```
unit rules400;

interface

 uses
   globals,cmmnds1,pusherr,pushStack,ruleAtoF;



     procedure r351; procedure r352; procedure r353; procedure
r354; procedure r355;
     procedure r356; procedure r357; procedure r358; procedure
r359; procedure r360;
     procedure r361; procedure r362; procedure r363; procedure
r364; procedure r365;
     procedure r366; procedure r367; procedure r368; procedure
r369; procedure r370;
     procedure r371; procedure r372; procedure r373; procedure
r374; procedure r375;
     procedure r376; procedure r377; procedure r378; procedure
r379; procedure r380;
     procedure r381; procedure r382; procedure r383; procedure
r384; procedure r385;
     procedure r386; procedure r387; procedure r388; procedure
r389; procedure r390;
     procedure r391; procedure r392; procedure r393; procedure
r394; procedure r395;
     procedure r396; procedure r397; procedure r398; procedure
r399; procedure r400;


implementation




procedure r351;
(************************************************)
(*                                              *)
(*  if g >= max[(p+1)/2,5] and e >= p+3 then    *)
(*      a) g <= 8, g <> 7 , p = 2g-1             *)
(*      b) nconn=econn=mindeg=2                 *)
(*      c) e = p+3                              *)
(*      d) G is nonplnar                       *)
(*                                              *)
(************************************************)
begin
  if (activerule[351]) and (max[girth] >= 5) and
     (max[nodes] < infinity) and (min[girth] < infinity) then
    begin
      rule:='351/ ';
```

```
z:=max[nodes];
if min[edges] >= z+3 then
      begin
        z:=z div 2;
        if z < 8 then z:=8;
        if max[girth]=7 then z:=6;
        if z < max[girth] then pushmax(girth);
      end;
z:=min[girth];
if (z >= 5) and (2*z-1 >= max[nodes]) and
   (min[edges] >= max[nodes]+3) then
    begin
      if min[girth]=7 then
           begin
             z:=8;
             pushmin(girth);
           end;
      z:=2*max[girth]-1;
      if z < max[nodes] then pushmax(nodes)
               else
                if z > max[nodes] then
                   begin
                     z:=(max[nodes]+1) div 2;
                     pushmax(girth);
                   end;
      z:=2*min[girth]-1;
      if z > min[nodes] then pushmin(nodes)
               else
                 if z < min[nodes] then
                    begin
                      z:=(min[nodes]+2) div 2;
                      pushmin(girth);
                    end;
      z:=2;
      if z < max[mindeg] then pushmax(mindeg);
      if z > min[nconn] then pushmin(nconn);
      z:=max[nodes]+3;
      if z < max[edges] then pushmax(edges)
          else
            if z > max[edges] then
              begin
                z:=max[edges]-3;
                pushmax(nodes);
              end;
      z:=min[nodes]+3;
      if z > min[edges] then pushmin(edges)
          else
            if z < min[edges] then
                begin
                  z:=min[edges]-3;
                  pushmin(nodes);
                end;
            z:=0;
            if z > max[plnar] then pushmax(plnar);
          end
         else
           if (min[girth] > 8) or ((min[girth] = 7)
              and (max[girth] = 7)) or (max[nodes] < 9) or
(min[nodes] > 15)
              or (max[nodes] < 2*min[girth]-1) or (min[nodes] > 2
*max[girth]-1)
              or (min[mindeg] > 2) or (max[nconn] < 2) or
(min[plnar]=1)
              or (min[edges] > max[nodes]+3) or (max[edges]
< min[nodes]+3)
            then
              if (min[girth] >= 5) and (2*min[girth] >=
max[nodes]+1) then
                   begin
                     z:=max[nodes]+2;
                     if z < max[edges] then pushmax(edges);
                     z:=min[edges]-2;
                     if z > min[nodes] then pushmin(nodes);
                   end
                 else
                   if (min[edges] >= max[nodes]+3) and
(min[girth] >= 5) then
                    begin
                      z:=2*min[girth];
                      if z > min[nodes] then pushmin(nodes);
                     end
                    else
                      if (2*min[girth] >= max[nodes]+1) and
                         (min[edges] >= max[nodes]+3) then
                           begin
                             z:=4;
                             if z < max[girth] then
pushmax(girth);
                           end;
    end;
end;

procedure r352;
(************************************************)
(*                                              *)
(* let t=(g-1) div 2, s=t-1 if mindeg <= 2      *)
(*                    s=(k**(t-1)-1)/(k-1)      *)
(*                       k=mindeg-1 >= 2        *)
(*      then                                    *)
(*      e <= (p(.5+sqrt((p-mindeg-1)/s+.25))/2  *)
(*                                              *)
(************************************************)
var s,t:longint;
begin
```

```
  if (activerule[352]) and (min[girth] >= 5) and (min[girth]
< infinity) then
    begin
      rule:='352/ ';
      t:=(min[girth]-1) div 2;
      if min[mindeg] <= 2 then s:=t-1
              else
                begin
                  power(min[mindeg]-1,t-1,z);
                  if z < infinity then s:=(z-1) div
(min[mindeg]-2)
                                  else s:=0;
                end;
      z:=max[nodes];
      if (z < infinity) and (s > 0) then
         begin
           rz:=0.5+sqrt((max[nodes]-min[mindeg]-1)/s+0.25);
           z:=trunk(z*rz/2);
           if z < max[edges] then pushmax(edges);
           rz:=2*min[edges]/max[nodes];
           z:=round((rz*rz-rz)*s+min[mindeg]+1+hf);
           if z > min[nodes] then pushmin(nodes);
         end;
    end;
end;

procedure r353;
(************************************************)
(*                                              *)
(*  let t = (g-1) div 2, k = e div p, and       *)
(*      R = t-1 if k <= 1 and                   *)
(*      R = (k**(t-1)-1)/(k-1) if k >= 2        *)
(*    then, if t >= 2,                          *)
(*        e <= (p*(.5+sqrt((p-k-2)/R+.25)))/2   *)
(*                                              *)
(************************************************)
var t,k,r:longint;
begin
  if (activerule[353]) and (min[girth] >= 5) and
       (max[nodes] < infinity) and (min[girth] < infinity) then
    begin
      rule:='353/ ';
      t:=(min[girth]-1) div 2;
      k:=min[edges] div max[nodes];
      if k <= 1 then r:=t-1
              else
                begin
                  power(k,t-1,z);
                  if z < infinity then r:=(z-1) div (k-1)
                                  else r:=0;
                end;
      if r > 0 then
```

```
               begin
                 rz:=0.5+sqrt((max[nodes]-k-2)/r+0.25);
                 z:=trunk(max[nodes]*rz/2);
                 if z < max[edges] then pushmax(edges);
                 rz:=2*min[edges]/max[nodes];
                 z:=round((rz*rz-rz)*r+k+2+hf);
                 if z > min[nodes] then pushmin(nodes);
               end;
    end;
end;

procedure r354;
(************************************************)
(*                                              *)
(* if not a forest then                         *)
(*  let k=(p-(g-1) div 2) div g   then          *)
(*      e <= p+k(2p-g(k+1))/(2*(g-1) div 4)     *)
(*                                              *)
(************************************************)
begin
  if (activerule[354]) and (min[girth] >= 5) and (min[forest] =
0) and
     (min[girth] < infinity) and (max[nodes] < infinity) then
    begin
      rule:='354/ ';
      z1:=(max[nodes]-(min[girth]-1) div 2) div min[girth];
      z:=z1*(2*max[nodes]-min[girth]*(z1+1));
      z:=max[nodes]+z div (2*((min[girth]-1) div 4));
      if max[forest] = 0 then
          begin
            if z < max[edges] then pushmax(edges);
          end
       else
         if z < min[edges] then
            begin
              z:=1;
              pushmin(forest);
            end;
    end;
end;

procedure r355;
(************************************************)
(*                                              *)
(* let t=g div 2, and                           *)
(*               s=1,t,((min-1)**t-1)/(min-2)   *)
(*      when min = 1,2,>=3,resp.                *)
(*   then p >= 1+max*s-(max-1)*(min-1)**(t-1)*k *)
(*       where k is 0 or 1 if g is odd or even  *)
(*                                              *)
(************************************************)
var m,s,t:longint;
```

```
function boundm(gi,mi:longint):longint;
begin
  t:= gi div 2;
  s:=infinity;
  boundm:=0;
  if (mi = 2) or (t = 1) then z1:=1
                         else z1:=0;
  if mi = 1 then s:=1
    else if mi = 2 then s:=t
      else
       begin
         power(mi-1,t-1,z1);
         if z1 < infinity then s:=(z1*(mi-1)-1) div (mi-2);
       end;
   if s < infinity then
      if odd(gi) then boundm:=1+min[maxdeg]*s
                 else boundm:=1+min[maxdeg]*s-
((min[maxdeg]-1)*z1);
 end;

begin
  if (activerule[355]) and (min[girth] < infinity) then
    begin
      rule:='355/ ';
      t:=min[girth] div 2;
      z:=min[mindeg];
      if z=1 then
         begin
           s:=1;
           z1:=0;
         end
        else
         if z = 2 then
             begin
               s:=t;
               z1:=1;
             end
         else
           begin
             power(z-1,t-1,z1);
             if z1 < infinity then s:=(z1*(min[mindeg]-1)-1) div
(z-2)
                              else s:=0;
           end;
      if s > 0 then
            if odd(min[girth]) then
               begin
                 z:=1+min[maxdeg]*s;
                 if z > min[nodes] then pushmin(nodes);
                 if max[nodes] < infinity then
                     begin
                       z:=(max[nodes]-1) div s;
                       if z < max[maxdeg] then pushmax(maxdeg);
                     end;
               end
             else
               begin
                 z:=1+min[maxdeg]*s-(min[maxdeg]-1)*z1;
                 if z > min[nodes] then pushmin(nodes);
                 if (max[nodes] < infinity) and (s > z1) then
                     begin
                       z:=(max[nodes]-1-z1) div (s-z1);
                       if z < max[maxdeg] then pushmax(maxdeg);
                     end;
                end;
      if max[nodes] < infinity then
            begin
              z:=max[girth];
              if (z < infinity) and (min[mindeg] >= 2) then
                begin
                  m:=boundm(z,min[mindeg]);
                  if m < infinity then
                    begin
                      while (m > max[nodes]) and (z >=
min[girth]) do
                         begin
                           z:=z-1;
                           m:=boundm(z,min[mindeg]);
                         end;
                      if z < max[girth] then pushmax(girth);
                    end;
                end;
              z:=max[mindeg];
              if z < infinity then
                begin
                  m:=boundm(min[girth],z);
                  if m < infinity then
                    begin
                      while (m > max[nodes]) and (z >=
min[mindeg]) do
                         begin
                           z:=z-1;
                           m:=boundm(min[girth],z);
                         end;
                      if z < max[mindeg] then pushmax(mindeg);
                    end;
                 end;
             end;
    end;
end;

procedure r356;
(**************************************************)
(*                                                *)
(*  if regular and eind < p/2 then echr=maxdeg+1 *)
(*                                                *)
(**************************************************)
begin
  if (activerule[356]) and (max[reg] = 1) then
    begin
      rule:='356/ ';
      if (min[reg] = 1) and (max[eind] < min[nodes]/2) then
         begin
           z:=min[maxdeg]+1;
           if z > min[echr] then pushmin(echr);
           if max[echr] < infinity then
             begin
               z:=max[echr]-1;
               if z < max[maxdeg] then pushmax(maxdeg);
             end;
         end
       else
         if (min[maxdeg]=max[echr]) and (min[reg] = 1) then
            begin
              z:=(min[nodes]+1) div 2;
              if z > min[eind] then pushmin(eind);
              if max[eind] < infinity then
                 begin
                   z:=2*max[eind];
                   if z < max[nodes] then pushmax(nodes);
                 end;
             end
         else
           if (2*max[eind] < min[nodes]) and (min[maxdeg] =
max[echr]) then
             begin
               z:=0;
               pushmax(reg);
             end;
    end;
end;

procedure r357;
(***************************************************)
(*                                                 *)
(* if regular then eind >= p*maxdeg/(2(maxdeg+1)) *)
(*                                                 *)
(***************************************************)
begin
  if (activerule[357]) and (max[reg] = 1) then
    begin
      rule:='357/ ';
      z:= (min[nodes]*min[maxdeg]-1) div (2*min[maxdeg]+2)+1;
      if min[reg] = 1 then
         begin
           if z > min[eind] then pushmin(eind);
           if max[eind] < infinity then
              begin
                z:=(2*max[eind]*(min[maxdeg]+1)) div min[maxdeg];
                if z < max[nodes] then pushmax(nodes);
                z:=min[nodes]-2*max[eind];
                if z > 0 then
                   begin
                     z:= (2*max[eind]) div z;
                     if z < max[maxdeg] then pushmax(maxdeg);
                   end;
              end;
          end
         else
           if max[eind] < z then
              begin
                z:=0;
                pushmax(reg);
              end;
    end;
end;

procedure r358;
(********************************************************)
(*                                                      *)
(*  nconn>= 2 and mindeg>= (p+nconn)/3 then Hamiltonian *)
(*                                                      *)
(********************************************************)
begin
  if (activerule[358]) and (min[hamil] = 0) and (max[nconn] >= 2)
then
    begin
      rule:='358/ ';
      if (min[nconn] >= 2) and (3*min[mindeg] >=
max[nodes]+max[nconn]) then
           begin
             z:=1;
             pushmin(hamil);
           end
       else
        if (max[hamil] = 0) and (min[nconn] >= 2) then
            begin
              if (max[nodes] < infinity) and (max[nconn]
< infinity) then
                 begin
                   z:=(max[nodes]+max[nconn]-1) div 3;
                   if z < max[mindeg] then pushmax(mindeg);
                  end;
              if max[nodes] < infinity then
                   begin
                     z:=3*min[mindeg]+1-max[nodes];
                     if z > min[nconn] then pushmin(nconn);
                   end;
              if max[nconn] < infinity then
                   begin
                     z:=3*min[mindeg]+1-max[nconn];
                     if z > min[nodes] then pushmin(nodes);
                   end;
            end
          else
            if (3*min[mindeg] >= max[nodes]+max[nconn]) and
(max[hamil] = 0)
               then
                 begin
                   z:=1;
                   pushmax(nconn);
                 end;
    end;
end;

procedure r359;
(************************************************)
(*                                              *)
(*  if nconn >= 3 then                          *)
(*          cir >= MIN[p,3*mindeg-nconn}        *)
(*                                              *)
(************************************************)
begin
  if (activerule[359]) and (max[nconn] >= 3) and (min[hamil] =0)
then
    begin
      rule:='359/ ';
      z:= 3*min[mindeg]-max[nconn];
      if z > min[nodes] then z:=min[nodes];
      if min[nconn] >= 3 then
         begin
           if z > min[circ] then pushmin(circ);
           if max[circ] < 3*min[mindeg]-max[nconn] then
                begin
                  z:=1;
                  pushmin(hamil);
                end
              else
                if (max[circ] < min[nodes]) or (max[hamil] =0)
then
                    begin
                      if max[nconn] < infinity then
                        begin
                          z:=(max[circ]+max[nconn]) div 3;
                          if z < max[mindeg] then
pushmax(mindeg);
                        end;
                      z:=3*min[mindeg]-max[circ];
                      if z > min[nconn] then pushmin(nconn);
                    end;
          end
        else if max[circ] < z then
                          begin
                            z:=2;
                            pushmax(nconn);
                          end;
    end;
end;

procedure r360;
(*****************************************************)
(*                                                   *)
(* if reg and p = 2*maxdeg-1 then nconn >= nind      *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[360]) and (max[reg] = 1) then
    begin
      rule:='360/ ';
      z:=max[nodes];
      if (min[reg] = 1) and (z = min[nodes]) and (max[maxdeg] =
min[maxdeg])
         and (z = 2*max[maxdeg]+1) then
         begin
           z:=min[nind];
           if z > min[nconn] then pushmin(nconn);
           z:=max[nconn];
           if z < max[nind] then pushmax(nind);
         end
         else
           if (max[nconn] < min[nind]) and (z = min[nodes]) and
              (max[maxdeg] = min[maxdeg]) and (z = 2*max[maxdeg]+
1) then
               begin
                 z:=0;
                 pushmax(reg);
               end;
    end;
end;

procedure r361;
(*****************************************************)
(*                                                   *)
(* if t=(gi-2) div 2 >= 1 and mindeg=2 then          *)
(*     nind >= maxdeg*((t+1) div 2))  {+1 if t even} *)
(*                                                   *)
(*****************************************************)
var t:longint;
begin
  if (activerule[361]) and (max[mindeg] >= 2)
      and (min[mindeg] <= 2) and (min[girth] >= 4)then
    begin
```

```
      rule:='361/ ';
      t:=(min[girth]-2) div 2;
      if odd(t) then z1:=0
                else z1:=1;
      z:=min[maxdeg]*((t+1-z1) div 2)+z1;
      if min[mindeg]=max[mindeg] then
           begin
             if z > min[nind] then pushmin(nind);
             if max[nind] < infinity then
               begin
                 z:=(2*(max[nind]-z1)) div (t+1-z1);
                 if z < max[maxdeg] then pushmax(maxdeg);
                 z:=4*((max[nind]-z1) div min[maxdeg])+2*z1+1;
                 if z < max[girth] then pushmax(girth);
               end;
            end
          else
            if max[nind] < z then
               begin
                 z:=1;
                 pushmax(mindeg);
               end;
    end;
end;

procedure r362;
(***********************************************************)
(*                                                         *)
(* if girth >= 4 and mindeg >= 3 then                      *)
(*     nind >= maxdeg*(mindeg-1)*s+k                       *)
(*                                                         *)
(*      where t = (girth-2) div 2 and                      *)
(*         s = ((mindeg-1)**t-1) div (mindeg*(mindeg-2))   *)
(*         and k = 1 if t even, else = maxdeg/mindeg       *)
(*                                                         *)
(***********************************************************)
var t:longint;
begin
  if (activerule[362]) and (min[girth] >= 4) and (min[girth]
< infinity)
     and (max[mindeg] >= 3) then
    begin
      rule:='362/ ';
      t:=(min[girth]-2) div 2;
      if (max[nind] < infinity) and (t >= 2) then
          begin
            if t = 2 then z:=1+max[nind] div min[maxdeg]
               else z:=1+trunk(root(max[nind]/min[maxdeg],t-1));
            if z < 2 then z:=2;
            if z < max[mindeg] then pushmax(mindeg);
          end;
      z:=min[mindeg];
      if z >= 3 then
        begin
          if t > 1 then
           begin
             power(z-1,t,z1);
             z1:=(z-1)*((z1-1) div (z*(z-2)));
           end;
          if odd(t) then
            begin
              if t = 1 then z:=min[maxdeg]
                 else z:=min[maxdeg]*z1+1+(min[maxdeg]-1) div z;
              if z > min[nind] then pushmin(nind);
              if max[nind] < infinity then
                 begin
                   if t = 1 then z:=max[nind]
                     else z:=(min[mindeg]*max[nind]) div (z1
*min[mindeg]+1);
                   if z < max[maxdeg] then pushmax(maxdeg);
                 end;
             end
           else
             begin
               z:=min[maxdeg]*z1+1;
               if z > min[nind] then pushmin(nind);
               if max[nind] < infinity then
                  begin
                    z:=(max[nind]-1) div z1;
                    if z < max[maxdeg] then pushmax(maxdeg);
                  end;
              end;
         end;
    end;
end;

procedure r363;
(************************************************)
(*                                              *)
(* if diam=2 and k(=nconn) >= 3 then            *)
(*    e >= (p-1)*(k+1)/2-k**2+2*k               *)
(*                                              *)
(************************************************)
begin
  if (activerule[363]) and (min[diam] <= 2) and (max[diam] >= 2)
     and (max[nconn] >= 3) then
    begin
      rule:='363/ ';
      z:=min[nconn];
      z:=((min[nodes]-1)*(z+1)+1) div 2-z*z+2*z;
      z1:=max[nconn];
      z1:=((min[nodes]-1)*(z1+1)+1) div 2-z1*z1+2*z1;
      if z1 < z then z:=z1;
      if (min[diam]=2) and (max[diam]=2) and (min[nconn] >= 3)
```

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

procedure r364;
(************************************************)
(*                                              *)
(*  if diam=2, k(=nconn) >= 3 and p >= 3*k**2+6 *)
(*      then e >= (p-1)*(k+1)/2                 *)
(*                                              *)
(************************************************)
begin
  if (activerule[364]) and (min[diam] <= 2) and (max[diam] >= 2)
     and (max[nconn] >= 3) then
    begin
      rule:='364/ ';
      if (min[diam] = max[diam]) and (max[nconn] < infinity) then
         begin
           if (min[nodes] >= 3*max[nconn]*max[nconn]+6)
              and (max[edges] < infinity) then
               begin
                 z:=(2*max[edges]) div (min[nodes]-1)-1;
                 if z < 2 then z:=2;
                 if z < max[nconn] then pushmax(nconn);
               end;
             z:=max[nconn];
             if (min[nconn] >= 3) and (max[edges] < infinity)
```

```
then
                  begin
                    z:=3*z*z+5;
                    z1:=(2*max[edges]) div (min[nconn]+1)+1;
                    if z < z1 then z:=z1;
                    if z < max[nodes] then pushmax(nodes);
                  end;
            end;
      z1:=min[nconn];
      z:=((min[nodes]-1)*(z1+1)+1) div 2;
      if (min[diam] = max[diam]) and (min[nconn] >= 3) and
         (min[nodes] >= 3*max[nconn]*max[nconn]+6) then
         begin
           if z > min[edges] then pushmin(edges);
          end
        else
          if (z > max[edges]) and (min[diam] = max[diam])
             and (min[nconn] >= 3) then
             begin
               z:= round(sqrt((min[nodes]-5)/3.0)+hf);
               if z > min[nconn] then pushmin(nconn);
             end
         else
           if (z > max[edges]) and (min[nodes] >= 3
*max[nconn]*max[nconn]+6) and
               (min[nconn] >= 3) then
              begin
                z:=3;
                pushmin(diam)
               end;
    end;
end;

procedure r365;
(************************************************)
(*                                              *)
(*   if bipartite then   xnum <=                *)
(*     (p/4)**2*((p-2)/4)**2           , p even *)
(*     ((p+1)/4)*((p-1)/4)**2*((p-3)/4) , p odd *)
(*                                              *)
(************************************************)
begin
  if (activerule[365]) and (max[bipart] = 1) then
    begin
      rule:='365/ ';
      z1:=max[nodes];
      if z1 < infinity then
         if odd(z1) then
            begin
              rz:=((z1+1) div 4)*((z1-3) div 4);
              z1:=(z1-1) div 4;
              rz:=rz*z1*z1;
            end
          else
            begin
              rz:=z1 div 4;
              z1:=(z1-2) div 4;
              rz:=rz*rz*z1*z1;
            end;
      if rz < infinity then
        begin
          z:=trunk(rz);
          if min[bipart]=1 then
            begin
              z1:=max[nodes];
              if z1 < infinity then
                 if z < max[xnum] then pushmax(xnum);
              z:=round(4*sqrt(sqrt(min[xnum]))+hf);
              if z > min[nodes] then pushmin(nodes);
            end
          else
            if max[nodes] < infinity then
               if min[xnum] > z then
                   begin
                     z:=0;
                     pushmax(bipart);
                   end;
        end;
    end;
end;

procedure r366;
(*****************************************************)
(*                                                   *)
(*  if connct then spectr >= 2*cos(3.1415927/(p+1))  *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[366]) and (max[connct] = 1) then
    begin
      rule:='366/ ';
      rz:=2*cos(3.1415927/(min[nodes]+1));
      z:=0;
      if min[connct]=1 then
        begin
          if rz > lammin then pushlammin;
        end
       else
         if lammax < rz then pushmax(connct);
    end;
end;

procedure r367;
(*****************************************************)
```

```
(*                                                   *)
(* if reg, mindeg >= 7(<>9) and odd, not bipartite,  *)
(*    and girth=4 then p >=2*fL((5*mindeg)/4)+4      *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[367]) and (max[reg] = 1) and (min[bipart] = 0)
and
     (min[girth] <= 4) and (max[girth] >= 4) then
    begin
      rule:='367/ ';
      z:=2*((5*min[mindeg]) div 4)+4;
      if min[mindeg]=9 then z:=z-2
        else
          if (min[mindeg]-4*(min[mindeg] div 4)=1)
             and (min[mindeg] < max[mindeg]) then z:=z-1;
      if (min[reg]=1) and (min[mindeg] >= 7) and
(odd(min[mindeg])) and
         (max[bipart]=0) and (min[girth]=max[girth]) and
(min[girth]=4)
          then
            begin
              if z > min[nodes] then pushmin(nodes);
              if max[nodes] < infinity then
                   begin
                     if min[mindeg]=9 then z:=max[nodes]-2
                       else
                         if (min[mindeg]-4*(min[mindeg] div 4)=1)
and
                            (min[mindeg] < max[mindeg]) then z:
=max[nodes]-3
                         else  z:=max[nodes]-4;
                     z:=(4*(z div 2)+3) div 5;
                     if z < max[mindeg] then pushmax(mindeg);
                   end;
            end
          else
            if (max[nodes] < z) and (min[reg]=1) and
(min[mindeg] >= 7) and
               (odd(min[mindeg])) and (max[bipart]=0) then
                begin
                  if max[girth] <= 4 then
                      begin
                        z:=3;
                        pushmax(girth);
                      end
                    else if min[girth] >= 4 then
                            begin
                              z:=5;
                              pushmin(girth);
                            end;
                end
              else
                if (min[girth]=max[girth]) and (min[girth]=4) and
                   (max[nodes] < z) and (min[reg]=1) and
(min[mindeg] >= 7)
                   and (odd(min[mindeg])) then
                    begin
                      z:=1;
                      pushmin(bipart);
                    end
               else
                 if (max[bipart]=0) and (min[girth]=max[girth])
and
                    (min[girth]=4) and (max[nodes] < z) and
(min[reg]=1)
                    and (min[mindeg] >= 7) then
                     begin
                       z:=min[mindeg];
                       if odd(z) then
                            begin
                              z:=z+1;
                              pushmin(mindeg);
                            end;
                       z:=max[mindeg];
                       if odd(z) then
                              begin
                                z:=z-1;
                                pushmax(mindeg);
                              end;
                     end
                else
                  if (min[mindeg] >= 7) and (odd(min[mindeg]))
and
                     (max[bipart]=0) and (min[girth]=max[girth])
and
                     (min[girth]=4) and (max[nodes] < z) then
                        begin
                          z:=0;
                          pushmax(reg);
                        end;
    end;
end;

procedure r368;
(************************************************)
(*                                              *)
(*   if bipartite then thick <= nodes/8 + 2     *)
(*                                              *)
(************************************************)
begin
  if (activerule[368]) and (max[bipart] = 1) then
    begin
      rule:='368/ ';
      if min[bipart] = 1 then
           begin
             z:=min[thick]*8-16;
             if z > min[nodes] then pushmin(nodes);
           end;
      if max[nodes] < infinity then
         begin
           z:=0;
           if min[bipart] = 1 then
                  begin
                    z:=max[nodes] div 8 + 2;
                    if z < max[thick] then pushmax(thick);
                  end
                else
                  if min[thick] > max[nodes] div 8 + 2 then
pushmax(bipart);
          end;
    end;
end;

procedure r369;
(************************************************)
(*                                              *)
(*  if clique <= 2 then thick <= genus+1        *)
(*                                              *)
(************************************************)
begin
  if (activerule[369]) and (min[clique] <= 2) then
    begin
      rule:='369/ ';
      if max[clique] <= 2 then
               begin
                 z:=min[thick]-1;
                 if z > min[genus] then pushmin(genus);
               end;
      if max[genus] < infinity then
         begin
           z:=3;
           if max[clique] <= 2 then
              begin
                z:=max[genus]+1;
                if z < max[thick] then pushmax(thick);
              end
            else
              if min[thick] > max[genus]+1 then pushmin(clique);
         end;
    end;
end;

procedure r370;
(************************************************)
(*                                              *)
(*  if genus <= 1 then thick = genus+1          *)
(*                                              *)
(************************************************)
begin
  if (activerule[370]) and (min[genus] <= 1) then
    begin
      rule:='370/ ';
      z:=2;
      if max[genus] <= 1 then
          begin
            z:=max[genus]+1;
            if z < max[thick] then pushmax(thick);
            z:=min[genus]+1;
            if z > min[thick] then pushmin(thick);
            z:=max[thick]-1;
            if z < max[genus] then pushmax(genus);
            z:=min[thick]-1;
            if z > min[genus] then pushmin(genus);
          end
         else
          if min[thick] >= 3  then pushmin(genus);
    end;
end;

procedure r371;
(**************************************)
(*                                    *)
(*    arbor <= earbor                 *)
(*                                    *)
(**************************************)
begin
  if activerule[371] then
    begin
      rule:='371/ ';
      z:=max[earbor];
      if z < max[arbor] then pushmax(arbor);
      z:=min[arbor];
      if z > min[earbor] then pushmin(earbor);
    end;
end;

procedure r372;
(*****************************)
(*                           *)
(*    thick <= earbor        *)
(*                           *)
(*****************************)
begin
  if activerule[372] then
    begin
      rule:='372/ ';
      z:=max[earbor];
      if z < max[thick] then pushmax(thick);
      z:=min[thick];
      if z > min[earbor] then pushmin(earbor);
    end;
end;

procedure r373;
(************************************************)
(*                                              *)
(*  if genus >= 1 then                          *)
(*    earbor <= 2 + CL((3*genus)**.5)           *)
(*                                              *)
(************************************************)
begin
  if (activerule[373]) and (max[genus] >= 1) then
    begin
      rule:='373/ ';
      if max[genus] < infinity then
        begin
          z:=2+round(sqrt(3*max[genus])+hf);
          if z < max[earbor] then pushmax(earbor);
        end;
      z:=min[earbor]-3;
      if (z > 0) and (min[genus] > 0) then
        begin
          z:=(z*z+2) div 3;
          if z > min[genus] then pushmin(genus);
        end;
    end;
end;

procedure r374;
(************************************************)
(*                                              *)
(*   thick <= 5+(2genus-2)**.5                  *)
(*                                              *)
(************************************************)
begin
  if activerule[374] then
    begin
      rule:='374/ ';
      z:=max[genus];
      if (z < infinity) and (z > 0) then
        begin
          z:=5+trunk(sqrt(2*z-2));
          if z < max[thick] then pushmax(thick);
        end;
      z:=min[thick]-5;
      if z >= 0 then
         begin
           z:=(z*z+3) div 2;
           if z > min[genus] then pushmin(genus);
         end;
    end;
end;

procedure r375;
(************************************************)
(*                                              *)
(*    if connected and regular then             *)
(*   earbor <= 4 + (6genus+2) div (p-1)         *)
(*                                              *)
(************************************************)
begin
  if (activerule[375]) and (max[connct] = 1) and
     (max[reg] = 1) then
    begin
      rule:='375/ ';
      if (min[connct]=1) and (min[reg]=1) then
         begin
           if max[genus] < infinity then
              begin
                z:=4+(6*max[genus]+2) div (min[nodes]-1);
                if z < max[earbor] then pushmax(earbor);
                if min[earbor] > 4 then
                    begin
                      z:=(6*max[genus]+2) div (min[earbor]-4)+1;
                      if z > max[nodes] then pushmax(nodes);
                    end;
              end;
           z:=((min[earbor]-4)*(min[nodes]-1)+3) div 6;
           if z > min[genus] then pushmin(genus);
         end
        else
          if max[genus] < infinity then
             begin
               z:=4+(6*max[genus]+2) div (min[nodes]-1);
               if (min[earbor] > z) and (min[reg]=1) then
                    begin
                      z:=0;
                      pushmax(connct);
                     end
                   else
                     if (min[earbor] > z) and (min[connct] = 1)
then
                        begin
                          z:=0;
                          pushmax(reg);
                        end;
              end;
     end;
end;

procedure r376;
```

```
(************************************************)
(*                                              *)
(*    genus <= (thick-1)*(p-1)                  *)
(*                                              *)
(************************************************)
begin
  if (activerule[376]) and (max[thick] < infinity) and
      (max[nodes] < infinity) then
    begin
      rule:='376/ ';
      z:=(max[thick]-1)*(max[nodes]-1);
      if z < max[genus] then pushmax(genus);
      z:=(min[genus]+max[nodes]-2) div (max[nodes]-1)+1;
      if z > min[thick] then pushmin(thick);
      if max[thick] > 1 then
         begin
            z:=(min[genus]-1) div (max[thick]-1)+2;
           if z > min[nodes] then pushmin(nodes);
         end;
    end;
end;

procedure r377;
(***************************************)
(*                                     *)
(*    earbor <= (maxdeg+2)/2           *)
(*                                     *)
(***************************************)
begin
  if activerule[377] then
    begin
      rule:='377/ ';
      if max[maxdeg] < infinity then
        begin
          z:=(max[maxdeg]+2) div 2;
          if z < max[earbor] then pushmax(earbor);
        end;
      z:=2*min[earbor]-2;
      if z > min[maxdeg] then pushmin(maxdeg);
    end;
end;

procedure r378;
(***********************************)
(*                                 *)
(*    earbor >= (mindeg+1)/2       *)
(*                                 *)
(***********************************)
begin
  if activerule[378] then
    begin
      rule:='378/ ';
      z:=(min[mindeg]+2) div 2;
      if z > min[earbor] then pushmin(earbor);
      if max[earbor] < infinity then
          begin
            z:=2*max[earbor]-1;
            if z < max[mindeg] then pushmax(mindeg);
          end;
    end;
end;

procedure r379;
(*************************************)
(*                                   *)
(*   earbor >= CL(edges/(p-ncomp))   *)
(*                                   *)
(*************************************)
begin
  if activerule[379] then
    begin
      rule:='379/ ';
      if max[nodes] < infinity then
           begin
             z:=(min[edges]-1) div (max[nodes]-min[ncomp])+1;
             if z > min[earbor] then pushmin(earbor);
             if max[earbor] < infinity then
                 begin
                   z:=max[earbor]*(max[nodes]-min[ncomp]);
                   if z < max[edges] then pushmax(edges);
                   z:=max[nodes]-1-((min[edges]-1) div
max[earbor]);
                   if z < max[ncomp] then pushmax(ncomp);
                 end;
           end;
       if max[earbor] < infinity then
           begin
             z:=(min[edges]-1) div max[earbor]+min[ncomp]+1;
             if z > min[nodes] then pushmin(nodes);
           end;
    end;
end;

procedure r380;
(***************************)
(*                         *)
(*   earbor <= 3*thick     *)
(*                         *)
(***************************)
begin
  if activerule[380] then
    begin
      rule:='380/ ';
      if max[thick] < infinity then
```

```
          begin
            z:=3*max[thick];
            if z < max[earbor] then pushmax(earbor);
          end;
      z:=(min[earbor]+2) div 3;
      if z > min[thick] then pushmin(thick);
    end;
end;

procedure r381;
(************************************************)
(*                                              *)
(*    if plnar and p >= 4 then                 *)
(*         edges <= 3*p-9+min{3,econn}          *)
(*                                              *)
(************************************************)
begin
  if (activerule[381]) and (max[nodes] >= 4) and (max[plnar] = 1)
then
    begin
      rule:='381/ ';
      if max[econn] > 2 then z:=3
                        else z:=max[econn];
      if min[plnar] = 1 then
        begin
          if max[nodes] < infinity then
             begin
               z:=3*max[nodes]-9+z;
               if z < max[edges] then pushmax(edges);
               z:=min[edges]-3*max[nodes]+9;
               if z > min[econn] then pushmin(econn);
              end;
           if min[nodes] >= 4 then
              begin
                if max[econn] > 2 then z:=3
                                  else z:=max[econn];
                z:=(min[edges]+11-z) div 3;
                if z > min[nodes] then pushmin(nodes);
              end;
         end
       else
         if max[nodes] < infinity then
             if (min[edges] > 3*max[nodes]-9+z) and
(min[nodes] >= 4) then
                  begin
                    z:=0;
                    pushmax(plnar);
                  end;
    end;
end;

procedure r382;
```

```
(************************************************)
(*                                              *)
(*  if plnar and econn < mindeg  and           *)
(*     (p >= 5 or mindeg >= 2) then             *)
(*     edges <= 3*p-11   when mindeg=econn+1=1  *)
(*           <= 3*p-12+econn   otherwise        *)
(*                                              *)
(************************************************)
begin
  if (activerule[382]) and (max[plnar] = 1) then
    begin
      rule:='382/ ';
      if (min[plnar] = 1) and (max[econn] < min[mindeg]) and
         ((min[nodes] >= 5) or (min[mindeg] >= 2)) then
           begin
             if (min[econn] = 0) and (min[mindeg] = 1) then
                 begin
                   if max[nodes] < infinity then
                       begin
                         z:=3*max[nodes]-11;
                         if z < max[edges] then pushmax(edges);
                       end;
                    z:=(min[edges]+13) div 3;
                    if z > min[nodes] then pushmin(nodes);
                  end
                 else
                   begin
                     if max[nodes] < infinity then
                          begin
                            z:=3*max[nodes]-12+max[econn];
                            if z < max[edges] then
pushmax(edges);
                            z:=min[edges]-3*max[nodes]+12;
                            if z > min[econn] then
pushmin(econn);
                          end;
                      z:=(min[edges]+14-max[econn]) div 3;
                      if z > min[nodes] then pushmin(nodes);
                    end;
            end
          else
            if max[nodes] < infinity then
               begin
                 if max[econn] <= 1 then z:=3*max[nodes]-11
                                    else z:=3*max[nodes]-12
+max[econn];
                 if (min[plnar] = 1) and (min[edges] > z) and
                    ((min[nodes] >= 5) or (min[mindeg] >= 2))
then
                     begin
                       z:=min[mindeg];
                       if z > min[econn] then pushmin(econn);
                       z:=max[econn];
                       if z < max[mindeg] then pushmax(mindeg);
                     end
                   else
                     if (max[econn] < min[mindeg]) and
(min[edges] > z) and
                        ((max[nodes] >= 5) or (min[mindeg] >= 2))
then
                        begin
                          z:=0;
                          pushmax(plnar);
                        end
                    else
                      if (min[plnar] = 1) and (max[econn]
< min[mindeg])
                         and (min[edges] > z) then
                         begin
                           z:=4;
                           if z < max[nodes] then pushmax(nodes);
                           z:=1;
                           if z < max[mindeg] then
pushmax(mindeg);
                         end;
                 end;
    end;
end;

procedure r383;
(************************************************)
(*                                              *)
(*  if not a forest then                        *)
(*     p >= maxdeg+Nc-2+(cir*(gi-3)+2)/(gi-2)   *)
(*                                              *)
(************************************************)
begin
  if (activerule[383]) and (min[forest]=0) then
    begin
      rule:='383/ ';
      z:=min[girth]-2;
      z:=min[maxdeg]+min[ncomp]-2+((min[circ]+1)*(z-1)+2) div z;
      if max[forest]=0 then
        begin
          if z > min[nodes] then pushmin(nodes);
          if max[nodes] < infinity then
              begin
                z:=max[nodes]-(z-min[maxdeg]);
                if z < max[maxdeg] then pushmax(maxdeg);
                z:=z+min[ncomp]-min[maxdeg];
                if z > max[ncomp] then pushmax(ncomp);
                if min[girth] >= 4 then
                  begin
                    z:=(max[nodes]-min[maxdeg]-min[ncomp]+
2)*(min[girth]-2);
                    z:=(z-2) div (min[girth]-3);
                    if z < max[circ] then pushmax(circ);
                  end;
                z:=min[circ]-2-max[nodes]+min[maxdeg]+min[ncomp];
                if z > 0 then
                    begin
                      z:=(min[circ]-2) div z+2;
                      if z < max[girth] then pushmax(girth);
                    end;
               end;
          end
         else
           if max[nodes] < z then
              begin
                z:=1;
                pushmin(forest);
              end;
    end;
end;

procedure r384;
(************************************************)
(*                                              *)
(*  ncov <= (2*nodes+edges-eind)/4              *)
(*                                              *)
(************************************************)
begin
  if (activerule[384]) and (max[nodes] < infinity)
     and (max[edges] < infinity) then
    begin
      rule:='384/ ';
      z1:=2*max[nodes]+max[edges]-min[eind];
      z:=z1 div 4;
      if z < max[ncov] then pushmax(ncov);
      z:=z1+min[eind]-4*min[ncov];
      if z < max[eind] then pushmax(eind);
      z:=4*min[ncov]+min[eind]-2*max[nodes];
      if z > min[edges] then pushmin(edges);
      z:=(z+2*max[nodes]-max[edges]+1) div 2;
      if z <  min[nodes] then pushmin(nodes);
    end;
end;

procedure r385;
(************************************************)
(*                                              *)
(* if genus <= (p*(sqrt(2*p)-7)/12+1 then       *)
(*         theta1 <= ncov*nind                  *)
(*                                              *)
(************************************************)
begin
```

```
  if (activerule[385]) and (max[nind] < infinity) and
     (max[ncov] < infinity) then
    begin
      rule:='385/ ';
      z:=min[nodes];
      rz:=(z*sqrt(2*z)-7)/12+1;
      rulef(ncov,nind);
      if max[genus] <= rz then
            begin
              if z < max[eccov] then pushmax(eccov);
              z:=(min[eccov]-1) div max[nind]+1;
              if z > min[ncov] then pushmin(ncov);
              z:=(min[eccov]-1) div max[ncov]+1;
              if z > min[nind] then pushmin(nind);
            end
          else
            if min[eccov] > z then
               begin
                 z:=trunk(rz)+1;
                 if z > min[genus] then pushmin(genus);
               end;
     end;
end;

procedure r386;
(************************************************)
(*                                              *)
(* if mindeg>=2 then  dom >= CL(girth/3)*ncomp  *)
(*                                              *)
(************************************************)
begin
  if (activerule[386]) and (min[girth] < infinity)
     and (max[mindeg] >= 2) then
    begin
      rule:='386/ ';
      z:=((min[girth]+2) div 3)*min[ncomp];
      if min[mindeg] >= 2 then
            begin
              if z > min[dom] then pushmin(dom);
              if max[dom] < infinity then
                begin
                  z:=max[dom] div ((min[girth]+2) div 3);
                  if z < max[ncomp] then pushmax(ncomp);
                  z:=(max[dom] div min[ncomp])*3;
                  if z < max[girth] then pushmax(girth);
                end;
            end
          else
            if max[dom] < z then
                 begin
                   z:=1;
                   pushmax(mindeg);
                 end;
    end;
end;

procedure r387;
(***********************************************************)
(*                                                         *)
(* if mindeg >= 2 and girth >= 5 then                      *)
(*    dom <=(p-FL(gi/3)-(gi-4)(mi-2)(mi-3)/2-2(mi-2)+1)/2  *)
(*                                                         *)
(***********************************************************)
begin
  if (activerule[387]) and (max[mindeg] >= 2) and (max[girth] >=
5)
     and (max[nodes] < infinity) then
    begin
      rule:='387/ ';
      z:=min[mindeg]-2;
      z1:=min[girth];
      z:=(max[nodes]-z1 div 3-((z1-4)*z*(z-1)) div 2-2*z+1) div
2;
      if (min[mindeg] >= 2) and (z1 >= 5) then
         begin
           if z < max[dom] then pushmax(dom);
           z:=min[mindeg]-2;
           z:=2*min[dom]+z1 div 3+((z1-4)*z*(z-1)) div 2+2*z-1;
           if z > min[nodes] then pushmin(nodes);
           z1:=min[mindeg]-2;
           z:=3*max[nodes]+5-6*min[dom];
           z:=z div ((3*z1*(z1-1)) div 2+1)+4;
           if z < max[girth] then pushmax(girth);
         end
       else
         if (min[dom] > z) and (min[mindeg] >= 2) then
            begin
              z:=4;
              pushmax(girth);
            end
           else
             if (min[girth] >= 5) and (min[dom] > z) then
               begin
                 z:=1;
                 pushmax(mindeg);
               end;
    end;
end;

procedure r388;
(************************************************)
(*                                              *)
(*  if mindeg >= 2 and girth >= 9 then          *)
(*   dom <= (p-FL(gi/3)-gi*(mi-2)*(mi-3)/2+1)/2 *)
(*                                              *)
(************************************************)
begin
  if (activerule[388]) and (max[mindeg] >= 2) and (max[girth] >=
9)
     and (max[nodes] < infinity) then
    begin
      rule:='388/ ';
      z:=3*max[nodes]-6*min[dom]+5;
      z1:=min[mindeg]-2;
      if z1 >= 0 then
        begin
          if z1 >= 1 then z:=(2*z) div (3*z1*(z1-1)+2);
          if z < 8 then z:=8;
          if z < max[girth] then pushmax(girth);
        end;
      z:=(max[nodes]-(min[girth] div 3)-min[girth]*z1*(z1-1) div
2+1) div 2;
      if (min[mindeg] >= 2) and (min[girth] >= 9) then
         begin
           if z < max[dom] then pushmax(dom);
           z:=2*min[dom]+(min[girth] div 3)+min[girth]*z1*(z1-1)
div 2-1;
           if z > min[nodes] then pushmin(nodes);
           z:=(2*max[nodes]-4*min[dom]-2*(min[girth] div 3)+2)
div min[girth];
           z:=4*z+1;
           if z >= 0 then
             begin
               z:=trunk((5+sqrt(z))/2);
               if z < max[mindeg] then pushmax(mindeg);
             end;
          end
         else
            if (min[girth] >= 9) and (min[dom] > z) then
              begin
                z:=1;
                pushmax(mindeg);
              end;
    end;
end;

procedure r389;
(************************************************)
(*                                              *)
(*  if maxdeg >= 6 and clique < maxdeg then     *)
(*          nodes <= maxdeg*nind-1              *)
(*                                              *)
(************************************************)
begin
  if (activerule[389]) and (max[maxdeg] >= 6)
     and (max[maxdeg] < infinity) and (max[nind] < infinity) then
    begin
      rule:='389/ ';
      z:=max[maxdeg]*max[nind]-1;
      if max[clique] < min[maxdeg] then
          begin
            if z < max[nodes] then pushmax(nodes);
            z:=min[nodes] div max[maxdeg]+1;
            if z > min[nind] then pushmin(nind);
            z:=min[nodes] div max[nind]+1;
            if z > min[maxdeg] then pushmin(maxdeg);
          end
        else
          if min[nodes] > z then
             begin
               z:=max[clique];
               if z < 5 then z:=5;
               if z < max[maxdeg] then pushmax(maxdeg);
               if min[maxdeg] >= 6 then
                  begin
                    z:=min[maxdeg];
                    if z > min[clique] then pushmin(clique);
                  end;
             end;
    end;
end;

procedure r390;
(************************************************)
(*                                              *)
(* if not cycle with 5 nodes then               *)
(*    if clique < .5*log2(2*sqrt(PI)*p) then    *)
(*              nind >= .5*log2(2*sqrt(PI)*p)   *)
(*                                              *)
(************************************************)
begin
  if (activerule[390]) and ((min[nodes] > 5) or (min[maxdeg] > 2)
or
     (max[cycle] = 0)) then
    begin
      rule:='390/ ';
      rhb:=2*sqrt(3.1415926);
      rz:=0.5*log2(rhb*min[nodes]);
      z:=round(rz+0.5);
      if max[clique] < rz then
           begin
             if z > min[nind] then pushmin(nind);
             power(4,max[nind],z);
             if z < infinity then
                begin
                  z:=trunk(z/rhb);
                  if max[nodes] > z then pushmax(nodes);
                end;
            end
          else
            if max[nind] < rz then
              begin
                if z > min[clique] then pushmin(clique);
                power(4,max[clique],z);
                if z < infinity then
                   begin
                     z:=trunk(z/rhb);
                     if max[nodes] > z then pushmax(nodes);
                   end;
               end;
    end;
end;

procedure r391;
(*****************************************************************
**)
(*
*)
(* if cir <= p-mindeg then edges <= p*(p-1)/2-mindeg(p-mindeg-1)
*)
(*
*)
(*****************************************************************
**)
var z2:integer;
begin
  if (activerule[391]) and (max[nodes] < infinity) then
    begin
      rule:='391/ ';
      z:=max[nodes];
      if z-1 >= max[mindeg]+min[mindeg] then z1:=min[mindeg]
                                        else z1:=max[mindeg];
      z:=z*(z-1) div 2-z1*(z-z1-1);
      if max[circ] <= min[nodes]-max[mindeg] then
          begin
            if z < max[edges] then pushmax(edges);
                   z:=2*max[mindeg]+1;
                      z2:=8*min[edges]+2-z*z;
                      if z2 >= 0 then
                          begin
                  z1:=2*min[mindeg]+1;
                  z1:=round((z1+sqrt(8*min[edges]+2-z1*z1))/2
+hf);
                  z:=round((z+sqrt(z2))/2+hf);
                  if z1 < z then z:=z1;
                  if z > min[nodes] then pushmin(nodes);
                             end;
          end
        else
          if min[edges] > z then
```

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

procedure r392;
(************************************************)
(*                                              *)
(* if girth = 5 then                            *)
(*         p >= 19  if mindeg = 4               *)
(*           >= 30  if mindeg = 5               *)
(*           >= 40  if mindeg >=6               *)
(*                                              *)
(************************************************)
begin
  if (activerule[392]) and (max[girth] = 5) and (min[girth] = 5)
then
    begin
      rule:='392/ ';
      z:=0;
      if min[mindeg] = 4 then z:=19
        else
          if min[mindeg] = 5 then z:=30
              else if min[mindeg] > 5 then z:=40;
      if z > min[nodes] then pushmin(nodes);
       z:=0;
      if max[nodes] < 19 then z:=3
        else if max[nodes] < 30 then z:=4
          else if max[nodes] < 40 then z:=5;
      if (z > 0) and (z < max[mindeg]) then pushmax(mindeg);
    end;
end;

procedure r393;
(************************************************)
(*                                              *)
(*  girth =6 and mindeg >= 7 then               *)
(*          p >= 93 if not regular              *)
(*            >= 90 otherwise                   *)
(*                                              *)
(************************************************)
begin
  if (activerule[393]) and (max[girth] = 6)
     and (min[girth] = 6) and (max[mindeg] >= 7) then
    begin
      rule:='393/ ';
      if min[mindeg] >= 7 then
         begin
           if max[reg] = 0 then z:=93
                  else
                    begin
                      if max[nodes] <= 92 then
                             begin
                               z:=1;
                               if min[reg]=0 then pushmin(reg);
                             end;
                       z:=90;
                      end;
            if z > min[nodes] then pushmin(nodes)
          end
        else
          if (max[nodes] < 90) or ((max[nodes] < 93) and
(max[reg] = 0)) then
             begin
               z:=6;
               pushmax(mindeg);
             end;
    end;
end;

procedure r394;
(************************************************)
(*                                              *)
(* if clique = 2 then nind >=                   *)
(*      p(2e/p*ln(2e/p)-2e/p+1)/(2e/p-1)**2     *)
(*                                              *)
(*       where ln is the natural log.           *)
(*       ln(x) > .693148*log2(x)                *)
(*                                              *)
(************************************************)
begin
  if (activerule[394]) and (max[edges] < infinity)
     and (min[clique] = 2) then
    begin
      rule:='394/ ';
      if 2*max[edges] = min[nodes] then rz:=min[nodes]/ 2
        else
          begin
            rz:=2*max[edges]/min[nodes];
            rz:=(min[nodes]*(rz*0.693148*log2(rz)-rz+
1))/((rz-1)*(rz-1));
          end;
      z:=round(rz+hf);
      if max[clique] = 2 then
        begin
          if z > min[nind] then pushmin(nind);
                 z:=min[edges];
                 if 2*z > min[nodes] then
```

```
                    begin
                         rz:=(2*z)/min[nodes];
               rz:=(min[nodes]*(rz*0.693148*log2(rz)-rz+
1))/((rz-1)*(rz-1));
                           while (rz > max[nind]) and (z <=
max[edges]) do
                               begin
                                    z:=z+1;
                                      rz:=2*z/min[nodes];
                     rz:=(min[nodes]*(rz*0.693148*log2(rz)-rz+
1))/((rz-1)*(rz-1));
                                 end;
                           if z > min[edges] then pushmin(edges);
                   end;
                 z:=max[nodes];
                 if 2*max[edges] > z then
                    begin
                         rz:=(2*max[edges])/z;
               rz:=(z*(rz*0.693148*log2(rz)-rz+
1))/((rz-1)*(rz-1));
                           while (rz > max[nind]) and (z >=
min[nodes]) do
                               begin
                                    z:=z-1;
                                      rz:=2*max[nodes]/z;
                     rz:=(z*(rz*0.693148*log2(rz)-rz+
1))/((rz-1)*(rz-1));
                                 end;
                           if z < max[nodes] then pushmax(nodes);
                   end;
        end
       else
         if max[nind] < rz then
            begin
              z:=3;
              pushmin(clique);
            end;
    end;
end;

procedure r395;
(************************************************)
(*                                              *)
(* if connected and not a tree then             *)
(*  p >= CL{(diam+1)/(g+k)}*                    *)
(*           {1+mindeg*S-((mindeg-1)**t)*k}     *)
(*                                              *)
(*       where k = (1+(-1)**g)/2                *)
(*             t = g div 2                      *)
(*             S = 1   if mindeg=1              *)
(*               = t   if mindeg=2              *)
(*               = ((mindeg-1)**t-1)/(mindeg-2) *)
(*                    if mindeg >= 3            *)
(*                                              *)
(************************************************)
var k,t,s,gi:longint;
function bound(deg,dia:longint):longint;
var z1:longint;
begin
  bound:=0;
  if (deg < infinity) and (dia < infinity) and (gi < infinity)
then
    begin
      if odd(gi) then k:=0
                 else k:=1;
      t:= gi div 2;
      z1:=0;
      if deg = 1 then s:=1
       else if deg = 2 then
                   begin
                     s:=t;
                     z1:=1;
                   end
         else
           begin
             power(deg-1,t,z1);
             if z1 < infinity then s:=(z1-1) div (deg-2)
                              else s:=infinity;
           end;
       if s < infinity then bound:=(dia div (gi+k)+1)*(1+deg*s-z1
*k);
     end;
end;

begin
  if (activerule[395]) and (min[diam] < infinity)
     and (min[girth] = max[girth]) and (min[mindeg] < infinity)
then
    begin
      rule:='395/ ';
      gi:=min[girth];
      z:=bound(min[mindeg],min[diam]);
      if (z > min[nodes]) and (z < infinity) then pushmin(nodes);
      if max[nodes] < infinity then
         begin
           z:=max[mindeg];
           if z < infinity then
             begin
               while (bound(z,min[diam]) > max[nodes]) and
                     (z >= min[mindeg]) do z:=z-1;
               if z < max[mindeg] then pushmax(mindeg);
             end;
           z:=max[diam];
           if z < infinity then
```

```
             begin
               while (bound(min[mindeg],z) > max[nodes]) and
                     (z >= min[diam]) do z:=z-1;
               if z < max[diam] then pushmax(diam);
             end;
         end;
    end;
end;

procedure r396;
(************************************************)
(*                                              *)
(*  if not a forest then                        *)
(*    edges <= p*(p-1)/(4m)+p/2                 *)
(*       where t = girth div 2                  *)
(*             m = S-((mindeg-1)**(t-1))/2      *)
(*                     when girth is odd        *)
(*               = S-(mindeg-1)**(t-1)          *)
(*                     when girth is even       *)
(*                                              *)
(*             S = 1      if mindeg = 1         *)
(*               = t      if mindeg = 2         *)
(*               = ((mindeg-1)**t-1)/(mindeg-2) *)
(*                        if mindeg >= 3        *)
(*                                              *)
(************************************************)
var t,s:longint;
    m:real;
function boundm(deg,gi:longint):real;
var z1:longint;
begin
  boundm:=infinity;
  if (deg < infinity) and (gi < infinity) then
    begin
      t:=gi div 2;
      power(deg-1,t-1,z1);
      if z1 < infinity then
         begin
           if deg = 1 then s:=1
               else if deg = 2 then s:=t
                 else if deg >= 3 then s:=((deg-1)*z1-1) div
(deg-2);
           if odd(gi) then boundm:=s-z1/2
                      else boundm:=s-z1;
         end;
    end;
end;


begin
  if (activerule[396]) and (min[mindeg] < infinity) and
      (min[girth] < infinity) then
    begin
      rule:='396/ ';
      m:=boundm(min[mindeg],min[girth]);
      if (m < infinity) and (m > 0) then
        begin
          rz:=2*m-1;
          z:=round((-rz+sqrt(rz*rz+16*m*min[edges]))/2+hf);
          if z > min[nodes] then pushmin(nodes);
          z:=max[nodes];
          if z < infinity then
            begin
              z:=trunk(z*(z-1)/(4*m)+z/2);
              if z < max[edges] then pushmax(edges);
              z:=max[nodes];
              z1:=4*min[edges]-2*z;
              if z1 > 0 then
                begin
                  rz:=z*(z-1)/z1;
                  z:=max[mindeg];
                  if z < infinity then
                     begin
                       z1:=min[girth];
                       m:=boundm(z,z1);
                       if m < infinity then
                          while (m > rz) and (z >= min[mindeg])
do
                            begin
                              z:=z-1;
                              m:=boundm(z,z1);
                            end;
                       if z < max[mindeg] then pushmax(mindeg);
                     end;
                  z:=max[girth];
                  if z < infinity then
                     begin
                       z1:=min[mindeg];
                       m:=boundm(z1,z);
                       if m < infinity then
                       while (m > rz) and (z >= min[girth]) do
                          begin
                            z:=z-1;
                            m:=boundm(z1,z);
                          end;
                      if z < max[girth] then pushmax(girth);
                     end;
                  end;
               end;
         end;
    end;
end;

procedure r397;
```

```
(****************************************************)
(*                                                  *)
(*  if girth >= 5+4*log3(max{1,genus} then arb <= 2 *)
(*                                                  *)
(****************************************************)
begin
  if (activerule[397]) and (max[arbor] >= 3) then
    begin
      rule:='397/ ';
      z:=max[genus];
      if z = 0 then z:=4
          else if z < infinity then z:=round(4
*log2(z)/log2(3)+hf)+4;
      if min[arbor] >= 3 then
         begin
           if z < max[girth] then pushmax(girth);
           z1:=min[girth];
           if z1 >= 4 then
              begin
                if z1 = 5 then z:=2
                   else
                     begin
                       realPower(sqrt(sqrt(3)),min[girth]-5,rz);
                       z:=trunk(rz)+1;
                     end;
                if z > min[genus] then pushmin(genus);
              end;
          end
         else
           if min[girth] > z then
              begin
                z:=2;
                pushmax(arbor);
              end;
    end;
end;

procedure r398;
(************************************************)
(*                                              *)
(*  if mindeg >= 2 then p >= gi*Nc+maxdeg-2     *)
(*                                              *)
(************************************************)
begin
  if (activerule[398]) and (max[mindeg] >= 2) then
    begin
      rule:='398/ ';
      z:=min[girth]*min[ncomp]+min[maxdeg]-2;
      if min[mindeg] >= 2 then
         begin
           if z > min[nodes] then pushmin(nodes);
           if max[nodes] < infinity then
```

```
             begin
               z:=max[nodes]-min[girth]*min[ncomp]+2;
               if z < max[maxdeg] then pushmax(maxdeg);
               z1:=max[nodes]-min[maxdeg]+2;
               z:=z1 div min[girth];
               if z < max[ncomp] then pushmax(ncomp);
               z:=z1 div min[ncomp];
               if z < max[girth] then pushmax(girth);
             end;
         end
       else
         if max[nodes] < z then
             begin
               z:=1;
               pushmax(mindeg);
             end;
    end;
end;

procedure r399;
(************************************************)
(*                                              *)
(*   if nconn > 0 then nconn >=                 *)
(*                     p(x-2)/((x-1)**diam+x-3  *)
(*           where x = maxdeg                   *)
(*                                              *)
(************************************************)
begin
  if (activerule[399]) and (min[nconn] > 0) and (max[maxdeg]
< infinity)
     and (max[diam] < infinity) and (max[maxdeg] > 2) then
    begin
      rule:='399/ ';
      z:=max[maxdeg];
      power(z-1,max[diam],k);
      if k < infinity then
         begin
           z1:=k+z-3;
           z:=(min[nodes]*(z-2)-1) div z1+1;
           if z > min[nconn] then pushmin(nconn);
           if max[nconn] < infinity then
             begin
               z:=(max[nconn]*z1) div (max[maxdeg]-2);
               if z < max[nodes] then pushmax(nodes);
               z:=(min[nodes]-1) div max[nconn];
               z:=z*(max[maxdeg]-2)+1;
               z:=round(log2(z)/log2(max[maxdeg]-1)+hf);
               if z > min[diam] then pushmin(diam);
               k:=(min[nodes]-1) div max[nconn];
               z1:=k-1;
               z:=min[maxdeg]-1;
               if z > 2 then
```

```
                 while z1 < k do
                  begin
                    z:=z+1;
                    power(z-1,max[diam],z1);
                    z1:=(z1-1) div (z-2);
                  end;
               if z > min[maxdeg] then pushmin(maxdeg);
             end;
          end;
    end;
end;

procedure r400;
(*******************************************************)
(*                                                     *)
(*  if diam = 2 , let m=maxdeg                         *)
(*                                                     *)
(*        2p-4,      if (2p-2)/3 <= m <  p-4, or m=p-2 *)
(*   e >= 3p-m-6,    if (3p-5)/5 <= m < (2p-2)/3       *)
(*        5p-4m-10,  if (5p-3)/9 <= m < (3p-3)/5       *)
(*        4p-2m-13,  if  (p+1)/2 <= m < (5p-3)/9       *)
(*                                                     *)
(*******************************************************)
var mz,mz1:longint;
begin
  if (activerule[400]) and (max[diam]=min[diam]) and (max[diam] =
2) and
     (2*min[maxdeg]-1 >= max[nodes]) and (max[maxdeg]
< min[nodes]-1) then
    begin
      rule:='400/ ';
      z:=min[nodes];
      mz:=max[nodes];
      z1:=max[maxdeg];
      mz1:=min[maxdeg];
      if ((mz+1)/2 <= mz1) and (z1 < (5*z-3)/9) then
        begin
          z:=4*z-2*z1-13;
          if z > min[edges] then pushmin(edges);
          z:=(4*min[nodes]-max[edges]-12) div 2;
          if z > min[maxdeg] then pushmin(maxdeg);
          z:=(max[edges]+2*z1+13) div 4;
          if z < max[nodes] then pushmax(nodes);
        end
       else
         if ((5*mz-3)/9 <= mz1) and (z1 < (3*z-3)/5) then
          begin
            z:=5*z-4*z1-10;
            if z > min[edges] then pushmin(edges);
            z:=(5*min[nodes]-max[edges]-7) div 4;
            if z > min[maxdeg] then pushmin(maxdeg);
            z:=(max[edges]+4*z1+10) div 5;
            if z < max[nodes] then pushmax(nodes);
           end
        else
          if ((3*mz-3)/5 <= mz1) and (z1 < (2*z-2)/3) then
           begin
             z:=3*z-z1-6;
             if z > min[edges] then pushmin(edges);
             z:=3*min[nodes]-max[edges]-6;
             if z > min[maxdeg] then pushmin(maxdeg);
             z:=(max[edges]+z1+6) div 3;
             if z < max[nodes] then pushmax(nodes);
           end
         else
           if (((2*mz-2)/3 <= mz1) and (z1 < z-4))
               or (min[maxdeg] = max[nodes]-2) then
           begin
             z:=2*z-4;
             if z > min[edges] then pushmin(edges);
             z:=(max[edges]+4) div 2;
             if z < max[nodes] then pushmax(nodes);
            end;
    end;
end;


end.
```