Bevezetés a matematikába

Járai Antal

Ezek a programok csak szemléltetésre szolgálnak.

1. Halmazok

2. Természetes számok

3. A számfogalom bővítése

4. Véges halmazok

5. Végtelen halmazok

6. Számelmélet

7. Gráfelmélet

7.1. Irányítatlan gráfok

7.1.1. Irányítatlan gráfok.

```
> restart;
> with(networks);
```

acycpoly, addedge, addvertex, adjacency, allpairs, ancestor, arrivals, bicomponents, charpoly, chrompoly, complement, complete, components, connect, connectivity, contract, countcuts, counttrees, cube, cycle, cyclebase, daughter, degreeseq, delete, departures, diameter, dinic, djspantree, dodecahedron, draw, draw3d, duplicate, edges, ends, eweight, flow, flowpoly, fundcyc, getlabel, girth, graph, graphical, gsimp, gunion, head, icosahedron, incidence, incident, indegree, induce, isplanar, maxdegree, mincut, mindegree, neighbors, new, octahedron, outdegree, path, petersen, random, rank, rankpoly, shortpathtree, show, shrink, (7.1.1.1) *span, spanpoly, spantree, tail, tetrahedron, tuttepoly, vdegree, vertices, void, vweight*]

```
> new(G1):addvertex({v1,v2,v3,v4,v5},G1);
  addedge([{v1,v2},{v1,v2},{v1,v4},{v3,v4},{v4,v4}],G1);
  edges(G1);vertices(G1);
  ends(e2,G1);
  edges({v1,v2},G1);
  incident(v1,G1);incident(v5,G1);
  neighbors(v1,G1);neighbors(v4,G1);
```

$$v1, v2, v3, v5, v4$$
$$e1, e2, e3, e4, e5$$
$$\{e1, e2, e3, e4, e5\}$$
$$\{v1, v2, v3, v5, v4\}$$
$$\{v1, v2\}$$
$$\{e1, e2\}$$
$$\{e1, e2, e3\}$$
$$\{\ \}$$
$$\{v2, v4\}$$
$$\{v1, v3, v4\} \quad (7.1.1.2)$$

```
> vdegree(v1,G1);vdegree(v4,G1);vdegree(v5,G1);degreeseq(G1);
  mindegree(G1);maxdegree(G1);
```

$$3$$
$$3$$
$$0$$
$$[0, 1, 2, 3, 3]$$
$$0$$
$$3 \quad (7.1.1.3)$$

```
> show(G1);
```

table([_*Tail* = *table*([ ]), _*Counttrees* = _*Counttrees*, _*Vertices* = ({*v1*, *v2*, *v3*, *v5*, *v4*}), _*Vweight* = *table*(*sparse*, [ ]), _*EdgeIndex* = *table*(*symmetric*, [(*v3*, *v4*) = {*e4*}, (*v1*, *v4*) = {*e3*}, *v4* = {*e5*}, (*v1*, *v2*) = ({*e1*, *e2*})]), _*Econnectivity* = _*Econnectivity*, _*Emaxname* = 5, _*Head* = *table*([ ]), _*Eweight* = *table*([*e4* = 1, *e1* = 1, *e5* = 1, *e2* = 1, *e3* = 1]), _*Edges* = ({*e1*, *e2*, *e3*, *e4*, *e5*}), _*Neighbors* = *table*([*v4* = ({*v1*, *v3*, *v4*}), *v5* = { }, *v2* = {*v1*}, *v3* = {*v4*}, *v1* = ({*v2*, *v4*})]), _*Status* = ({*MULTIGRAPH*, *LOOPS*}), _*Ends* = *table*([*e4* = ({*v3*, *v4*}), *e1* = ({*v1*, *v2*}), *e5* = {*v4*}, *e2* = ({*v1*, *v2*}), *e3* = ({*v1*, *v4*})]), _*Bicomponents* = _*Bicomponents*, _*Countcuts* = _*Countcuts*]) (7.1.1.4)

```
> G:=void(10):vertices(G);edges(G);
  addedge([{1,2},{2,3},{3,4}],names=[cica,alma,kutya],G);show
  (G);
```

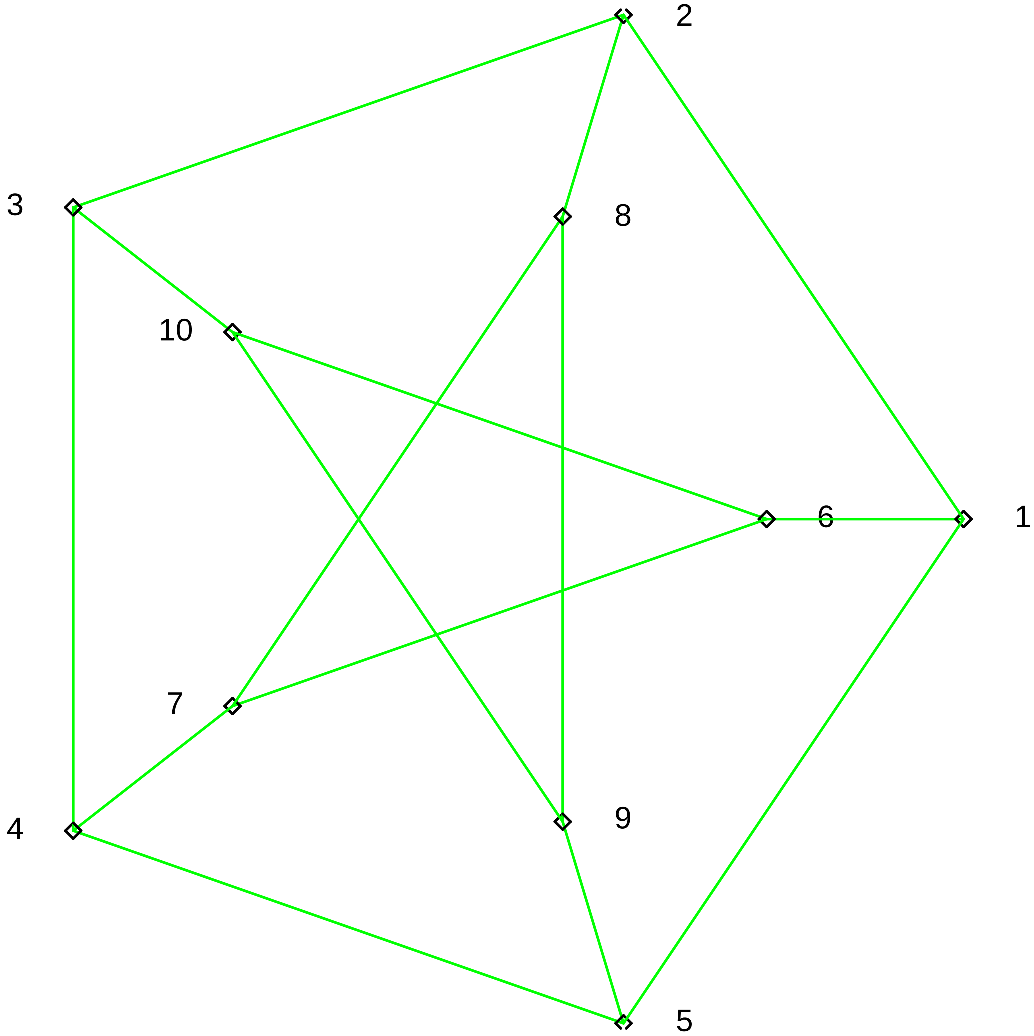
$$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

$$\{\ \}$$

$$cica, alma, kutya$$

$$table([\_Tail = table([\,]), \_Counttrees = \_Counttrees, \_Vertices = (\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}), \_Vweight = table(sparse, [\,]), \_EdgeIndex = table(symmetric, [(3, 4) = \{kutya\}, (2, 3) = \{alma\}, (1, 2) = \{cica\}]), \_Econnectivity = \_Econnectivity, \_Head = table([\,]), \_Eweight = table([alma = 1, kutya = 1, cica = 1]), \_Edges = (\{cica, alma, kutya\}), \_Neighbors = table([1 = \{2\}, 2 = (\{1, 3\}), 3 = (\{2, 4\}), 5 = \{\}, 4 = \{3\}, 7 = \{\}, 6 = \{\}, 10 = \{\}, 8 = \{\}, 9 = \{\}]), \_Ends = table([alma = (\{2, 3\}), kutya = (\{3, 4\}), cica = (\{1, 2\})]), \_Bicomponents = \_Bicomponents, \_Countcuts = \_Countcuts]) \quad (7.1.1.5)$$

```
> G:=petersen():draw(G);degreeseq(G);
```



[3, 3, 3, 3, 3, 3, 3, 3, 3, 3] (7.1.1.6)

▶ *–>7.1.2. Feladat.*

▶ *7.1.3. Feladat.*

▶ *7.1.4. Feladat.*

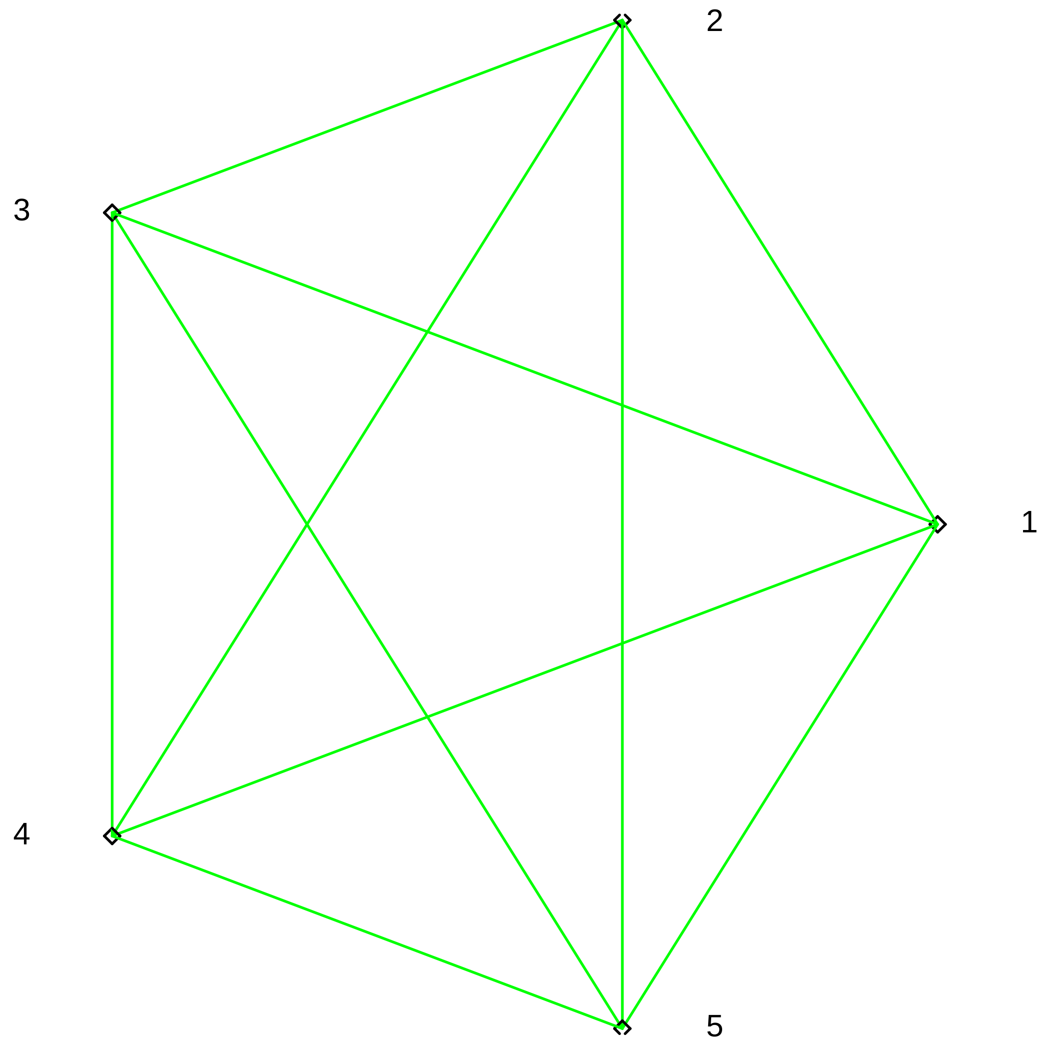
▶ *7.1.5. Feladat.*

▶ **7.1.6. Feladat.*

▼ *7.1.7. Gráfok izomorfiája.*

▼ *7.1.8. Példák.*

```
> G21:=complete(5):draw(G21);
```



7.1.9. Gráfok Descartes-szorzata.

->7.1.10. Feladat.

->7.1.11. Feladat.

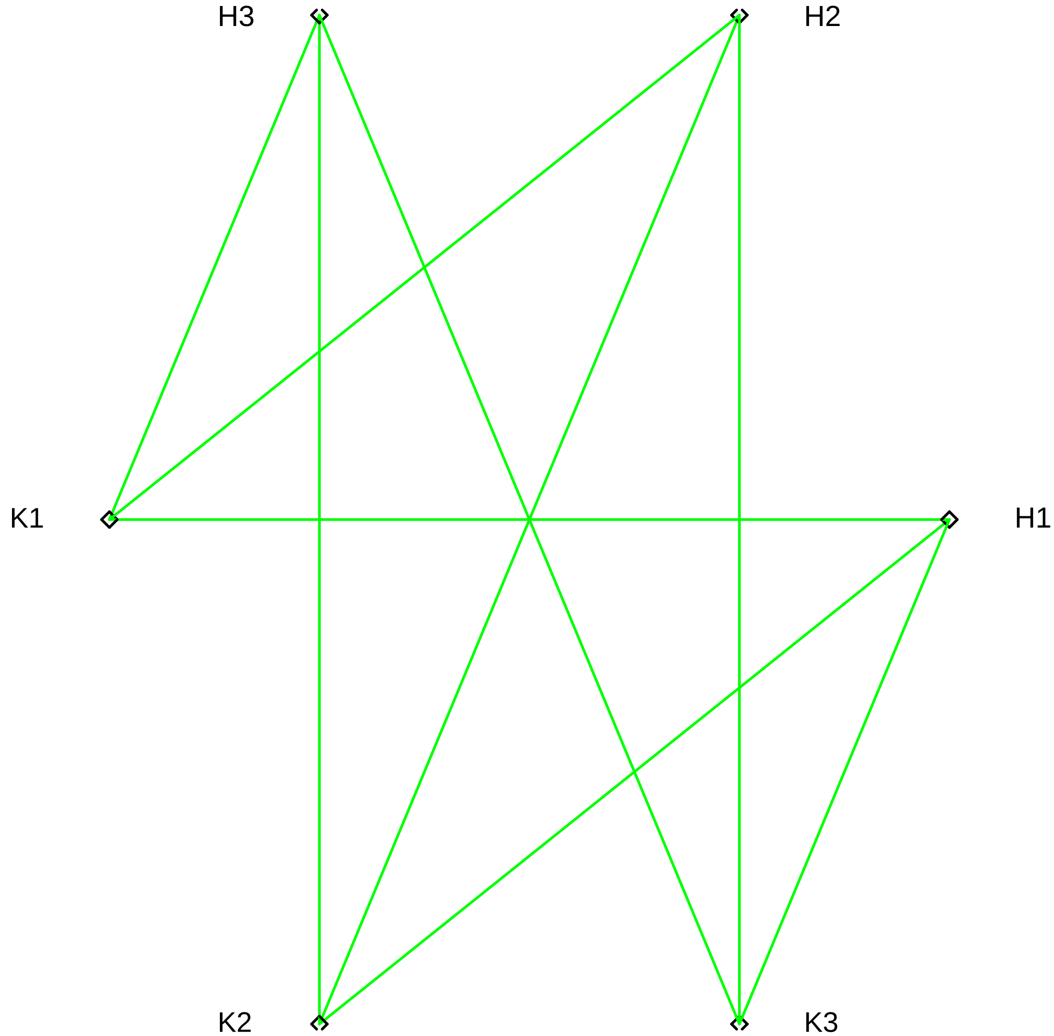
->7.1.12. Feladat.

7.1.13. Páros gráfok.

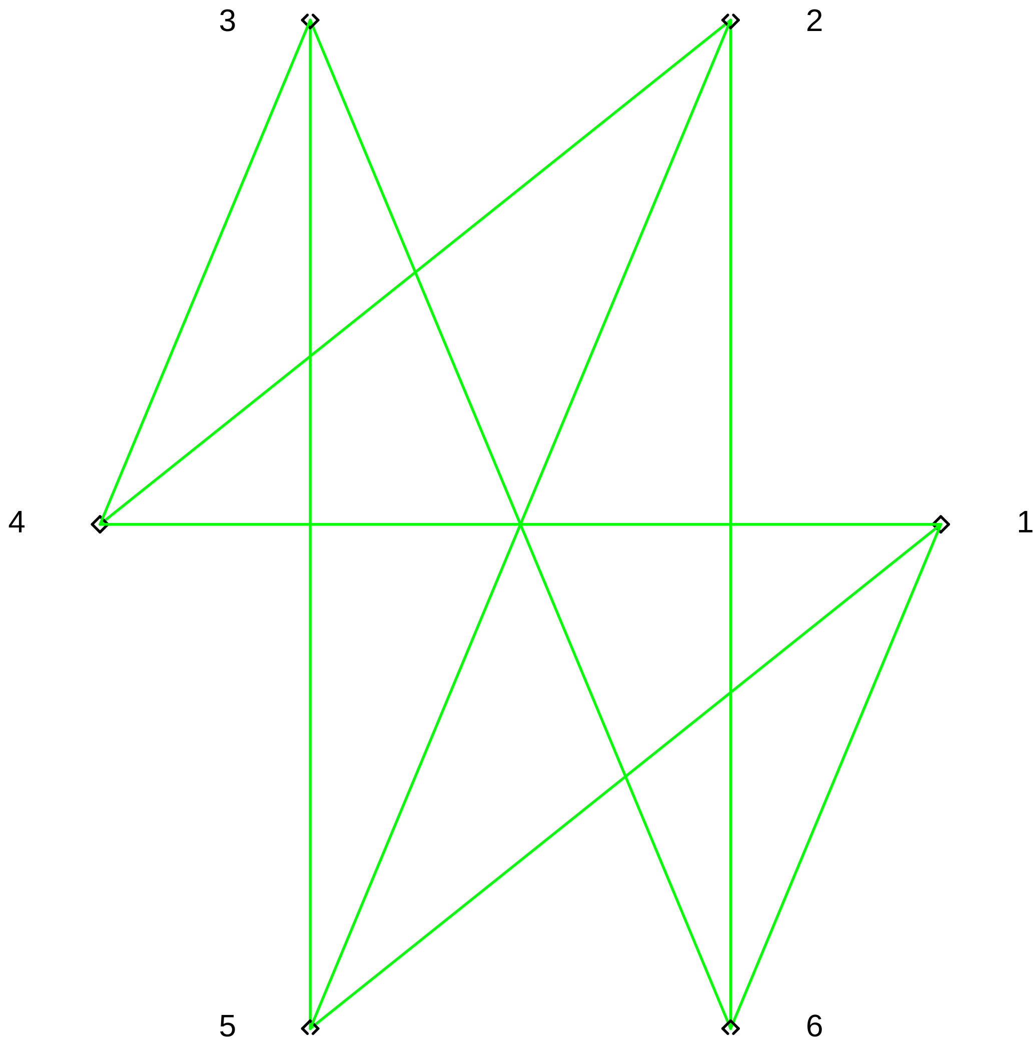
```
> new(G22):addvertices([H1,H2,H3,K1,K2,K3],G22);connect({H1,
  H2,H3},{K1,K2,K3},G22);draw(G22);
```

H1, H2, H3, K2, K1, K3

e1, e2, e3, e4, e5, e6, e7, e8, e9




```
> G:=complete(3,3):draw(G);
```



->*7.1.14. Feladat.*

7.1.15. Részgráf.

```
> show(G);
```

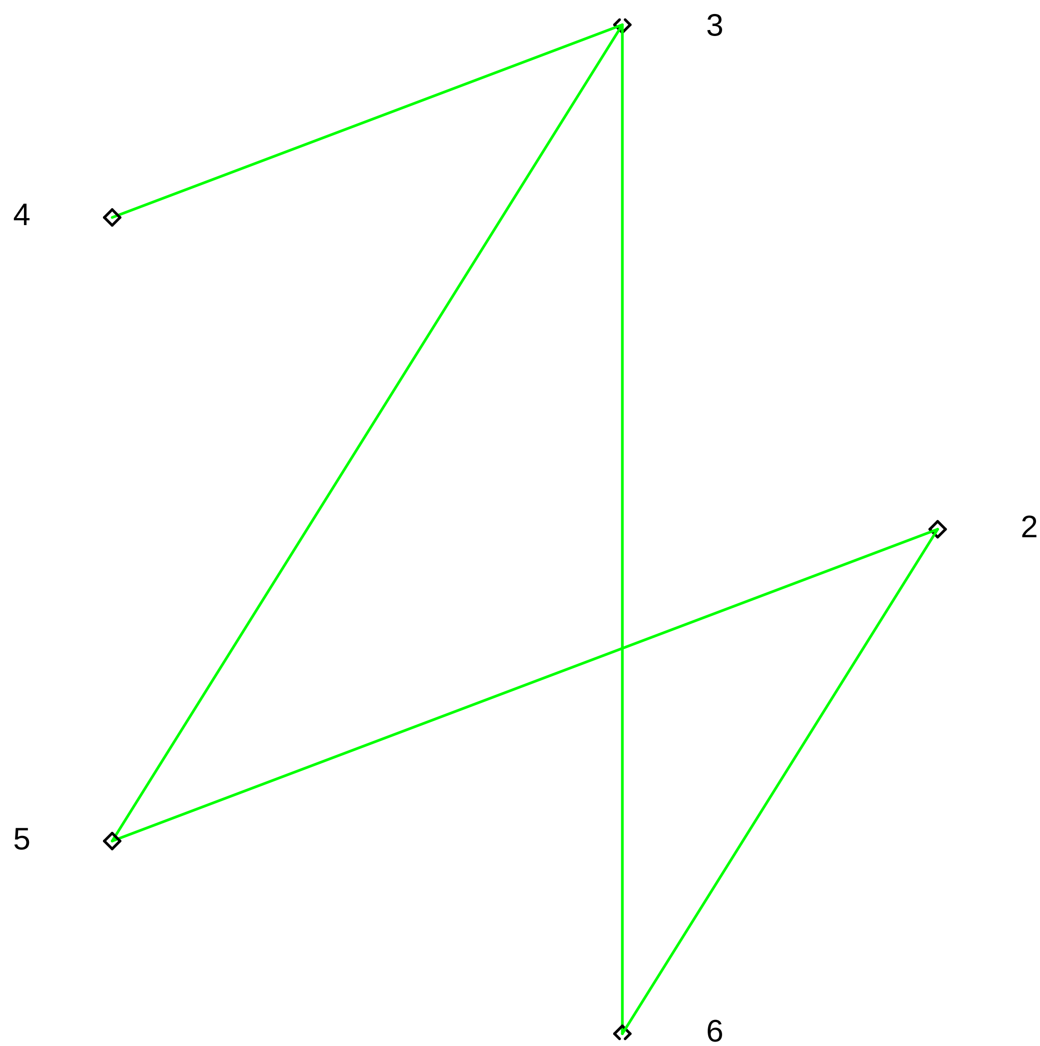
$table([\_Tail = table([\,]), \_Counttrees = \_Counttrees, \_Vertices = (\{1, 2, 3, 4, 5, 6\}), \_Vweight = table(sparse, [\,]), \_EdgeIndex = table(symmetric, [(1, 5) = \{e2\}, (1, 4) = \{e1\}, (2, 5) = \{e5\}, (3, 6) = \{e9\}, (2, 4) = \{e4\}, (2, 6) = \{e6\}, (3, 4) = \{e7\}, (3, 5) = \{e8\}, (1, 6) = \{e3\}]), \_Econnectivity = \_Econnectivity, \_Emaxname = 9, \_Head = table([\,]), \_Eweight = table([e4 = 1, e9 = 1, e7 = 1, e1 = 1, e5 = 1, e6 = 1, e2 = 1, e8 = 1, e3 = 1]), \_Edges = (\{e1, e2, e3, e4, e5, e6, e7, e8, e9\}), \_Neighbors = table([1 = (\{4, 5, 6\}), 2 = (\{4, 5, 6\}), 3 = (\{4, 5, 6\}), 5 = (\{1, 2, 3\}), 4 = (\{1, 2, 3\}), 6 = (\{1, 2, 3\})]), \_Status = (\{SIMPLE, BIPARTITE\}),$ (7.1.15.1)

_Ends = table([e4 = ({2, 4}), e9 = ({3, 6}), e7 = ({3, 4}),
e1 = ({1, 4}), e5 = ({2, 5}), e6 = ({2, 6}), e2 = ({1, 5}),
e8 = ({3, 5}), e3 = ({1, 6})]), _Bicomponents = _Bicomponents,
_Countcuts = _Countcuts])

```
> delete({e4},delete({1},G)):vertices(%);edges(%%);draw(%%%);
```

{2, 3, 4, 5, 6}

{e5, e6, e7, e8, e9}




```
> show(G1);
```

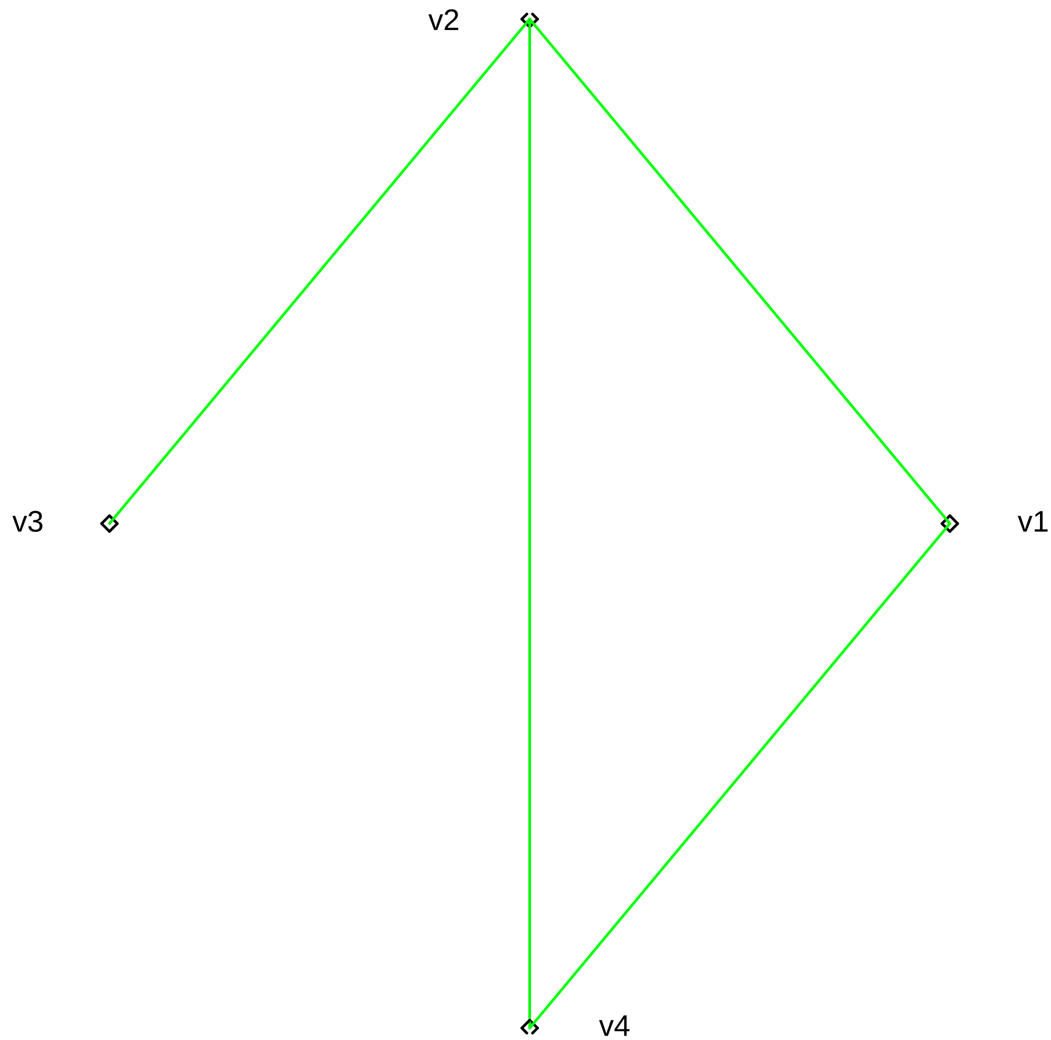
table([_Tail = table([]), _Counttrees = _Counttrees, (7.1.15.2)
_Vertices = ({v1, v2, v3, v5, v4}), _Vweight = table(sparse, []),
_EdgeIndex = table(symmetric, [(v3, v4) = {e4}, (v1,
v4) = {e3}, v4 = {e5}, (v1, v2) = ({e1, e2})]),
_Econnectivity = _Econnectivity, _Emaxname = 5,
_Head = table([]), _Eweight = table([e4 = 1, e1 = 1, e5 = 1,
e2 = 1, e3 = 1]), _Edges = ({e1, e2, e3, e4, e5}),
_Neighbors = table([v4 = ({v1, v3, v4}), v5 = {}, v2 = {v1},

$v3 = \{v4\}, v1 = (\{v2, v4\})]), \_Status = (\{MULTIGRAPH,$
$LOOPS\}), \_Ends = table([e4 = (\{v3, v4\}), e1 = (\{v1, v2\}),$
$e5 = \{v4\}, e2 = (\{v1, v2\}), e3 = (\{v1, v4\})]),$
$\_Bicomponents = \_Bicomponents, \_Countcuts = \_Countcuts])$

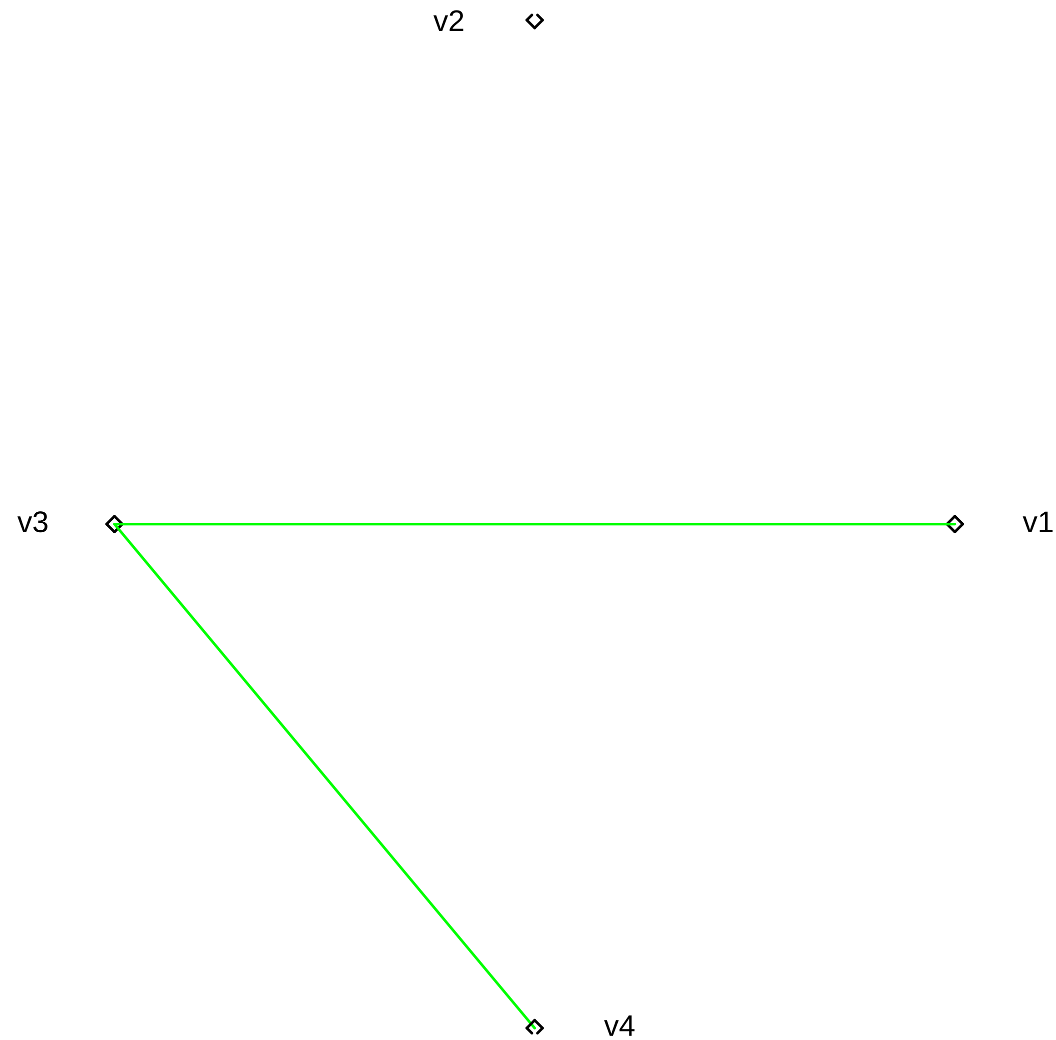
```
> new(G3):addvertex({v1,v2,v3,v4},G3);addedge([{v1,v2},{v2,
v3},{v1,v4},{v2,v4}],G3);draw(G3);
```

$$v1, v2, v3, v4$$

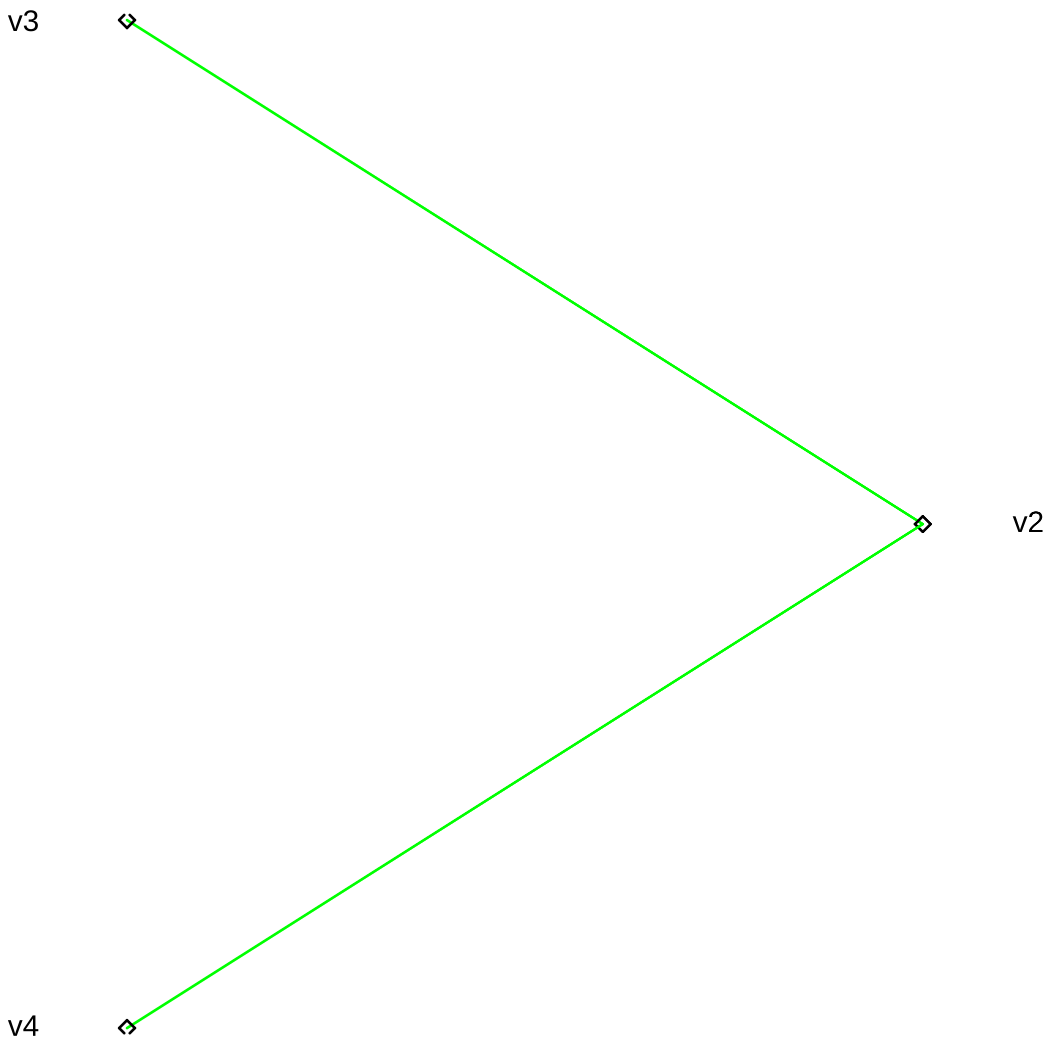
$$e1, e2, e3, e4$$



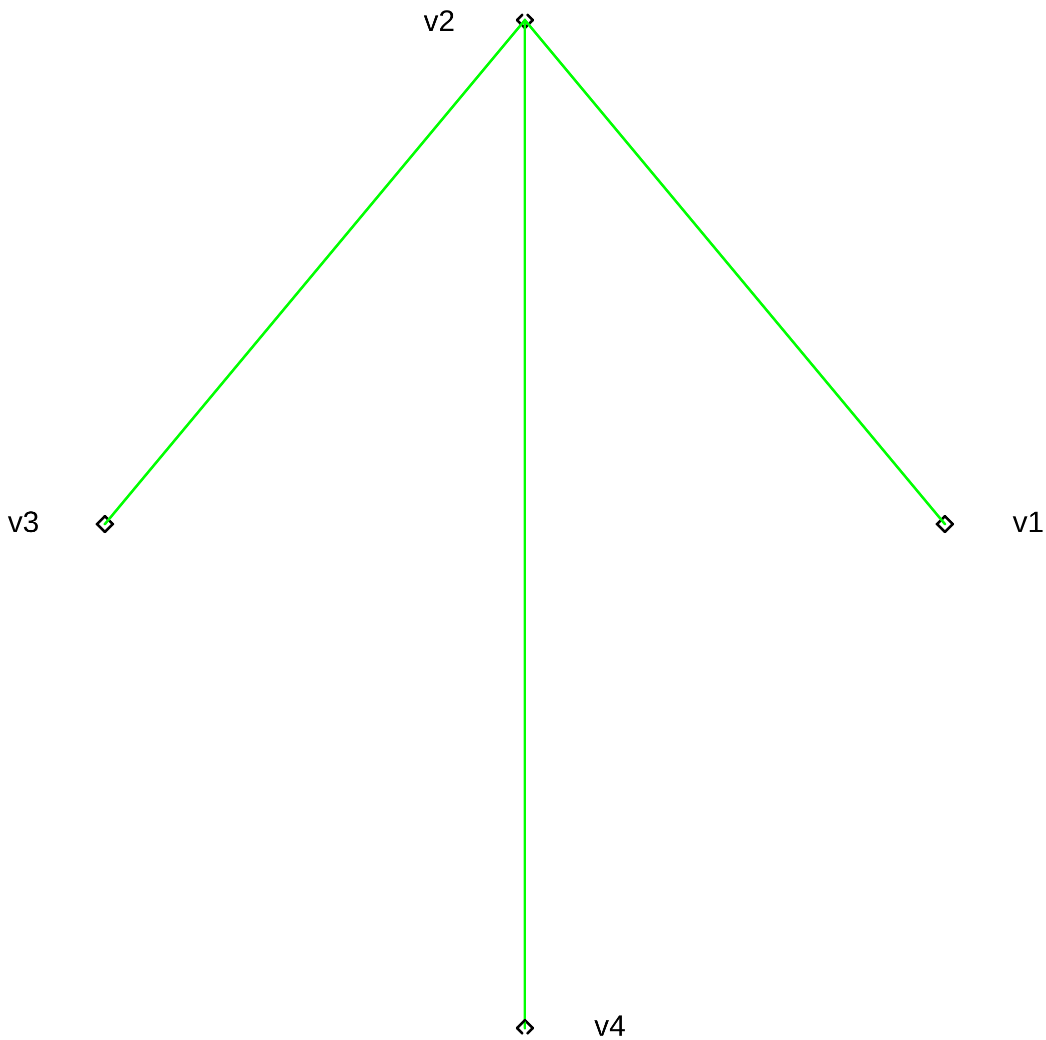

```
> draw(complement(G3));
```




```
> induce({v2,v3,v4},G3):draw(%);
```




```
> induce({e2,e1,e4},G3):draw(%);
```



->7.1.16. Feladat.

->7.1.17. Feladat.

->7.1.18. Feladat.

7.1.19. Séták, vonalak, utak, körök.

```
> G4:=void(9):addedge(Path(1,2,3,4,5,6,7,3,8,9,8),G4);show
  (G4);
```

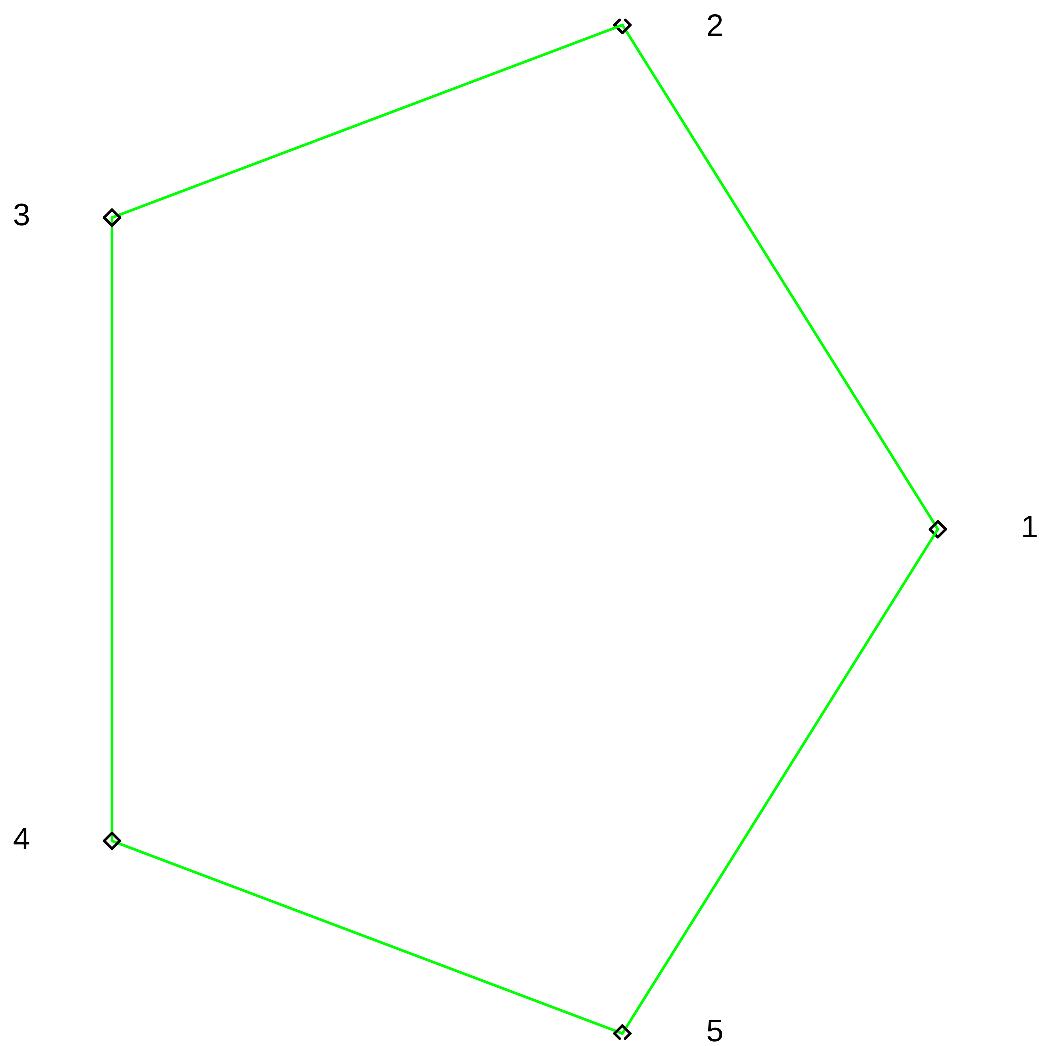
$e1, e2, e3, e4, e5, e6, e7, e8, e9, e10$

$table([\_Tail = table([]), \_Counttrees = \_Counttrees, \_Vertices = (\{1, 2, 3, 4, 5, 6, 7, 8, 9\}), \_Vweight = table(sparse, []), \_EdgeIndex = table(symmetric, [(8, 9) = (\{e9, e10\}), (3, 7) = \{e7\}, (3, 4) = \{e3\}, (3, 8) = \{e8\}, (2, 3) = \{e2\}, (5, 6) = \{e5\}, (1, 2) = \{e1\}, (6, 7) = \{e6\}, (4, 5) = \{e4\}]),$ (7.1.19.1)

_Econnectivity = _Econnectivity, _Emaxname = 10,
_Head = table([ ]), _Eweight = table([e4 = 1, e9 = 1, e7 = 1,
e1 = 1, e5 = 1, e6 = 1, e10 = 1, e2 = 1, e8 = 1, e3 = 1]),
_Edges = ({e1, e2, e3, e4, e5, e6, e7, e8, e9, e10}),
_Neighbors = table([1 = {2}, 2 = ({1, 3}), 3 = ({2, 4, 7, 8}),
5 = ({4, 6}), 4 = ({3, 5}), 7 = ({3, 6}), 6 = ({5, 7}), 8 = ({3,
9}), 9 = {8}]), _Status = {MULTIGRAPH},
_Ends = table([e4 = ({4, 5}), e9 = ({8, 9}), e7 = ({3, 7}),
e1 = ({1, 2}), e5 = ({5, 6}), e6 = ({6, 7}), e10 = ({8, 9}),
e2 = ({2, 3}), e8 = ({3, 8}), e3 = ({3, 4})]),
_Bicomponents = _Bicomponents, _Countcuts = _Countcuts])

```
> G:=void(5):addedge(Cycle(1,2,3,4,5),G);draw(G);
```

e1, e2, e3, e4, e5



▶ ***7.1.20. Állítás.***

7.1.21. Állítás.

->7.1.22. Feladat.

->7.1.23. Feladat.

7.1.24. Összefüggőség.

```
> G:=random(12,6):ends(G);components(G);
```

$$\{\{2, 3\}, \{6, 8\}, \{3, 8\}, \{4, 6\}, \{2, 10\}, \{2, 12\}\}$$

$$\{\{1\}, \{5\}, \{7\}, \{9\}, \{11\}, \{2, 3, 4, 6, 8, 10, 12\}\} \qquad (7.1.24.1)$$

->7.1.25. Feladat.

->7.1.26. Feladat.

7.1.27. Feladat.

7.1.28. Fák.

7.1.29. Tétel.

7.1.30. Tétel.

7.1.31. Tétel.

->7.1.32. Feladat.

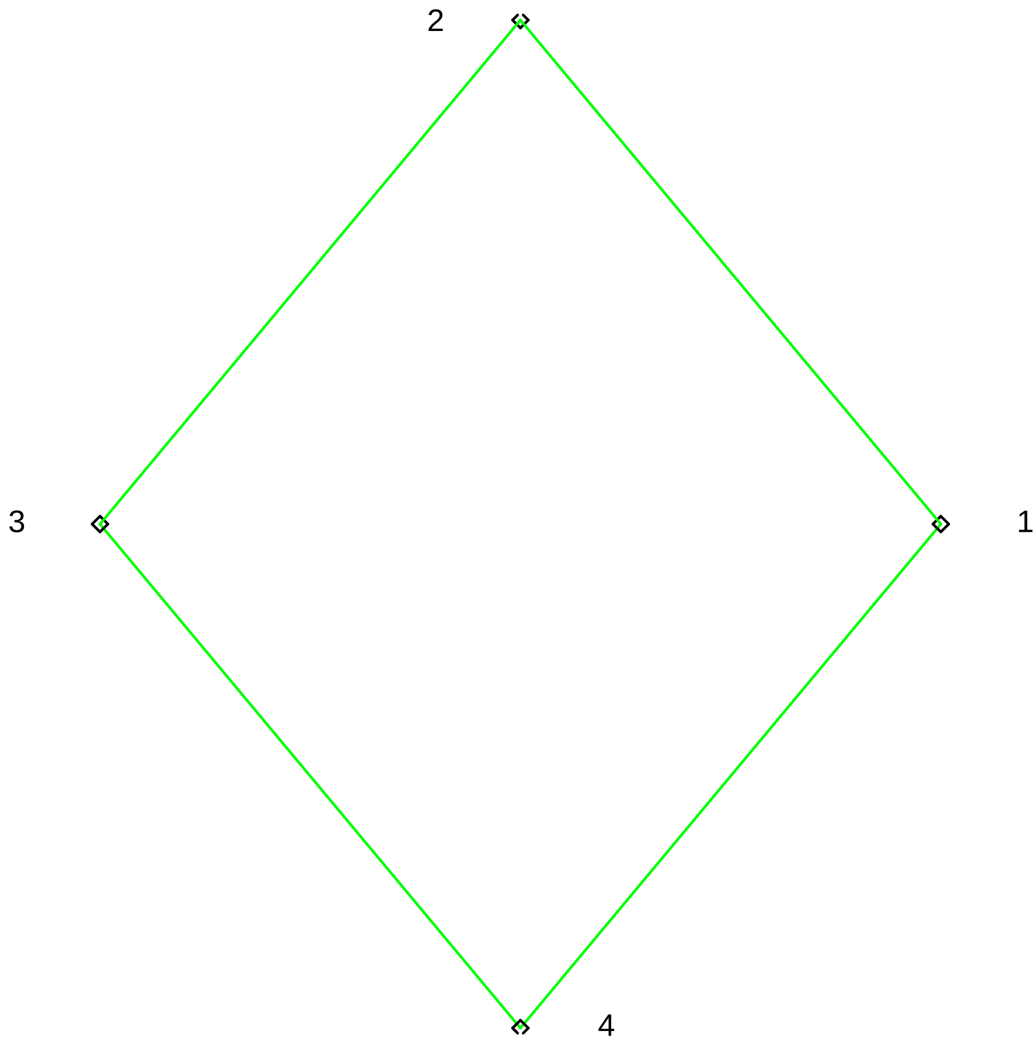
->7.1.33. Feladat.

7.1.34. Feladat.

****7.1.35. Feladat.***

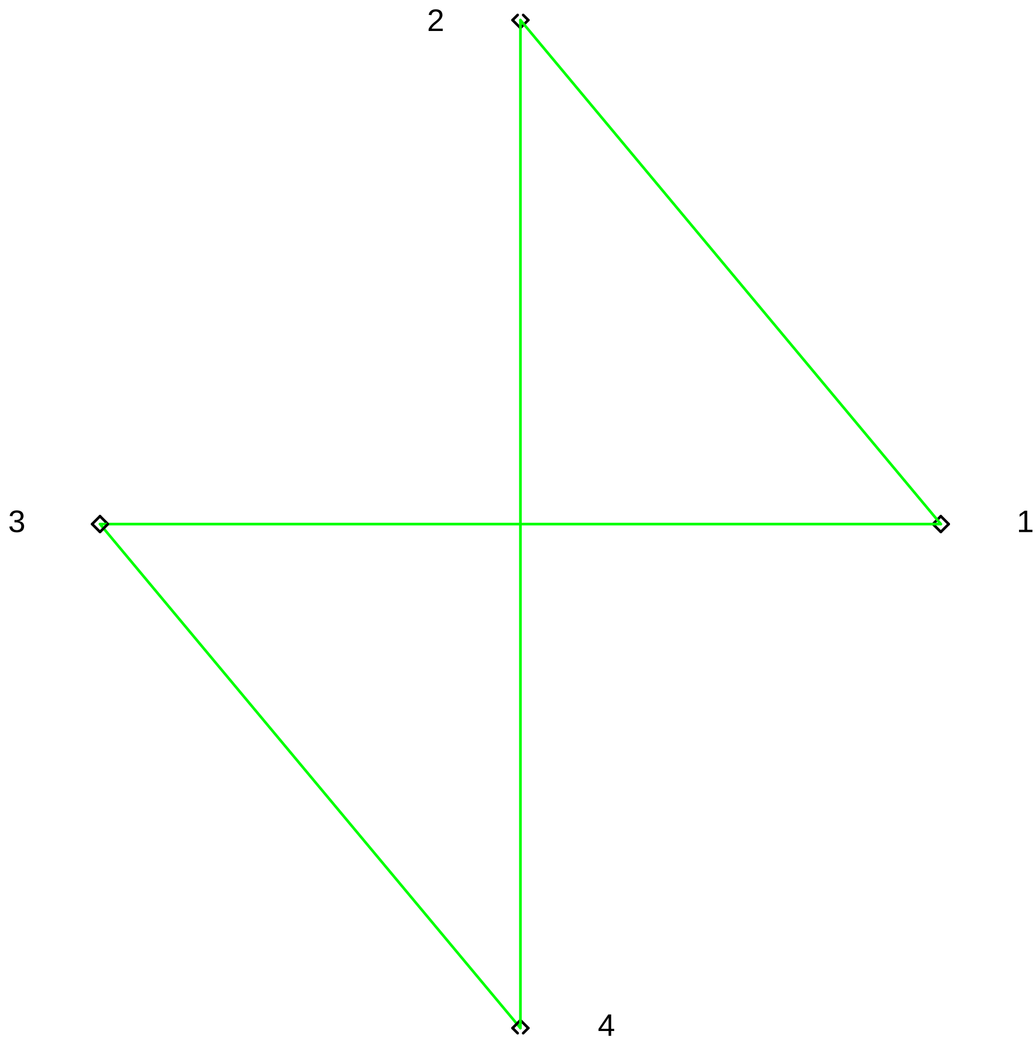
7.1.36. Feszítőfa.

```
> G51:=cycle(4):draw(G51);
```




```
> G52:=void(4):addedge(Cycle(1,2,4,3),G52);draw(G52);
```

e1, e2, e3, e4




```
> spantree(G52):edges(%);counttrees(G52);
```

$\{e1, e2, e4\}$

4 (7.1.36.1)

7.1.37. Állítás.

7.1.38. Állítás.

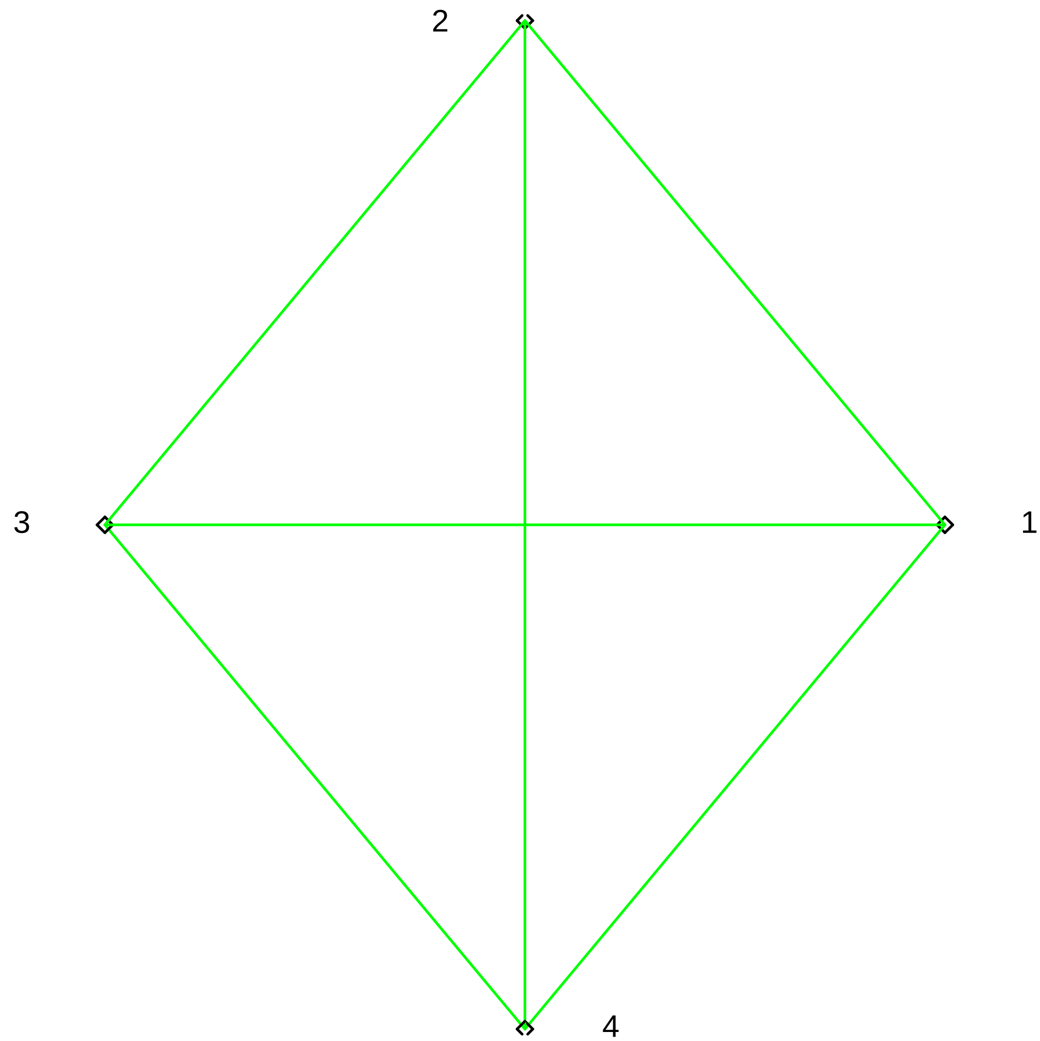
7.1.39. Megjegyzés.

```
> G6:=tetrahedron():show(G);cyclebase(G6);draw(G6);
```

table([_Tail = *table*([]), _Counttrees = _Counttrees, _Vertices = ({1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}), _Vweight = *table*(*sparse*, []), _EdgeIndex = *table*(*symmetric*, [(6, 8) = {e2}, (4, 6) = {e4}, (3, 8) = {e3}, (2, 3) = {e1}, (2, 12) = {e6}, (2, 10) = {e5}]), _Econnectivity = _Econnectivity, _Emaxname = 6, _Head = *table*([]),

$\_Eweight = table([e4 = 1, e1 = 1, e5 = 1, e6 = 1, e2 = 1, e3 = 1]),$
$\_Edges = (\{e1, e2, e3, e4, e5, e6\}), \_Neighbors = table([1 = \{\}, 2 = (\{3, 10, 12\}), 3 = (\{2, 8\}), 5 = \{\}, 4 = \{6\}, 7 = \{\}, 6 = (\{4, 8\}), 10 = \{2\}, 11 = \{\}, 8 = (\{3, 6\}), 9 = \{\}, 12 = \{2\}]), \_Ends = table([e4 = (\{4, 6\}), e1 = (\{2, 3\}), e5 = (\{2, 10\}), e6 = (\{2, 12\}), e2 = (\{6, 8\}), e3 = (\{3, 8\})]), \_Bicomponents = \_Bicomponents, \_Countcuts = \_Countcuts])$

$$\{\{e2, e3, e6\}, \{e1, e3, e4, e6\}, \{e1, e3, e5\}\}$$



▶ ***7.1.40. Vágás.***

▼ ***7.1.41. Állítás.***

```
> cycle(4):countcuts(%);
```

$$6 \qquad (7.1.41.1)$$

▶ ***7.1.42. Erdő.***

▶ ***->7.1.43. Feladat.***

▶ *7.1.44. Euler-vonal.*
▶ *7.1.45. Állítás.*
▶ *->7.1.46. Feladat.*
▶ *->7.1.47. Feladat.*
▼ *7.1.48. Hamilton-út.*

▶ *->7.1.49. Feladat.*
▶ *->7.1.50. Feladat.*
▶ *7.1.51. Feladat.*
▶ *->7.1.52. Feladat.*
▶ *->7.1.53. Feladat.*
▶ *->7.1.54. Feladat.*
▶ *7.1.55. Feladat.*
▶ *7.1.56. Feladat.*
▶ *7.1.57. Feladat.*
▶ **7.1.58. Feladat: Dirac tétele.*
▶ **7.1.59. Feladat.*
▼ *7.1.60. Címkézett és súlyozott gráfok.*

```
> new(G9):addvertex([v1,v2,v3,v4],weights=[2,4,6,8],G9);
  addedge([{v1,v2},{v2,v3},{v1,v3},{v3,v4}],weights=[1,1,1,3]
  ,G9);show(G9);
```

v1, v2, v3, v4

e1, e2, e3, e4

table([_Tail = *table*([ ]), _Counttrees = _Counttrees, _Vertices = ({v1, v2, v3, v4}), _Vweight = *table*(*sparse*, [v4 = 8, v2 = 4, v3 = 6, v1 = 2]), _EdgeIndex = *table*(*symmetric*, [(v3, v4) = {e4}, (v2, v3) = {e2}, (v1, v2) = {e1}, (v1, v3) = {e3}]), _Econnectivity = _Econnectivity, _Emaxname = 4, _Head = *table*([ ]), _Eweight = *table*([e4 = 3, e1 = 1, e2 = 1, e3 = 1]), _Edges = ({e1, e2, e3, e4}), _Neighbors = *table*([v4 = {v3}, v2 = ({v1, v3}), v3 = ({v1, v2, v4}), v1 = ({v2, v3})]), _Ends = *table*([e4 = ({v3, v4}), e1 = ({v1, v2}), e2 = ({v2, v3}), e3 = ({v1, v3})]), _Bicomponents = _Bicomponents, _Countcuts = _Countcuts]) (7.1.60.1)

▼ *7.1.61. Mohó algoritmus minimális összsúlyú feszítőerdő konstrukciójára.*

```
> spantree(G9):ends(%);
```
{{v1, v2}, {v3, v4}, {v1, v3}} (7.1.61.1)

7.1.62. Mohó algoritmusok.

7.1.63. Feladat: minimális összsúlyú feszítőfa fanövesztéssel.

7.1.64. Feladat: piros-kék algoritmus.

*7.1.65. Feladat: a kínai postás-probléma.

*7.1.66. Feladat: az utazó ügynök problémája.

7.1.67. További feladatok.

7.2. Irányított gráfok

7.2.1. Irányított gráfok.

```
> restart;with(networks);
```
[acycpoly, addedge, addvertex, adjacency, allpairs, ancestor, arrivals, bicomponents, charpoly, chrompoly, complement, complete, components, connect, connectivity, contract, countcuts, counttrees, cube, cycle, cyclebase, daughter, degreeseq, delete, departures, diameter, dinic, djspantree, dodecahedron, draw, draw3d, duplicate, edges, ends, eweight, flow, flowpoly, fundcyc, getlabel, girth, graph, graphical, gsimp, gunion, head, icosahedron, incidence, incident, indegree, induce, isplanar, maxdegree, mincut, mindegree, neighbors, new, octahedron, outdegree, path, petersen, random, rank, rankpoly, shortpathtree, show, shrink, span, spanpoly, spantree, tail, tetrahedron, tuttepoly, vdegree, vertices, void, vweight] (7.2.1.1)

```
> G:=void(6):addedge([[1,2],[2,4],[1,3],[3,6],[2,6],[1,5]],G)
;
tail(e1,G);head(e1,G);
```
e1, e2, e3, e4, e5, e6

1

2 (7.2.1.2)

```
> indegree(2,G);outdegree(2,G);
```
1

2 (7.2.1.3)

```
> addedge([1,2],G);tail(%,G);head(%%,G);
```

e7

1

2 (7.2.1.4)

```
> show(G);
```

$table([\_Bicomponents = \_Bicomponents, \_Vweight = table(sparse, [\,]), \_Ends = table([e3 = (\{1, 3\}), e6 = (\{1, 5\}), e2 = (\{2, 4\}), e5 = (\{2, 6\}), e4 = (\{3, 6\}), e1 = (\{1, 2\}), e7 = (\{1, 2\})]), \_Tail = table([e3 = 1, e6 = 1, e2 = 2, e5 = 2, e4 = 3, e1 = 1, e7 = 1]), \_Econnectivity = \_Econnectivity, \_Vertices = (\{1, 2, 3, 4, 5, 6\}), \_EdgeIndex = table(symmetric, [(2, 6) = \{e5\}, (3, 6) = \{e4\}, (1, 2) = (\{e1, e7\}), (1, 5) = \{e6\}, (1, 3) = \{e3\}, (2, 4) = \{e2\}]), \_Countcuts = \_Countcuts, \_Counttrees = \_Counttrees, \_Head = table([e3 = 3, e6 = 5, e2 = 4, e5 = 6, e4 = 6, e1 = 2, e7 = 2]), \_Neighbors = table([1 = (\{2, 3, 5\}), 2 = (\{1, 4, 6\}), 3 = (\{1, 6\}), 5 = \{1\}, 4 = \{2\}, 6 = (\{2, 3\})]), \_Status = (\{MULTIGRAPH, DIRECTED\}), \_Emaxname = 7, \_Eweight = table([e3 = 1, e6 = 1, e2 = 1, e5 = 1, e4 = 1, e1 = 1, e7 = 1]), \_Edges = (\{e1, e2, e3, e4, e5, e6, e7\})])$ (7.2.1.5)

▶ ***7.2.2. Példa.***

▶ ***->7.2.3. Feladat.***

▶ ***->7.2.4. Feladat.***

▶ ***->7.2.5. Feladat.***

▶ ***7.2.6. Feladat.***

▶ ***7.2.7. Feladat.***

▶ ***7.2.8. Feladat.***

▶ ***7.2.9. Irányított gráfok izomorfiája.***

▼ ***7.2.10. Példák.***

▼ ***7.2.11. Véges gráfok éllistás ábrázolása.***

▶ ***7.2.12. Irányított részgráf.***

▶ ***7.2.13. Irányított séták, vonalak, utak és körök.***

▼ ****7.2.14. Topologikus rendezés.***

▶ ***7.2.15. Erős összefüggőség.***

▶ ***7.2.16. Irányított fa.***

7.2.17. Kőnig-lemma.

***7.2.18. Kupac.**

***7.2.19. Kupacrendezés.**

***7.2.20. B-fa.**

***7.2.21. Oszd meg és uralkodj.**

***7.2.22. Dijksra modszere.**

***7.2.23. Megjegyzés.**

***7.2.24. Dinamikus programozás.**

***7.2.25. Feladat.**

***7.2.26. Feladat.**

***7.2.27. Feladat.**

***7.2.28. Feladat.**

***7.2.29. Szélességi bejárás.**

***7.2.30. Mélységi bejárás.**

***7.2.31. Feladat: Warshall-Floyd-algoritmus.**

***7.2.32. Feladat.**

***7.2.33. Feladat: erős komponensek meghatározása.**

***7.2.34. PERT.**

***7.2.35. Feladat.**

***7.2.36. Folyamprobléma.**

***7.2.37. Lemma.**

***7.2.38. Ford-Fulkerson-tétel (maximális folyam, minimális vágás tétel).**

***7.2.39. Edmonds-Karp-heurisztika.**

**7.2.40. Dinic módszere.*

**7.2.41. Feladat.*

**7.2.42. Feladat.*

**7.2.43. Feladat: általánosított folyamprobléma.*

**7.2.44. Feladat: Menger tétele elvágó élhalmazra.*

**7.2.45. Feladat: Menger tétele elvágó csúcshalmazra.*

**7.2.46. Feladat: König tétele.*

**7.2.47. Gráfok rajzolhatósága.*

**7.2.48. Állítás.*

**7.2.49. Segédtétel.*

**7.2.50. Tétel.*

->7.2.51. Feladat.

->7.2.52. Feladat.

**7.2.53. Tartományok.*

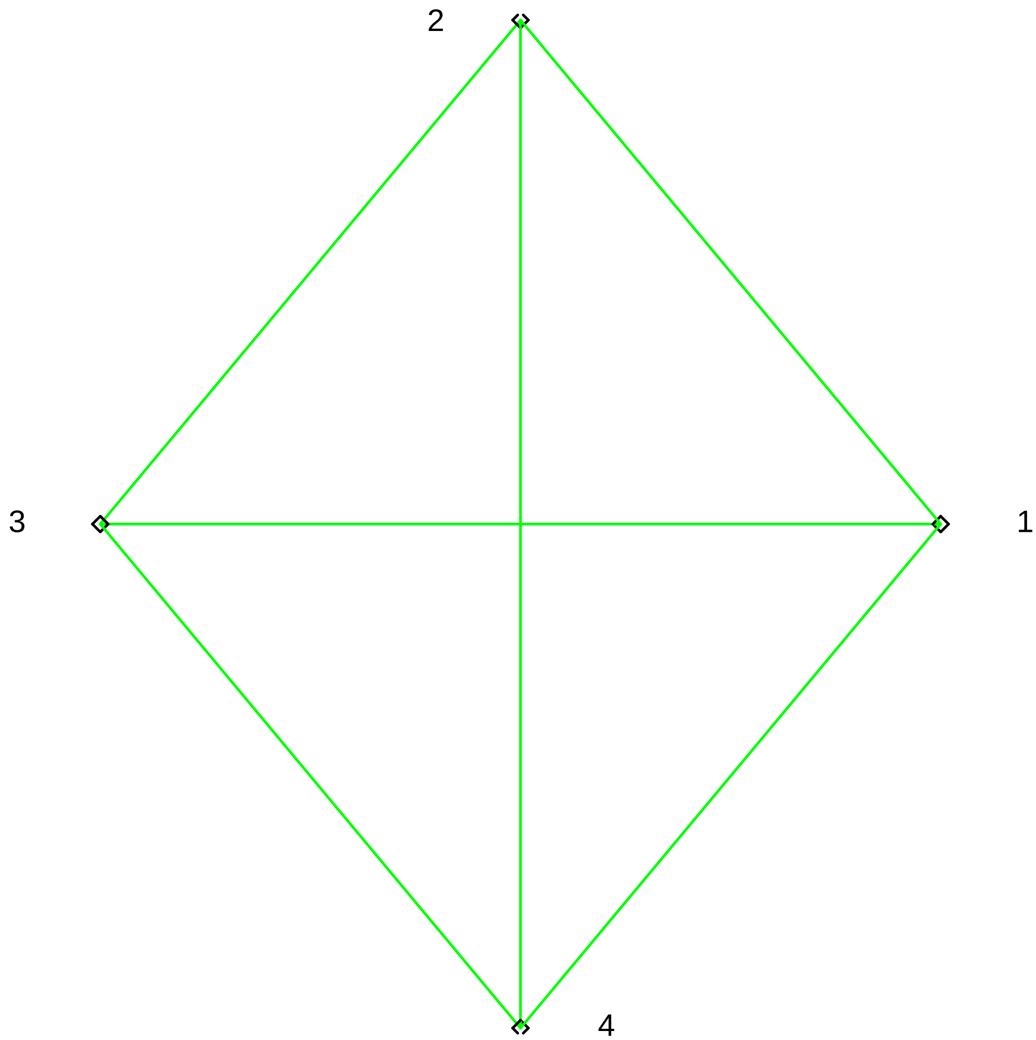
**7.2.54. Euler tétele.*

**7.2.55. Megjegyzés.*

**7.2.56. Gráfok topologikus izomorfizmusa.*

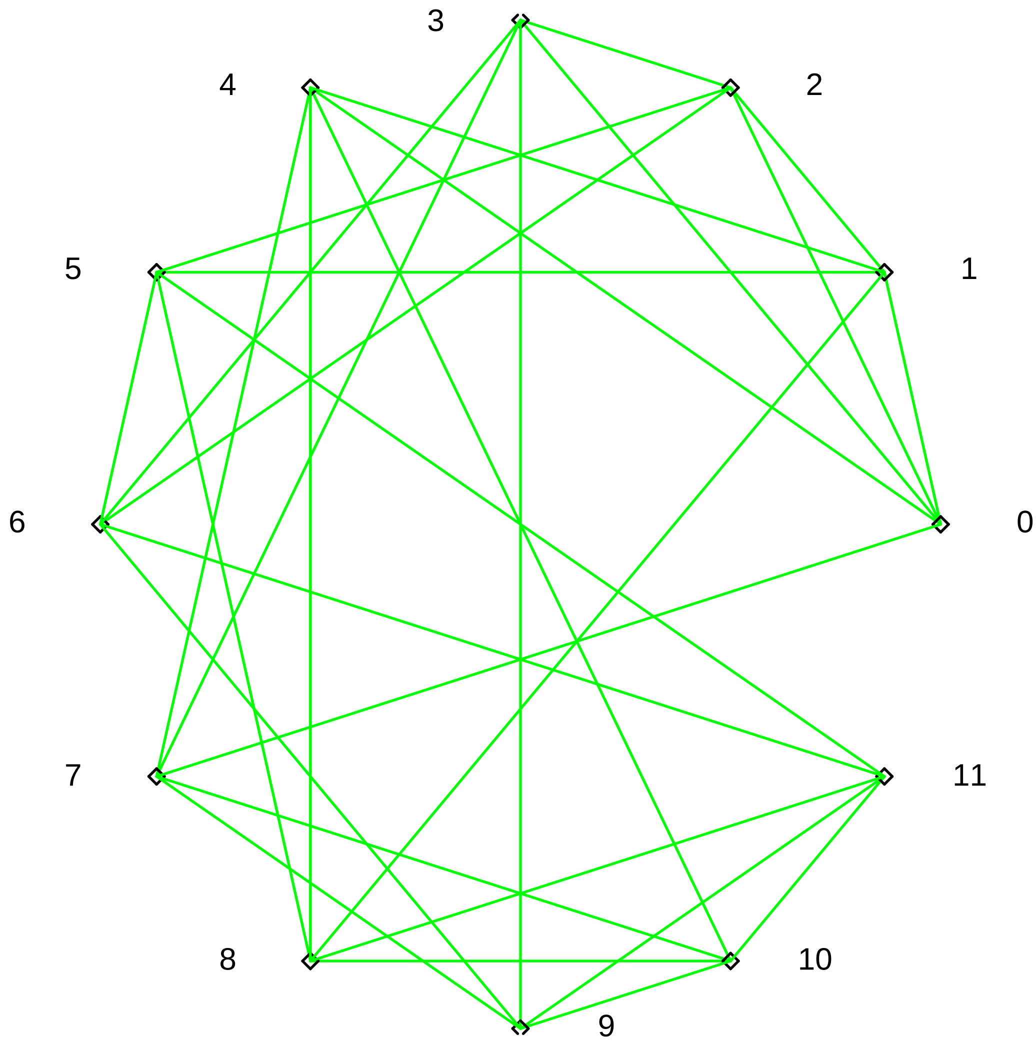
**7.2.57. Kuratowski tétele.*

```
> G:=tetrahedron():draw(G);isplanar(G);
```



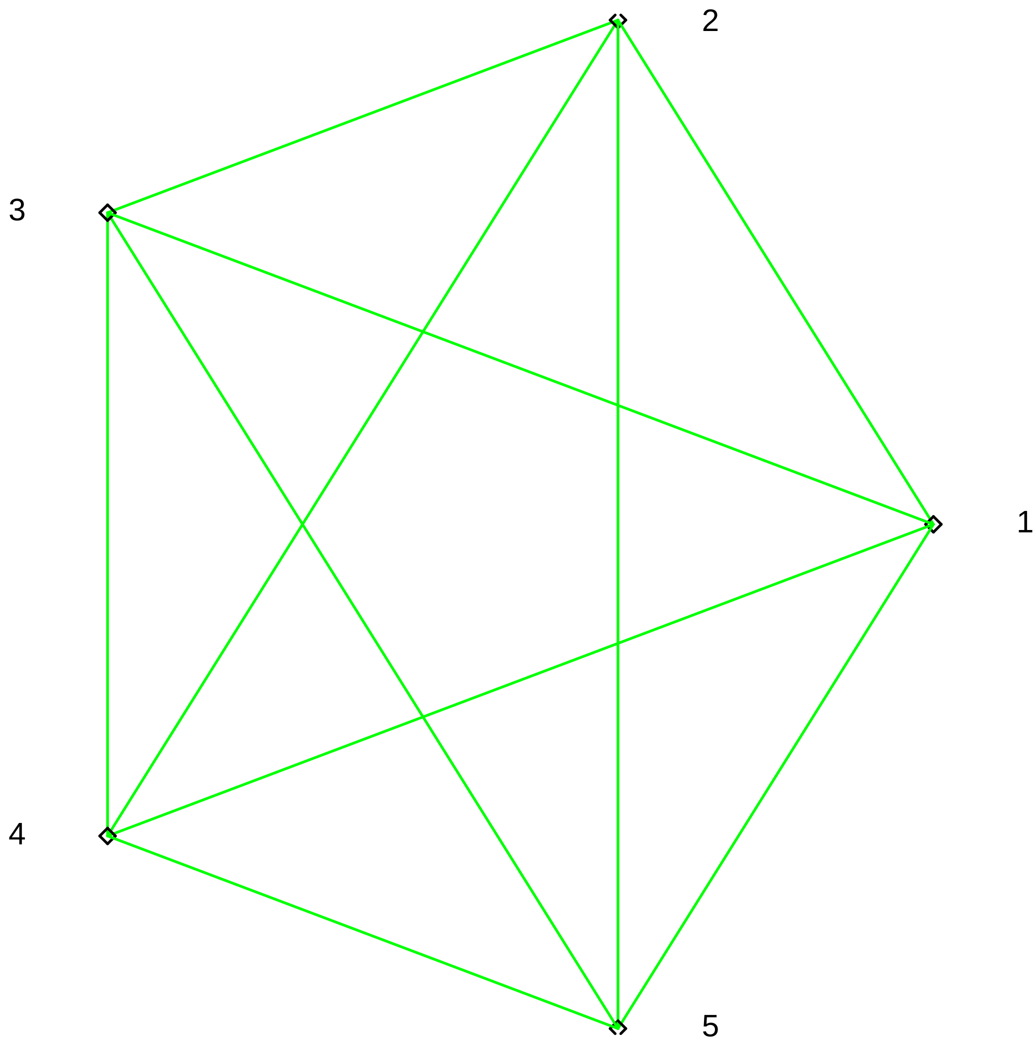
$$true \qquad (7.2.57.1)$$

```
> G:=icosahedron():draw(G);isplanar(G);
```



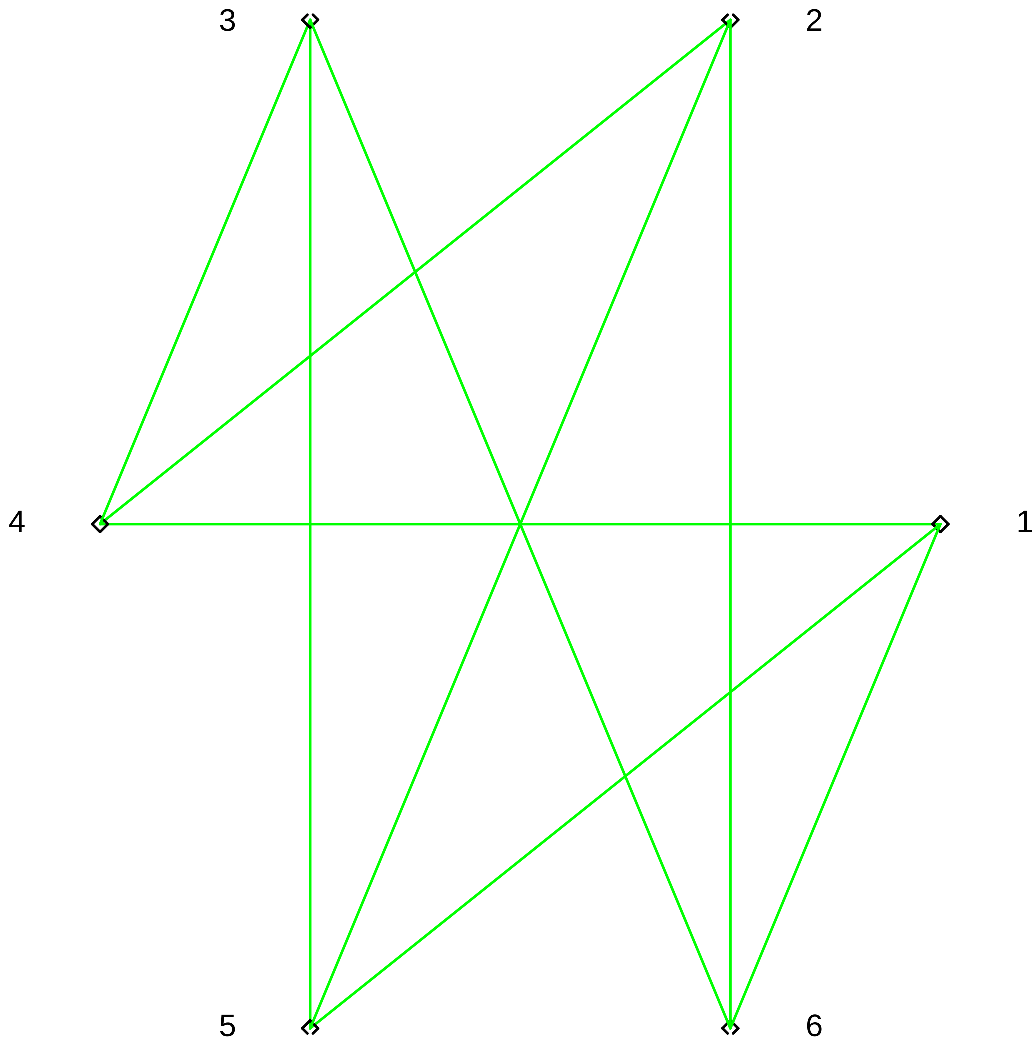
true (7.2.57.2)

```
> G:=complete(5):draw(G);isplanar(G);
```



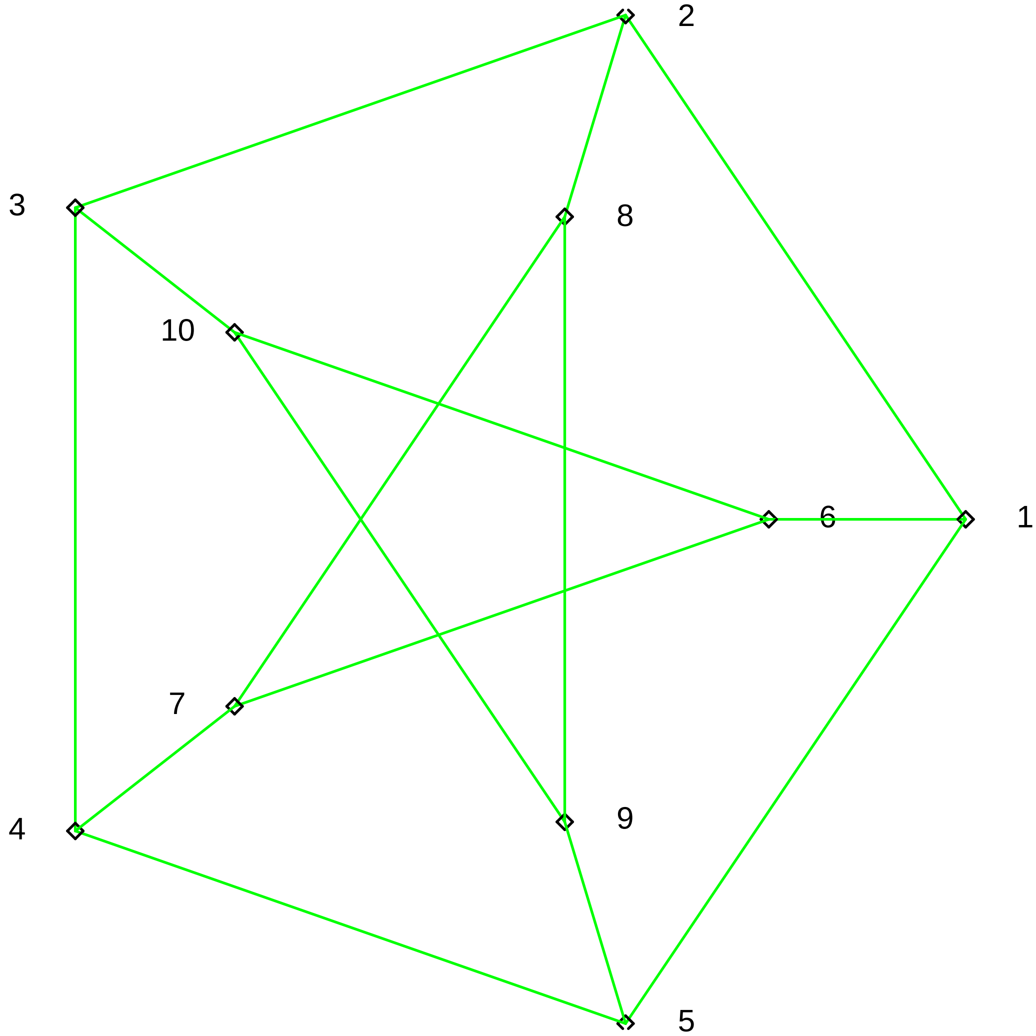
false (7.2.57.3)

```
> G:=complete(3,3):draw(G);isplanar(G);
```



false (7.2.57.4)

```
> G:=petersen():draw(G);isplanar(G);
```



false (7.2.57.5)

7.2.58. Feladat.

7.2.59. Feladat.

7.2.60. Feladat.

7.2.61. Feladat.

7.2.62. Feladat.

7.2.63. Kromatikus szám.

–>7.2.64. Feladat.

7.2.65. Feladat.

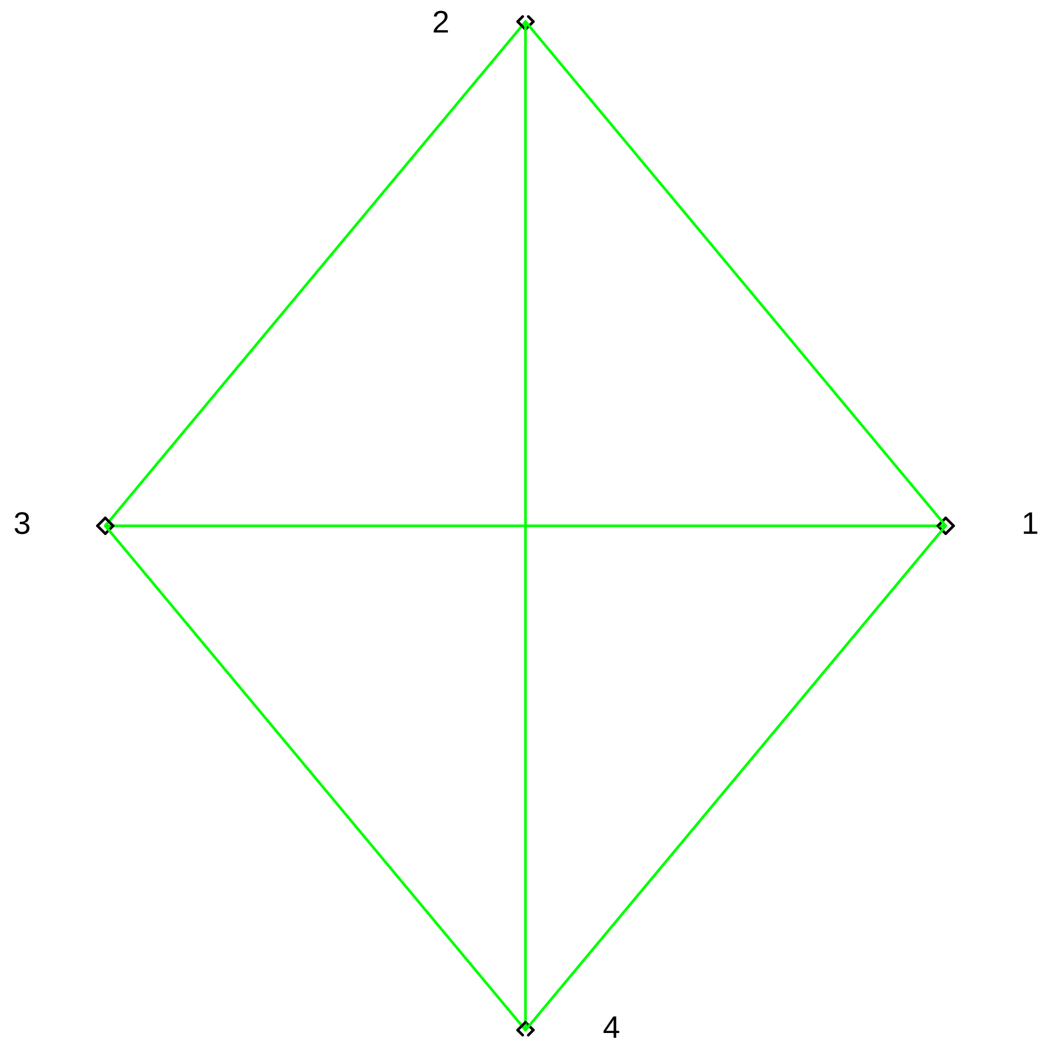
7.2.66. Feladat.

7.2.67. Feladat.

7.2.68. Feladat.

7.2.69. Gráfok mátrixai.

```
> G:=tetrahedron():draw(G);adjacency(G);incidence(G);
```



$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \tag{7.2.69.1}$$

7.2.70. Feladat.

7.2.71. További feladatok.

8. Algebra

9. Kódolás

10. Algoritmusok