushmax(edges);
                 z:=round(2*sqrt(min[edges])+hf);
                 if z > min[nodes] then pushmin(nodes);
                end;
     end;
end;

procedure r123;
(*************************************************)
(*                                               *)
(*       if  P <= 2*Econn+3                      *)
(*            then mindeg=Econn                  *)
(*                                               *)
(*************************************************)
begin
  if activerule[123] then
    begin
      rule:='123/ ';
      if max[nodes] <= 2*min[econn]+3 then
            rulea(mindeg,econn,0)
        else
          if max[econn] < min[mindeg] then
             begin
               z:=2*min[econn]+4;
               if z > min[nodes] then pushmin(nodes);
               z:=max[nodes];
               if z < infinity then
                    begin
                      z:=(z-4) div 2;
                      if z < max[econn] then pushmax(econn);
                    end;
             end;
    end;
end;

procedure r124;
(*********************************)
(*                               *)
(*   if connected then           *)
(*      Bwidth >= (P-1)/diam     *)
(*                               *)
(*********************************)
begin if (activerule[124]) and (max[connct]=1) then
  begin
    rule:='124/ ';
```

```
    z:=(min[nodes]-2+max[diam]) div max[diam];
    if min[connct] = 1 then
     begin
       if max[diam] < infinity then
            begin
              if z > min[bwidth] then pushmin(bwidth);
              if max[bwidth] < infinity then
                      begin
                        z:=max[bwidth]*max[diam]+1;
                        if z > max[nodes] then pushmax(nodes);
                      end;
            end;
        if max[bwidth] < infinity then
             begin
               z:=(min[nodes]-2) div max[bwidth]+1;
               if z > min[diam] then pushmin(diam);
             end;
       end
       else
         if max[bwidth] < z then
              begin
                z:=0;
                pushmax(connct);
              end;
    end;
end;

procedure r125;
(*************************************************)
(*                                               *)
(*      if genus >= 1 then                       *)
(*          P >= trunc((9*girth-9)/4) + k        *)
(*               where                           *)
(*                      k = 1 if girth=3 mod 4   *)
(*                        = 0 otherwise          *)
(*                                               *)
(*************************************************)
begin
  if (activerule[125]) and (max[genus] > 0) then
     begin
       rule:='125/ ';
       z:=min[girth];
       if (z = max[girth]) and (z+1 = ((z+1) div 4)*4) then k:=1
                                                       else k:=0;
       z:=(9*z-9) div 4 +k;
       if min[genus] > 0 then
          begin
            if z > min[nodes] then pushmin(nodes);
            if max[nodes] < infinity then
              begin
                z:=((max[nodes]-k)*4+12) div 9;
                if z < max[girth] then pushmax(girth);
```

```
              end;
          end
        else
          if max[nodes] < z then
                begin
                   z:=0;
                   pushmax(genus);
                 end;
      end;
end;

procedure r126;
(*****************************************)
(*                                       *)
(*     if mindeg >= P div 2 then         *)
(*                  P <= 2*Econn+3       *)
(*                                       *)
(*****************************************)
begin
  if activerule[126] then
    begin
      rule:='126/ ';
      z:=2*max[econn]+3;
      if min[mindeg] >= max[nodes] div 2 then
              begin
                if z < max[nodes] then pushmax(nodes);
                z:=(min[nodes]-2) div 2;
                if z > min[econn] then pushmin(econn);
              end
        else
          if min[nodes] > z then
             begin
               if max[nodes] < infinity then
                   begin
                     z:=max[nodes] div 2 -1;
                     if z < max[mindeg] then pushmax(mindeg);
                   end;
               z:=2*min[mindeg]+2;
               if z > min[nodes] then pushmin(nodes);
             end;
    end;
end;

procedure r127;
(*************************************************)
(*                                               *)
(*    Hamiltonian,P even,maxdeg=3 ==> echr=maxdeg   *)
(*                                               *)
(*************************************************)
begin
  if (activerule[127]) and (max[hamil]=1) and (max[echr] >= 4)
then
```

```
    begin
      rule:='127/ ';
      if (min[hamil]=1) and ((pReven=eq) or
((max[nodes]=min[nodes]) and
          (not(odd(max[nodes])))) and (min[maxdeg]=max[maxdeg])
and
          (max[maxdeg]=3) then
            begin
              z:=3;
              pushmax(echr);
            end
        else
         if (min[echr]=4) and (min[hamil]=1) and ((pReven=eq) or
            ((max[nodes]=min[nodes]) and (not(odd(max[nodes])))))
then
             begin
               if max[maxdeg]=3 then
                    begin
                      z:=2;
                      pushmax(maxdeg);
                    end
                 else
                   if min[maxdeg]=3 then
                      begin
                        z:=4;
                        pushmin(maxdeg);
                      end;
               end
            else
              if (min[maxdeg] = max[maxdeg]) and (max[maxdeg] =
3) and
                 (min[echr] = 4) and (min[hamil] = 1) then
                   begin
                     pReven:=ne;
                     if not(odd(max[nodes])) then
                           begin
                             z:=max[nodes]-1;
                             pushmax(nodes);
                           end;
                     if not(odd(min[nodes])) then
                           begin
                             z:=min[nodes]+1;
                             pushmin(nodes);
                           end;
                    end
                else
                  if (((max[nodes] = min[nodes]) and
(not(odd(max[nodes])))) or
                     (pReven = eq)) and (min[maxdeg] =
max[maxdeg]) and
                     (max[maxdeg] = 3) and (min[echr] = 4) then
                      begin
```

```
                      z:=0;
                      pushmax(hamil);
                    end;
    end;
end;

procedure r128;
(*********************************************************)
(*                                                       *)
(*    if P >= 2*eind+1 then                              *)
(*     if maxdeg<= 2*eind then                           *)
(*      |  if P<=2*eind+eind/((maxdeg+1)/2)) then        *)
(*      |   |   if maxdeg odd then                       *)
(*      |   |    |  E<=min(P*maxdeg/2,eind*maxdeg+      *)
(*      |   |    | 2*((P-eind)/(maxdeg+3))(maxdeg-1)/2  *)
(*      |   |    |else E<=P*maxdeg/2                     *)
(*      |   |else                                       *)
(*      |         E<=eind*maxdeg+eind/((maxdeg+1)/2)*   *)
(*      |                 maxdeg/2                      *)
(*      |else if P >=eind+maxdeg then                    *)
(*             |          E<=eind*maxdeg                *)
(*             |else                                    *)
(*                E<=max(eind*(2*eind+1),               *)
(*                    eind*(P+maxdeg-eind)/2)           *)
(*                                                       *)
(*********************************************************)
var p:longint;
begin
  p:=max[nodes];
  if activerule[128] then
   if (min[nodes]>=2*max[eind]+1) and (max[nodes] < infinity) and
      (max[eind] < infinity) and (max[maxdeg] < infinity) then
      begin
        rule:='128/ ';
        if max[maxdeg] <= 2*min[eind] then
          begin
            z:=2*min[eind]+min[eind] div ((max[maxdeg]+1) div 2);
            if p <= z then
               begin
                 if max[maxdeg]=min[maxdeg] then
                    if odd(max[maxdeg]) then
                          begin
                            z:=2*(p-max[eind]) div (max[maxdeg]+
3);
                            z:
=max[eind]*max[maxdeg]+z*(max[maxdeg]-1) div 2;
                            if z > p*max[maxdeg] div 2 then
                                    z:=p*max[maxdeg] div 2;
                          end
                        else z:=p*max[maxdeg] div 2
                    else z:=max[edges];
               end
```

```
             else
               begin
                 z:=2*max[eind]+max[eind] div ((min[maxdeg]+1)
div 2);
                 if min[nodes] > z then
                   begin
                     z:=max[eind];
                     if (min[maxdeg]=max[maxdeg]) and
(odd(max[maxdeg])) then
                          z:=(z*(max[maxdeg] div 2)) div
((max[maxdeg]+1) div 2);
                     z:=max[eind]*max[maxdeg]+z;
                   end
                  else z:=max[edges];
               end;
           end
         else
           begin
             z:=max[edges];
             if min[maxdeg] > 2*max[eind] then
               if min[nodes] >= max[eind]+max[maxdeg] then
                    z:=max[eind]*max[maxdeg]
                 else if p < min[eind]+min[maxdeg] then
                   begin
                     z:=max[eind]*(p+max[maxdeg]-max[eind]) div
2;
                     if z < max[eind]*(2*max[eind]+1) then
                                z:=max[eind]*(2*max[eind]+1);
                   end;
            end;
       if z < max[edges] then pushmax(edges);
     end;
end;

procedure r129;
(*********************************************)
(*                                           *)
(*      if 4 <= diam < infinity and          *)
(*        let k= diam div 3 +1               *)
(*           mindeg <= (P-d-3)/k+2           *)
(*                                           *)
(*********************************************)
begin
  if (activerule[129]) and (max[diam] < infinity) and
(min[diam] > 3) then
    begin
      rule:='129/ ';
      k:=min[diam] div 3 +1;
      if max[nodes] < infinity then
             begin
               z:=(max[nodes]-min[diam]-3) div k+2;
               if z < max[mindeg] then pushmax(mindeg);
```

```
               z:=(3*max[nodes]-6) div (min[mindeg]+1)-1;
               z1:=(min[mindeg]-2)*(z div 3)+z+min[mindeg]+1;
               while z1 > max[nodes] do
                    begin
                      z:=z-1;
                      z1:=(min[mindeg]-2)*(z div
3)+z+min[mindeg]+1;
                    end;
               if z < max[diam] then pushmax(diam);
             end;
       z:=min[mindeg]*k+min[diam]-2*k+3;
       if z > min[nodes] then pushmin(nodes);
     end;
end;

procedure r130;
(*************************************************)
(*                                               *)
(*      if diam = 2 then                         *)
(*          if maxdeg**2 <= 8*P                  *)
(*             then E >= P(P-1)/(2*maxdeg)       *)
(*             else E >= P(P-1)/(maxdeg+8*P/maxdeg) *)
(*                                               *)
(*************************************************)
begin
  if activerule[130] then
    if (max[diam]=min[diam]) and (max[diam]=2) and (max[maxdeg]
< infinity) then
          begin
            rule:='130/ ';
            k:=max[maxdeg];
            z1:=min[nodes];
            z:=round((z1-1)/(k+8*z1/k)*z1+hf);
            z1:=round((z1-1)/(2*k)*z1+hf);
            if z1 < z then z:=z1;
            if z > min[edges] then pushmin(edges);
            z1:=min[nodes];
            if k*k <= 8*z1 then
                   begin
                     z:=round((z1-1)/(2*k)*z1+hf);
                     if z > min[edges] then pushmin(edges);
                     if max[edges] < infinity then
                       begin
                         z:=round((z1-1)/(2*max[edges])*z1+hf);
                         if z > min[maxdeg] then pushmin(maxdeg);
                         rz:=sqrt(1+8.0*k*max[edges]);
                         z:=trunk((1+rz)/2);
                         if z < max[nodes] then pushmax(nodes);
                       end;
                   end
               else
                if min[maxdeg]*min[maxdeg] > 8*max[nodes] then
```

```
                  begin
                    z:=round((z1-1)/(k+8*z1/k)*z1+hf);
                    if z > min[edges] then pushmin(edges);
                  end;
          end;
end;

procedure r131;
(*************************************************)
(*                                               *)
(*      chr <= maxdeg + 1 - (maxdeg+1) div       *)
(*                      max(4,clique+1)          *)
(*                                               *)
(*************************************************)
begin
  if (activerule[131]) and (max[clique] < infinity) then
    begin
      rule:='131/ ';
      if max[clique] > 3 then k:=max[clique]+1
                         else k:=4;
      if max[maxdeg] < infinity then
         begin
           z:=(max[maxdeg]+1) div k;
           z:=max[maxdeg]+1-z;
           if z < max[chr] then pushmax(chr);
         end;
      z:=(min[chr]*k-1) div (k-1)-1;
      if z > min[maxdeg] then pushmin(maxdeg);
    end;
end;

procedure r132;
(*********************************************************)
(*                                                       *)
(*      mindeg <= z/(P-maxdeg-1)                         *)
(*          where      |  ((P-1)/2)**2     if P<>4t+3    *)
(*                z  =|                                  *)
(*                   |  (P-3)*(P+1)/4     if P=4t+3     *)
(*                                                       *)
(*********************************************************)
begin
  if (activerule[132]) and (max[nodes] = min[nodes]) and
     (max[maxdeg] < infinity) then
      begin
        rule:='132/ ';
        z:=max[nodes];
        if z+1=((z+1) div 4)*4 then k:=(z-3)*(z+1) div 4
           else
             begin
               rz:=(z-1)/2;
               k:=trunk(rz*rz);
             end;
```

```
        if max[maxdeg] < z-1 then
           begin
             z:=k div (z-max[maxdeg]-1);
             if z < max[mindeg] then pushmax(mindeg);
           end;
        z:=max[nodes]-1-k div min[mindeg];
        if z > min[maxdeg] then pushmin(maxdeg);
      end;
end;

procedure r133;
(*******************************************************)
(*                                                     *)
(*      let z = 1 + 2*E div P                          *)
(*          k = round(P - 2*E/z+0.5)                   *)
(*      then  nind >= k+round((P-k*z)/(z+1)+0.5)       *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[133]) and (max[nodes]=min[nodes]) and
(min[edges]=max[edges])
       and (min[edges] < infinity) then
      begin
        rule:='133/ ';
        z:=1+(2*min[edges]) div max[nodes];
        k:=max[nodes]-(2*min[edges]) div z;
        z:=k+(max[nodes]-k*z+z) div (z+1);
        if z > min[nind] then pushmin(nind);
      end;
end;

procedure r134;
(************************************************)
(*                                              *)
(*      if rad=diam=2  then                     *)
(*                  |  4         if P=4         *)
(*           E  >=  |                           *)
(*                  |  2*P-5     if P>=5        *)
(*                                              *)
(************************************************)
begin
  if (activerule[134]) and (min[radius] = max[diam]) and
(max[diam] = 2)
     and (min[nodes] >= 4) then
   begin
     rule:='134/ ';
     if min[nodes] = 4 then
         begin
           z:=4;
           if z > min[edges] then pushmin(edges);
         end
       else
```

```
         begin
           z:=2*min[nodes]-5;
           if z > min[edges] then pushmin(edges);
           if max[edges] < infinity then
                begin
                  z:=(max[edges]+5) div 2;
                  if z < max[nodes] then pushmax(nodes);
                end;
          end;
    end;
end;

procedure r135;
(*************************)
(*                       *)
(*   E >= 2*ncov - Ncomp    *)
(*                       *)
(*************************)
begin
  if activerule[135] then
   begin
     rule:='135/ ';
     z:=2*min[ncov]-max[ncomp];
     if z > min[edges] then pushmin(edges);
     if max[edges] < infinity then
           begin
             z:=(max[edges]+max[ncomp]) div 2;
             if z < max[ncov] then pushmax(ncov);
             z:=2*min[ncov]-max[edges];
             if z > min[ncomp] then pushmin(ncomp);
           end;
    end;
end;

procedure r136;
(*********************************************************)
(*                                                       *)
(*  if maxdeg <= 2*eind and maxdeg is odd  then          *)
(*      E<=eind*maxdeg+((maxdeg-1)/2)*(2*eind/(maxdeg+1))   *)
(*                                                       *)
(*********************************************************)
begin
  if activerule[136] then
    begin
      rule:='136/ ';
      z:=max[maxdeg];
      z:=max[eind]*z+((z-1) div 2)*((2*max[eind]) div (z+1));
      if (min[maxdeg]=max[maxdeg]) and (odd(max[maxdeg])) and
         (max[eind] < infinity) and (max[maxdeg] <= 2*min[eind])
then
           begin
             if z < max[edges] then pushmax(edges);
```

```
              z:=max[maxdeg]+1;
              z:=(min[edges]*z-1) div (z*z-2)+1;
              if z > min[eind] then pushmin(eind);
            end
           else
             if (min[edges] > z) and (min[maxdeg] = max[maxdeg])
and
                (odd(max[maxdeg])) then
                   begin
                     z:=(max[maxdeg]-1) div 2;
                     if z < max[eind] then pushmax(eind);
                   end;
     end;
end;

(*                 R137 has been "RETIRED"
procedure r137;                                                 *)
(*************************************************************)
(*                                                           *)
(*      E >= P*(P-1)/2*((maxdeg-2) / ((maxdeg-1)**diam-1))   *)
(*                                                           *)
(*************************************************************)

procedure r138;
(*****************************************************)
(*                                                   *)
(*      if E >= (P**2-5*P+14)/2 then circ>=P-1       *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[138]) and (max[nodes] < infinity) then
    begin
      rule:='138/ ';
      z:=max[nodes];
      z1:=(z*z-5*z+14) div 2;
      if min[edges] >= z1 then
                   begin
                     z:=min[nodes]-1;
                     if z > min[circ] then pushmin(circ);
                     z:=max[circ]+1;
                     if z < max[nodes] then pushmax(nodes);
                   end
                 else
                   if max[circ] < min[nodes]-1 then
                      begin
                        z:=z1-1;
                        if z < max[edges] then pushmax(edges);
                        if min[edges] > 3 then
                             begin
                               z:=round((5+sqrt(8*min[edges]-27))/2
+hf);
                               if z > min[nodes] then
```

```
pushmin(nodes);
                            end;
                    end;
    end;
end;

procedure r139;
(*************************************************************)
(*                                                           *)
(*    if E >= (circ*(2*P-circ)+1)/4 then girth=3             *)
(*                                                           *)
(*************************************************************)
begin
  if (activerule[139]) and (max[girth] > 3) and (max[nodes]
< infinity)
      and (max[circ] < infinity) then
    begin
      rule:='139/ ';
      z:=max[nodes];
      z:=(max[circ]*(2*z-max[circ])+4) div 4;
      if min[edges] >= z then
          begin
            z:=3;
            pushmax(girth);
          end
        else
          if min[girth] > 3 then
                begin
                  z:=z-1;
                  if z < max[edges] then pushmax(edges);
                  z:=max[nodes];
                  rz:=z*z-4*min[edges];
                  if rz >= 0 then
                     begin
                       z:=round(z-sqrt(rz)+hf);
                       if z > min[circ] then pushmin(circ);
                     end;
             (*     z:=round(2*min[edges]/max[circ]+max[circ]/2
+hf);
                    if z > min[nodes] then pushmin(nodes);  *)
                end;
      end;

end;

procedure r140;
(*********************************************)
(*                                           *)
(*    if E >= 4*P then                       *)
(*             Xnum > E**3/(100*P**2)        *)
(*                                           *)
(*********************************************)
```

```
begin
  if activerule[140] then
     begin
       rule:='140/ ';
       if max[xnum] < infinity then
         begin
           if max[xnum] > 0 then
             begin
               rz:=min[edges];
               z:=round(sqrt((rz*rz*rz+1)/max[xnum])/10+hf);
               if z > (min[edges]+4) div 4 then z:=(min[edges]+4)
div 4;
               if z > min[nodes] then pushmin(nodes);
             end;
           if max[nodes] < infinity then
              begin
                rz:=100.0*max[nodes]*max[nodes]*max[xnum]+1;
                z:=trunk(root(rz,3));
                if z < 4*max[nodes]-1 then z:=4*max[nodes]-1;
                if z < max[edges] then pushmax(edges);
             end;
         end;
       k:=max[nodes];
       if k < infinity then
        begin
          rz:=(min[edges]/k)/10;
          z:=round((min[edges]*rz*rz+0.01)+hf);
          if (min[edges] >= 4*k) and (z > min[xnum]) and (z
< infinity)
                then pushmin(xnum);
        end;
     end;
end;

procedure r141;
(*************************************)
(*                                   *)
(*        circ >= 2*E/(P-1)          *)
(*                                   *)
(*************************************)
begin
  if activerule[141] then
     begin
       rule:='141/ ';
       z:=max[nodes];
       if z < infinity then
         begin
           z:=(2*min[edges]+z-2) div (z-1);
           if z > min[circ] then pushmin(circ);
           if max[circ] < infinity then
               begin
                 z:=(max[circ]*(max[nodes]-1)) div 2;
```

```
                if z < max[edges] then pushmax(edges);
               end;
        end;
       z:=max[circ];
       if z < infinity then
             begin
               z:=(2*min[edges]+z-2) div max[circ]+1;
               if z > min[nodes] then pushmin(nodes);
             end;
     end;
end;

(*  procedure r142;  *)
(*********************************************************)
(*                                                       *)
(*      if E >1+P+sqrt(P**3)/2  then                     *)
(*                              girth <= 4               *)
(*                                                       *)
(*********************************************************)
(*                                                       *)
(*     REPLACED BY  R343                                 *)
(*                                                       *)
(*********************************************************)

procedure r143;
(*********************************************)
(*                                           *)
(*       let   R=nccov                       *)
(*         R <= .5+sqrt(.25+P**2-P-2*E)      *)
(*                                           *)
(*********************************************)
begin
  if activerule[143] then
     begin
       rule:='143/ ';
       if max[nodes] < infinity then
         begin
           k:=max[nodes];
           z:=trunk(0.5+sqrt(0.25+k*k-k-2*min[edges]));
           if z < max[nccov] then pushmax(nccov);
           z:=min[nccov];
           z:=(k*k-k-z*z+z) div 2;
           if z < max[edges] then pushmax(edges);
         end;
       k:=min[nccov];
       k:=1+8*min[edges]+4*(k*k-k);
       z:=round((1+sqrt(k))/2+hf);
       if z > min[nodes] then pushmin(nodes);
     end;
end;

procedure r144;
```

```
(*********************************)
(*                               *)
(*   chr <= .5 + sqrt(.25+2*E)   *)
(*                               *)
(*********************************)
begin
  if activerule[144] then
     begin
       rule:='144/ ';
       if max[edges] < infinity then
             begin
               z:=trunk(0.5+sqrt(0.25+2*max[edges]));
               if z < max[chr] then pushmax(chr);
             end;
       z:=min[chr];
       z:=(z*z-z+1) div 2;
       if z > min[edges] then pushmin(edges);
     end;
end;

procedure r145;
(*************************************)
(*                                   *)
(*  if clique = 2 then               *)
(*         nind>=(sqrt(8*P+9)-3)/2   *)
(*                                   *)
(*************************************)
begin
  if (activerule[145]) and (min[clique] = 2) then
    begin
      rule:='145/ ';
      if max[clique]=2 then
                     begin
                       z:=round((sqrt(8*min[nodes]+9)-3)/2+hf);
                       if z > min[nind] then pushmin(nind);
                       if max[nind] < infinity then
                          begin
                            k:=max[nind];
                            z:=(k*k+3*k) div 2;
                            if z < max[nodes] then
pushmax(nodes);
                          end;
                     end
                else
                  begin
                    rz:=max[nind];
                    rz:=rz*(rz+3);
                    if rz < infinity then
                        if max[nind]*(max[nind]+3) < 2*min[nodes]
then
                              begin
                                z:=3;
```

```
                                pushmin(clique);
                              end;
                  end;
    end;
end;

procedure r146;
(*********************************************)
(*                                           *)
(*    if connected then  Bwidth <=P-diam      *)
(*                                           *)
(*********************************************)
begin
  if (activerule[146]) and (min[connct]=1) then
     begin
       rule:='146/ ';
       rulee(nodes,bwidth,diam,0)
     end;
end;

procedure r147;
(*************************************************)
(*                                               *)
(*   if clique=2 then                            *)
(*             2  if maxdeg >= P-2 or P <= 4     *)
(*  chr <=     3  if 5 <= P <= 10                *)
(*             (P-maxdeg+10)/4  otherwise        *)
(*                                               *)
(*************************************************)
begin
  if (activerule[147]) and (min[clique] < 3) and (max[nodes]
< infinity) then
     begin
       rule:='147/ ';
       z:=3;
       if max[clique]=2 then
          begin
            if max[nodes] < 11 then
                 begin
                   if (min[maxdeg] >= max[nodes]-2) or
                      (max[nodes] < 5) then z:=2;
                 end
                else z:=(max[nodes]-min[maxdeg]+10) div 4;
            if z < max[chr] then pushmax(chr);
           end
         else
           if min[chr] > (max[nodes]-min[maxdeg]+10) div 4 then
pushmin(clique);
     end;
end;

procedure r148;
```

```
(*****************************************************)
(*                                                   *)
(*  if 3-reg,plnar, and Econn > 1 then echr = maxdeg *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[148]) and (max[plnar] = 1) and (max[econn] > 1)
     and (min[mindeg] <= 3) and (max[mindeg] >= 3) and
     (min[maxdeg] <= 3) and (max[maxdeg] >= 3) and (max[reg] = 1)
then
     begin
       rule:='148/ ';
       z:=0;
       if (min[mindeg]=max[maxdeg]) and (min[mindeg]=3) and
          (min[plnar]=1) and (min[econn] > 1) then
              rulea(echr,maxdeg,0)
            else
              if max[maxdeg] < min[echr] then
                if (min[mindeg]=max[maxdeg]) and (min[maxdeg]=3)
and
                   (min[plnar]=1) then
                     begin
                       z:=1;
                       pushmax(econn);
                     end
                 else
                   if (min[mindeg]=max[maxdeg]) and (min[mindeg]=
3) and
                      (min[econn] > 1) then pushmax(plnar)
                     else
                       if (min[plnar]=1) and (min[econn] > 1)
then
                          if min[reg]=1 then
                               if min[mindeg]=3 then
                                 begin
                                   z:=4;
                                   pushmin(mindeg);
                                 end
                                else
                                  if max[mindeg]=3 then
                                       begin
                                         z:=2;
                                         pushmax(mindeg);
                                       end;
    end;
end;

procedure r149;
(*****************************************************)
(*                                                   *)
(* if plnar and maxdeg >= 8 then echr = maxdeg     *)
(*                                                   *)
```

```
(************************************************)
begin
  if (activerule[149]) and (max[plnar] = 1) and (max[maxdeg] >=
8) then
    begin
      rule:='149/ ';
      if (min[plnar]=1) and (min[maxdeg] > 7) then
           rulea(echr,maxdeg,0)
        else
          if max[maxdeg] < min[echr] then
             if min[plnar]=1 then
                 begin
                   z:=7;
                   pushmax(maxdeg);
                 end
               else
                 if min[maxdeg] > 7 then
                    begin
                      z:=0;
                      pushmax(plnar);
                    end;
     end;
end;

procedure r150;
(************************************************)
(*                                              *)
(* if  spectr <= maxdeg/2 then echr = maxdeg    *)
(*                                              *)
(************************************************)
begin
  if activerule[150] then
    begin
      rule:='150/ ';
      if lammax <= min[maxdeg]/2 then
             rulea(echr,maxdeg,0)
          else
             if max[maxdeg] < min[echr] then
                begin
                  rz:=min[maxdeg]/2;
                  if rz > lammin then pushlammin;
                  z:=trunk(lammax*2);
                  if z < max[maxdeg] then pushmax(maxdeg);
                end;
     end;
end;


end.
```