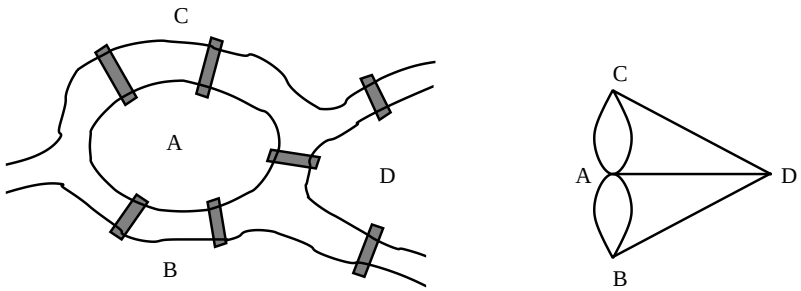
1η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

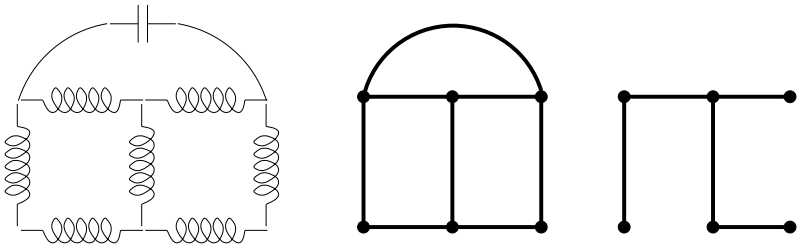
- Βασικοί ορισμοί και συμβολισμοί
- Μορφές γραφημάτων
  - Μηδενικό γράφημα
  - Τετριμμένο γράφημα
  - Πλήρες γράφημα
  - Τυχαίο γράφημα
- Υπογραφήματα
- Συμπλήρωμα
- Βαθμοί
  - Λήμμα χειραψίας
  - Ο αλγόριθμος Havel - Hakimi

Εισαγωγή – ιστορικό

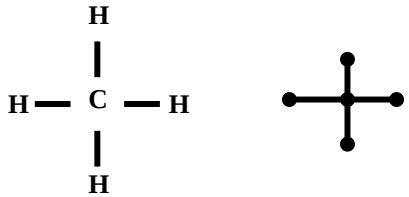


Euler (1736): Γέφυρες του Königsberg

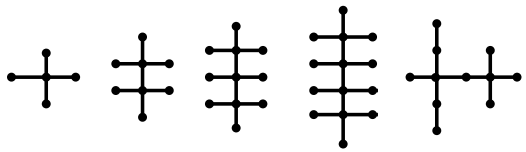
Μπορεί κάποιος να περάσει ακριβώς μια φορά από κάθε γέφυρα;



Kirchhoff (1847): Γενετικό δένδρο



Cayley (1857): Πλήθος κορεσμένων υδρογονάνθρακων C_nH_{2n+2}



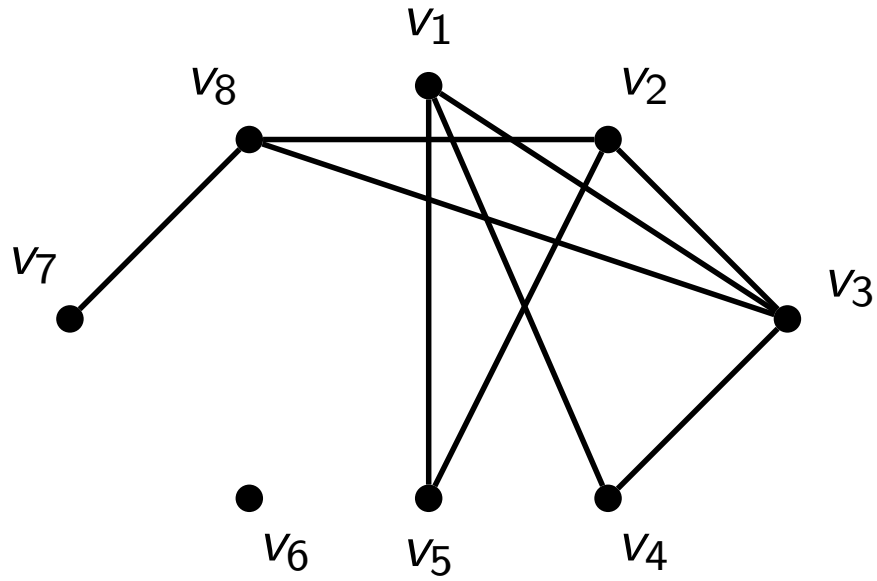
Μερικές από τις εφαρμογές της Θεωρίας Γραφημάτων :

- Πληροφορική (Δένδρα, Δυαδικά δένδρα, Διατεταγμένα δένδρα, Διάτρεξη (διάσχιση) δένδρων, Προγραμματισμός, Συνδεσμολογία κ.λπ.).
- Αλγόριθμοι (Αλγόριθμοι γραφημάτων, Αναζήτηση πρώτα κατά πλάτος, Αναζήτηση πρώτα κατά βάθος, Τοπολογική διάταξη, Κατάταξη έργων με προθεσμίες κ.λπ.).
- Διοίκηση Επιχειρήσεων (Οργανογράμματα, Κεντρικά σημεία κ.λπ.).
- Οδοποιΐα (Οδικά δίκτυα - χωρητικότητα - μέγιστη ροή, Σηματοδότηση δρόμων).
- Υδραυλικά (Δίκτυα - χωρητικότητα -μέγιστη ροή).
- Ιστορία - Κοινωνιολογία (Γενεαλογικά δένδρα, Φιλία (γραφήματα δεσμών), Έρωτας (γραφήματα τόξων)).

Γραφήματα δεσμών

Κάθε δυάδα $G = (V(G), E(G))$, ή (V, E) όπου V είναι ένα μη κενό σύνολο και E είναι ένα σύνολο από (μη διατεταγμένα) ζεύγη $\{v, u\}$, με $v, u \in V$ ονομάζεται **γράφημα δεσμών**, ή **απροσανατόλιστο γράφημα**.
Τα στοιχεία του V καλούνται **κορυφές**, ή **σημεία**, ή **κόμβοι** (vertices, points), ενώ τα στοιχεία του E καλούνται **δεσμοί**, ή **γραμμές**, ή **χορδές**, ή **πλευρές**, ή **ακμές** (edges, lines). Αν $v, u \in V$, με $\{v, u\} \in E$, τότε τα v, u ονομάζονται **άκρα** του δεσμού $\{v, u\}$ και είναι **γειτονικά**. Λέμε επίσης τότε ότι τα v, u **ανήκουν** στο $\{v, u\}$, ή ότι ο $\{v, u\}$ **ενώνει** ή **περιέχει** τα v, u.
Θα ασχοληθούμε εδώ με **πεπερασμένα γραφήματα**, (δηλαδή $|V| \in \mathbb{N}^*$). Το E μπορεί να είναι $\emptyset$. Συχνά γράφουμε $|V| = n$ και $|E| = m$. Ο πληθάριθμος $|V|$ (αντ. $|E|$) ονομάζεται **τάξη** (order) (αντ. **μέγεθος** (size)) του γραφήματος.

Παράδειγμα Η δυάδα $G = (V, E)$ όπου $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$ και $E = \{\{v_1, v_3\}, \{v_1, v_4\}, \{v_1, v_5\}, \{v_2, v_3\}, \{v_2, v_5\}, \{v_2, v_8\}, \{v_3, v_4\}, \{v_3, v_8\}, \{v_7, v_8\}\}$ είναι ένα γράφημα δεσμών. Η γραφική του απεικόνιση είναι η ακόλουθη:

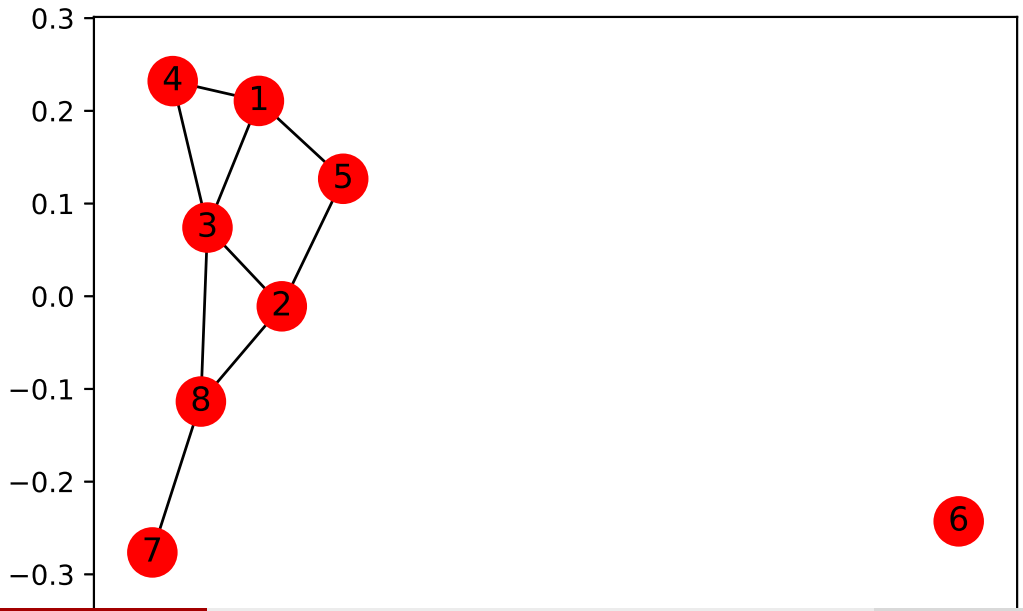


Στο επόμενο πρόγραμμα χρησιμοποιούμε την βιβλιοθήκη networkx της Python για να ορίσουμε το γράφημα G του πρώτου παραδείγματος.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph() #Create an empty graph

V = [1,2,3,4,5,6,7,8] #V is the set of vertices of G
E = [[1,3],[1,4],[1,5],[2,3],[2,5],[2,8],[3,4],[3,8],[7,8]] #E is
    the set of edges of G

G.add_nodes_from(V)
G.add_edges_from(E)

print("G has order |V(G)|=",G.order(),"and size |E(G)|=",G.size())
print("V(G):",G.nodes()) #Print the nodes of G
print("E(G):", G.edges()) #Print the edges of G
for v in G:
    print("The neighbors of", v, "are:", list(G.neighbors(v)))

nx.draw_networkx(G) #Draw the graph G
plt.savefig("lect01a.eps") #Save the drawing of G
plt.show() #Show the drawing of G on screen
```

Output:

```
G has order |V(G)|= 8 and size |E(G)|= 9
V(G): [1, 2, 3, 4, 5, 6, 7, 8]
E(G): [(1, 3), (1, 4), (1, 5), (2, 3), (2, 5), (2, 8), (3, 4), (3,
    8), (7, 8)]
The neighbors of 1 are: [3, 4, 5]
The neighbors of 2 are: [3, 5, 8]
The neighbors of 3 are: [1, 2, 4, 8]
The neighbors of 4 are: [1, 3]
The neighbors of 5 are: [1, 2]
The neighbors of 6 are: []
The neighbors of 7 are: [8]
The neighbors of 8 are: [2, 3, 7]
```



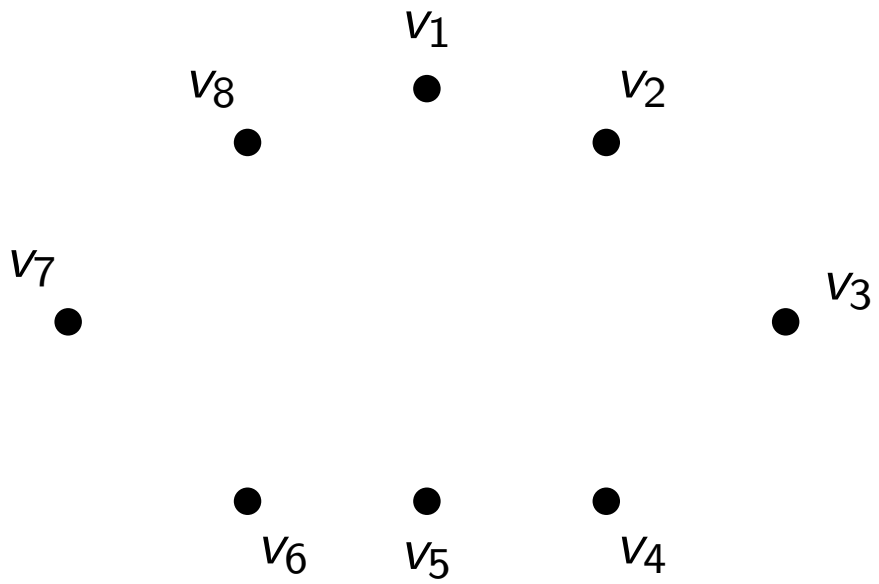
Αν οι v, u ταυτίζονται έχουμε ένα **βρόχο**.

Παρατήρηση. Δεδομένου ότι σε ένα σύνολο επιτρέπεται μια μόνο εμφάνιση κάθε στοιχείου του, από τον ορισμό του γραφήματος δεσμών προκύπτει ότι σε αυτό δεν επιτρέπονται ούτε βρόχοι ούτε πολλαπλοί δεσμοί που να συνδέουν το ίδιο ζεύγος κορυφών. Τα γραφήματα αυτά ονομάζονται **απλά γραφήματα** και με τέτοια θα ασχοληθούμε, εκτός αν αναφερθεί ρητά το αντίθετο.

Μορφές γραφημάτων

- **Μηδενικό γράφημα** ονομάζεται ένα γράφημα $G = (V, E)$ με $E = \emptyset$.

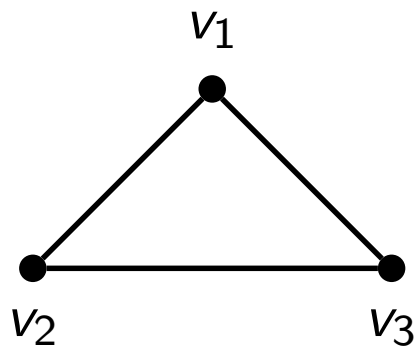
  Παράδειγμα



- **Τετριμμένο γράφημα** ονομάζεται ένα γράφημα $G = (V, E)$ με $|V| = 1$.

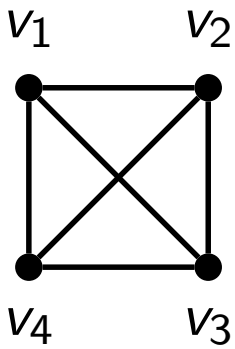
- **Πλήρες γράφημα** ονομάζεται ένα γράφημα $G = (V, E)$ τέτοιο ώστε $\forall u, v \in V$ με $v \neq u$ ισχύει ότι $\{v, u\} \in E$.
  Παρατήρηση. Το πλήρες γράφημα με n κόμβους συμβολίζεται με K_n.
  Παράδειγμα Το γράφημα K_3 είναι το

K_3



ενώ το γράφημα K_4 είναι το :

K_4



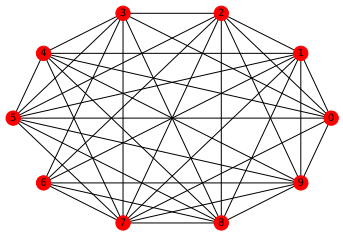
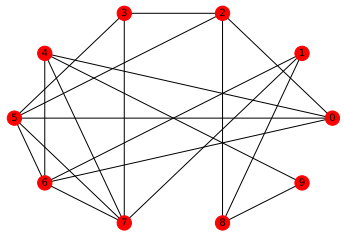
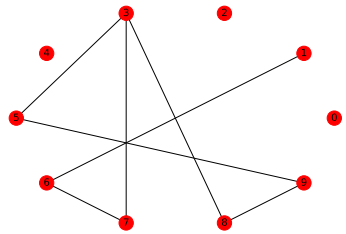
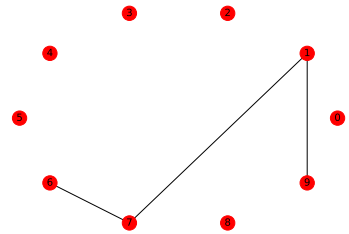
Στην βιβλιοθήκη networkx το πλήρες γράφημα με n κορυφές κατασκευάζεται χρησιμοποιώντας την μέθοδο `complete_graph(n)`, ή χρησιμοποιώντας τις επόμενες εντολές:

```
import networkx as nx
import matplotlib.pyplot as plt

n = 7 #number of vertices

Kn = nx.complete_graph(n)
nx.draw_circular(Kn,with_labels=True)
plt.show()

Kn = nx.Graph()
Kn.add_nodes_from(range(1,n+1))
for i in range(1,n+1):
    for j in range(i+1,n+1):
        Kn.add_edge(i,j)
nx.draw_circular(Kn,with_labels=True)
plt.show()
```

- **Τυχαίο γράφημα**: Κάποιες φορές καλούμαστε να δοκιμάσουμε αλγορίθμους ή ιδέες μας πάνω σε διάφορα παραδείγματα γραφημάτων. Μπορούμε να φτιάχνουμε τέτοια ‘‘τυχαία’’ παραδείγματα χρησιμοποιώντας έτοιμες μεθόδους της βιβλιοθήκης networkx ή γράφοντας δικές μας μεθόδους, όπως στα επόμενα παραδείγματα.
  Η πιο απλή ιδέα κατασκευής ενός τυχαίου γραφήματος με n κορυφές είναι το **μοντέλο των Erdős - Renyi** όπου για κάθε ζεύγος κορυφών επιλέγουμε να δημιουργήσουμε τον δεσμό που τις συνδέει με πιθανότητα p.

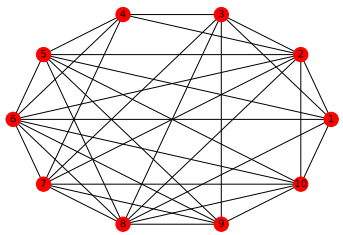
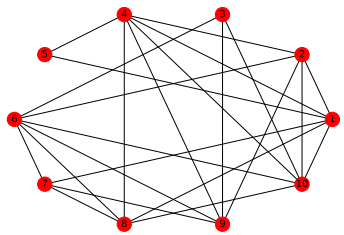
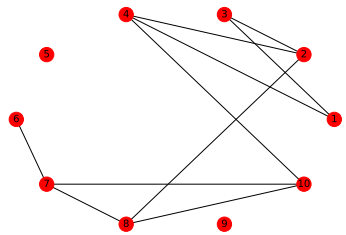
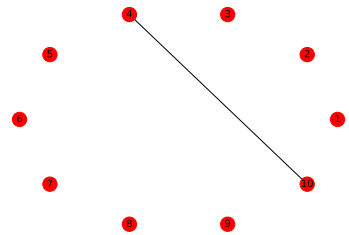
Ένα τέτοιο γράφημα προκύπτει χρησιμοποιώντας την μέθοδο gnp_random_graph(n,p):

```
import networkx as nx
import matplotlib.pyplot as plt

def random_gnp_graph(n,p,name):
    R = nx.gnp_random_graph(n,p)
    nx.draw_circular(R,with_labels=True)
    plt.savefig(name+".eps")
    plt.show()

random_gnp_graph(10,0.8,"R1")
random_gnp_graph(10,0.5,"R2")
random_gnp_graph(10,0.2,"R3")
random_gnp_graph(10,0.1,"R4")
```

			
$n = 10,\ p = 0.8$	$n = 10,\ p = 0.5$	$n = 10,\ p = 0.2$	$n = 10,\ p = 0.1$

Μια απλή υλοποίηση της μεθόδου gnp_random_graph(n,p):

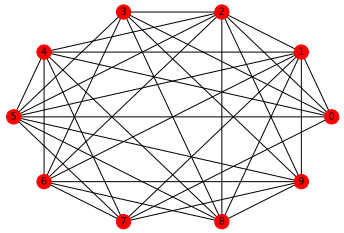
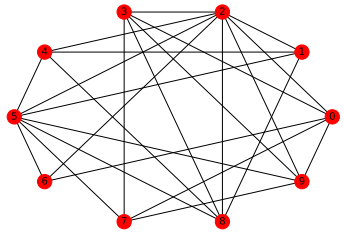
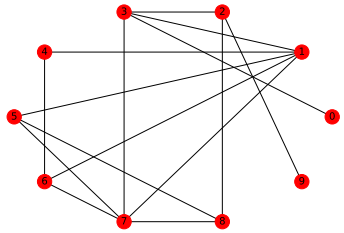
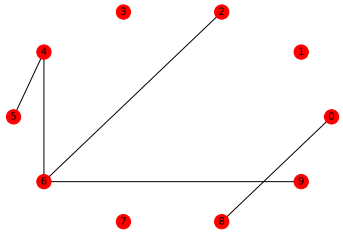
```
import networkx as nx
import matplotlib.pyplot as plt
import random #random numbers

def random_gnp_graph2(n,p,name):
    R = nx.Graph()
    R.add_nodes_from(range(1,n+1))
    for i in range(1,n+1):
        for j in range(i+1,n+1):
            if random.uniform(0,1) <= p:
                R.add_edge(i,j)
    nx.draw_circular(R,with_labels=True)
    plt.savefig(name+".eps")
    plt.show()

random_gnp_graph2(10,0.8,"R9")
random_gnp_graph2(10,0.5,"R10")
random_gnp_graph2(10,0.2,"R11")
random_gnp_graph2(10,0.1,"R12")
```

			
$n = 10,\ p = 0.8$	$n = 10,\ p = 0.5$	$n = 10,\ p = 0.2$	$n = 10,\ p = 0.1$

Παρατήρηση: Επειδή κάθε ένας από τους $\binom{n}{2}$ πιθανούς δεσμούς επιλέγεται με πιθανότητα p έπεται ότι το τυχαίο γράφημα που προκύπτει κατά μέσο όρο αναμένεται να έχει $p\binom{n}{2}$ δεσμούς.

Στην περίπτωση όπου θέλουμε το τυχαίο γράφημα να έχει n κορυφές και ακριβώς m δεσμούς μπορούμε να χρησιμοποιήσουμε την μέθοδο gnm_random_graph(n,m):

```
import networkx as nx
import matplotlib.pyplot as plt

def random_gnm_graph(n,m,name):
    R = nx.gnm_random_graph(n,m) #0 <= m <= n(n-1)/2
    nx.draw_circular(R,with_labels=True)
    plt.savefig(name+".eps")
    plt.show()

random_gnm_graph(10,35,"R5")
random_gnm_graph(10,25,"R6")
random_gnm_graph(10,15,"R7")
random_gnm_graph(10,5,"R8")
```

			
$n = 10,\ m = 35$	$n = 10,\ m = 25$	$n = 10,\ m = 15$	$n = 10,\ m = 5$

Παρατήρηση (*): Μια υλοποίηση της μεθόδου `gnm_random_graph(n,m)` μπορεί να γίνει κατασκευάζοντας ένα τυχαίο υποσύνολο του $[\binom{n}{2}]$ με m στοιχεία.

Πηγές ανοιχτών δεδομένων για γραφήματα

Υπάρχουν αρκετοί ιστότοποι με συλλογές ανοιχτών δεδομένων που αφορούν γραφήματα τα οποία εμφανίζονται σε πραγματικές καταστάσεις. Ένας τέτοιος ιστότοπος είναι το Network Repository (`https://networkrepository.com/`):

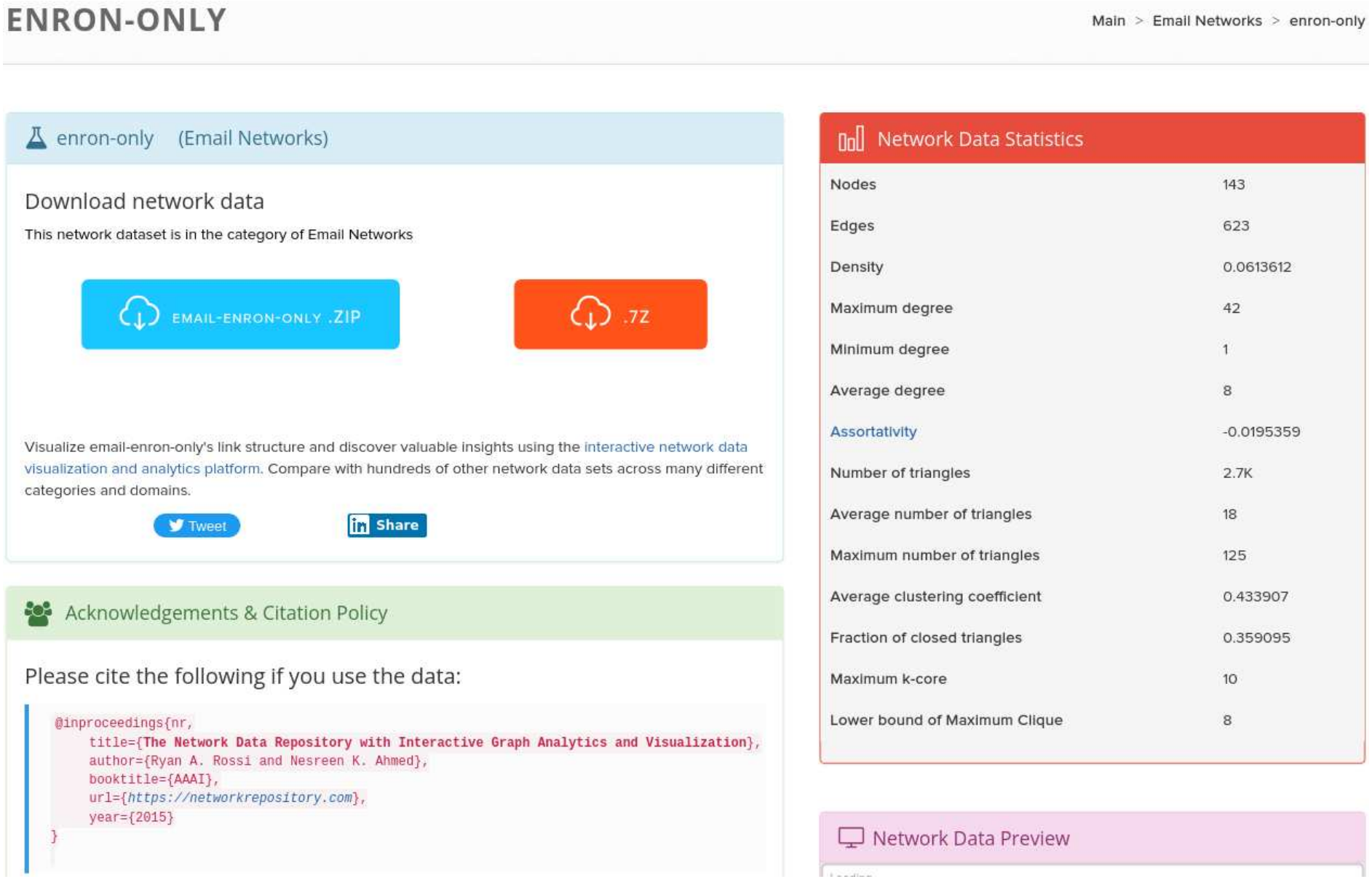
Συνήθως, τα δεδομένα που αφορούν τα γραφήματα είναι διαθέσιμα σε μορφή αρχείου κειμένου που περιέχει λίστα με δεσμούς του γραφήματος (δύο αριθμοί σε κάθε γραμμή, οι αριθμοί δηλώνουν τις ετικέτες των κορυφών που ενώνει ο δεσμός).
Η βιβλιοθήκη `networkx` έχει την μέθοδο `read_edgelist('filename.edges')` για την ανάγνωση του γραφήματος από το αρχείο `filename.edges` το οποίο περιέχει την λίστα των δεσμών του γραφήματος.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.read_edgelist('realgraphs/email-EU.edges')

print("Number of nodes:", G.order(), "Number of edges:", G.size())
```

Output:

```
Number of nodes: 32430 Number of edges: 54397
```

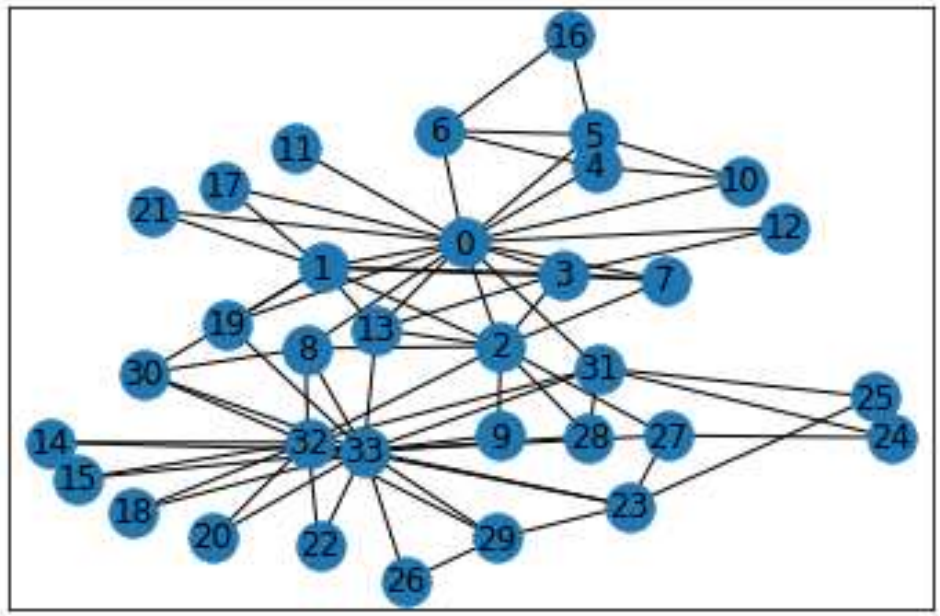
PE



Σχήμα: Παράδειγμα από το Network Repository

Επίσης, αρκετά δημοφιλής είναι ο τύπος αρχείου `mtx` (matrix market file) το οποίο εκτός από την λίστα των δεσμών περιέχει το πλήθος των δεσμών και τον συνολικό αριθμό των κορυφών, επίσης μπορεί να περιέχει σχόλια. Για την ανάγνωση αρχείων `mtx` μπορεί να χρησιμοποιηθεί η μέθοδος `mmread('filename.mtx')` της βιβλιοθήκης `scipy.io`.

```
import networkx as nx
import matplotlib.pyplot as plt
from scipy.io import mmread

a = mmread('realgraphs/soc-
    karate.mtx')
G = nx.Graph(a)

pos = nx.layout.
    kamada_kawai_layout(G)
nx.draw_networkx(G,pos)

plt.show()
```

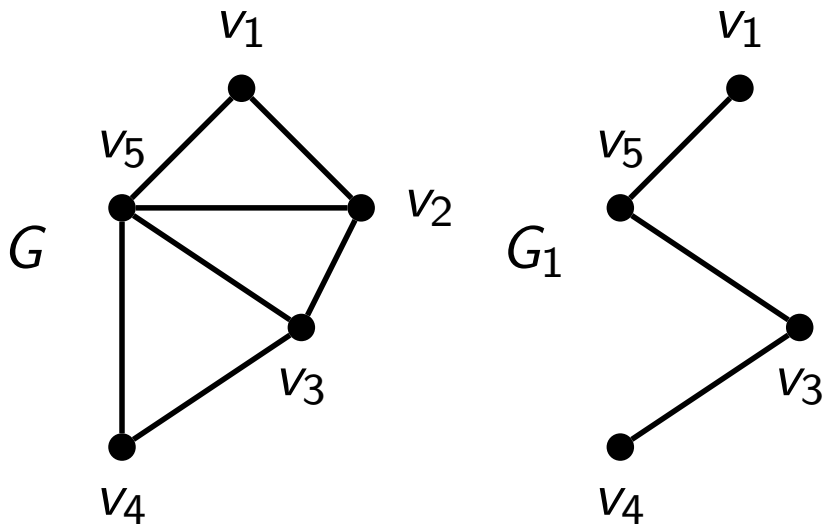


Output:

Υπογραφήματα

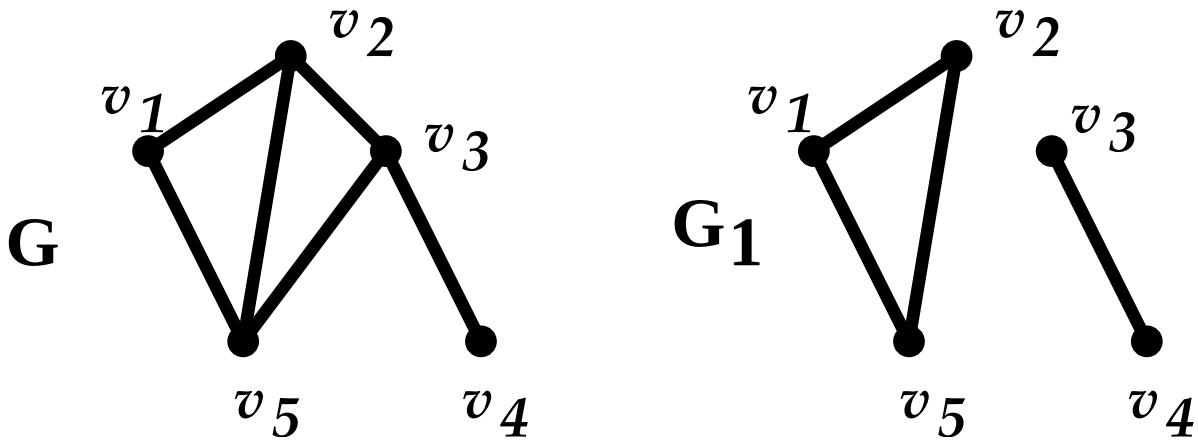
- **Υπογράφημα** του $G = (V, E)$ ονομάζεται ένα γράφημα $G_1 = (V_1, E_1)$ με $V_1 \subseteq V$ και $E_1 \subseteq E$.

Παράδειγμα

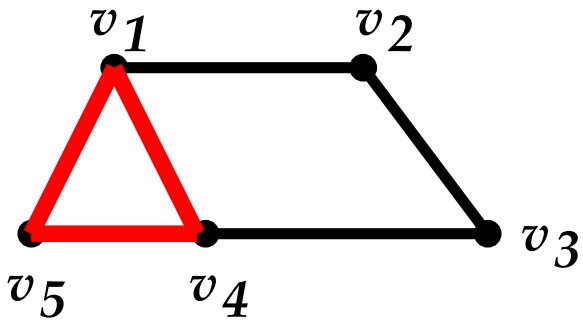


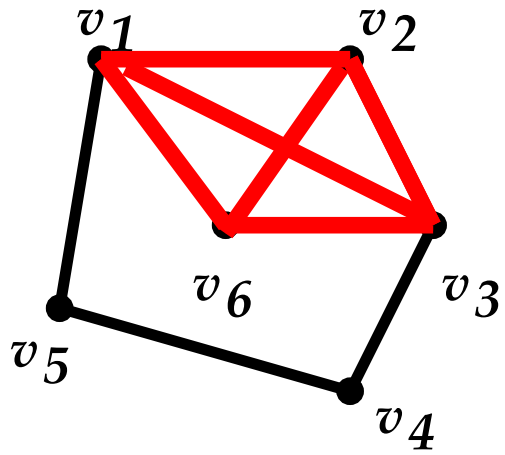
- **Γενετικό** (ή **γεννητικό**, ή **μερικό**) **γράφημα**, ή **γράφημα ζεύξης** του $G = (V, E)$ ονομάζεται ένα γράφημα $G_1 = (V_1, E_1)$ με $V_1 = V$ και $E_1 \subseteq E$.

  Παράδειγμα



- **Κλίκα** ενός γραφήματος ονομάζεται κάθε πλήρες υπογράφημά του. **Μέγιστη κλίκα** ενός γραφήματος ονομάζεται κάθε κλίκα του με το μέγιστο δυνατό αριθμό κόμβων.
  Παραδείγματα

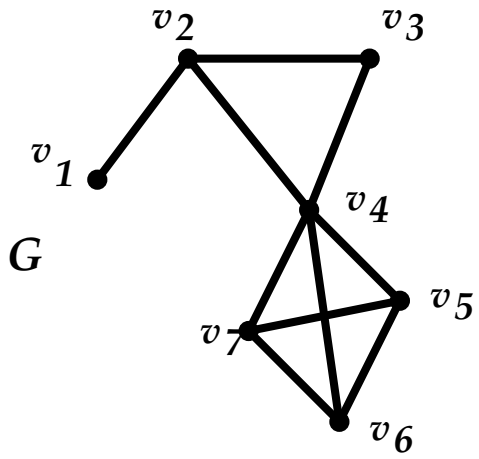


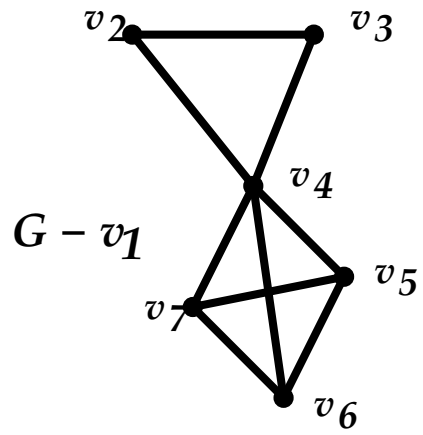


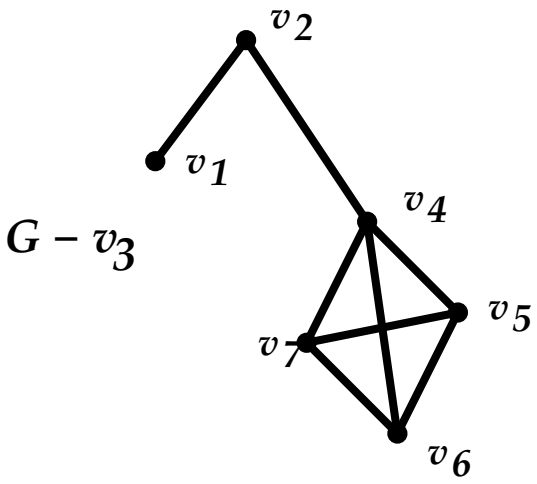
Οι μέγιστες κλίκες των δύο παρακάτω γραφημάτων είναι οι K_3 και K_4 αντίστοιχα.

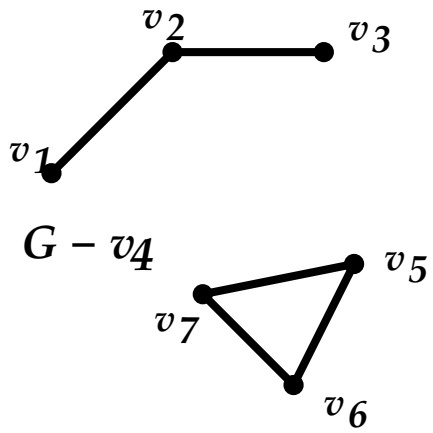
- Αν $G = (V, E)$ και $v \in V$, $e \in E$ ορίζουμε τα υπογραφήματα $\mathbf{G - v}$, $\mathbf{G - e}$ ως εξής:
  $V(G - v) = V \setminus \{v\}$, $E(G - v) = E \setminus \{e_i \in E : v \in e_i\}$, ενώ
  $V(G - e) = V$, $E(G - e) = E \setminus \{e\}$.

  Παραδείγματα





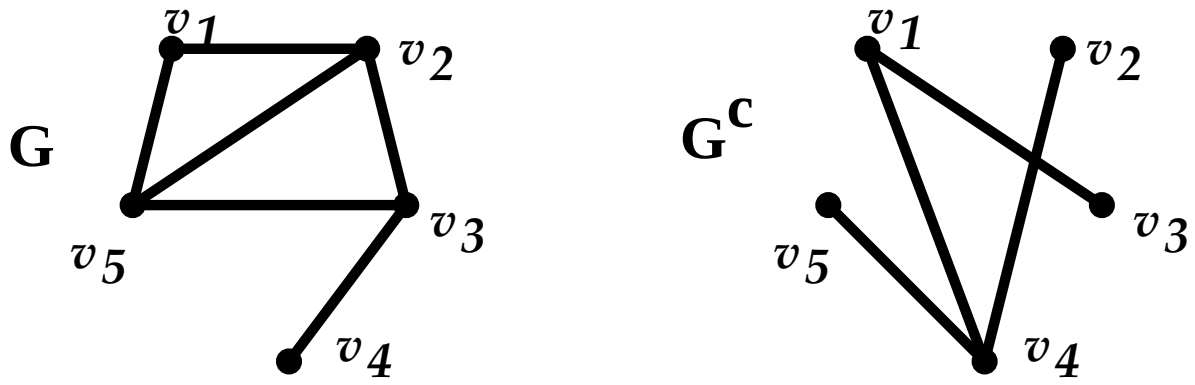




Συμπλήρωμα

Συμπλήρωμα G^c του $G = (V, E)$ με $|V| = n$ ονομάζεται το γράφημα G^c ή $\bar{G}$ ή $G^* = (V, E^c)$ με $E^c = E(K_n) \setminus E(G)$.

Παράδειγμα



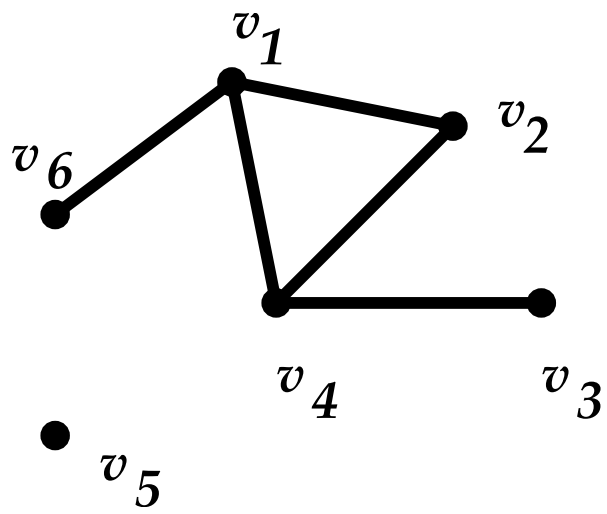
Βαθμοί

Αν $G = (V, E)$, για κάθε $v \in V$ ορίζουμε

$$\Gamma_G(v) = \{u \in V(G) : \{v, u\} \in E(G)\}.$$

Τότε $|\Gamma_G(v)|$ ονομάζεται **βαθμός** του κόμβου v και συμβολίζεται με $d_G(v)$, ή $d(v)$, ή $\deg(v)$.
Δηλαδή, βαθμός του v στο G, λέγεται το πλήθος των δεσμών του G των οποίων ο v είναι άκρο, ή ισοδύναμα ο αριθμός των γειτόνων του v.

Παράδειγμα



Στο παραπάνω γράφημα, οι κόμβοι του έχουν τους ακόλουθους βαθμούς:

$$d(v_1) = d(v_4) = 3,$$
$$d(v_2) = 2,$$
$$d(v_3) = d(v_6) = 1,$$
$$d(v_5) = 0.$$

Κάθε κόμβος βαθμού μηδέν λέγεται **μεμονωμένος** κόμβος.
Ένα γράφημα $G = (V, E)$ λέγεται d-**κανονικό** αν $d_G(v) = d$, για κάθε $v \in V$.
Ο **ελάχιστος** (αντ. **μέγιστος**) **βαθμός** των κορυφών ενός γραφήματος G θα συμβολίζεται με $\delta(G)$ (αντ. $\Delta(G)$).
Έστω $G = (V, E)$ με $V = \{v_1, v_2, \ldots, v_n\}$.
Ακολουθία βαθμών του G λέγεται η πεπερασμένη ακολουθία

$$(d(v_1), d(v_2), \ldots, d(v_n)).$$

Παράδειγμα Η ακολουθία βαθμών του παραπάνω γραφήματος είναι $(3, 3, 2, 1, 1, 0)$.
Παρατήρηση. Συνήθως γράφουμε την ακολουθία βαθμών ενός γραφήματος σε φθίνουσα σειρά.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph() #Create an empty graph

V = [1,2,3,4,5,6] #V is the set of vertices of G
E = [[1,2],[1,4],[1,6],[2,4],[3,4]] #E is the set of edges of G

G.add_nodes_from(V)
G.add_edges_from(E)

DegreeSeq = []
for v in G.nodes:
    DegreeSeq.append(G.degree(v))
    print("Vertex",v,"has degree",G.degree(v))
DegreeSeq.sort(reverse=True)
print("Degree sequence of G:", DegreeSeq)
```

Output:

```
Vertex 1 has degree 3
Vertex 2 has degree 2
Vertex 3 has degree 1
Vertex 4 has degree 3
Vertex 5 has degree 0
Vertex 6 has degree 1
Degree sequence of G: [3, 3, 2, 1, 1, 0]
```

Οι ακολουθίες βαθμών έχουν ορισμένους περιορισμούς, δηλαδή δεν αντιστοιχούν όλες οι ακολουθίες φυσικών αριθμών σε ακολουθίες βαθμών γραφημάτων. Για παράδειγμα, δεν υπάρχει γράφημα με 5 κορυφές το οποίο έχει ακολουθία βαθμών $(6, 4, 4, 4, 4)$ διότι σε κάθε γράφημα G με 5 κορυφές ισχύει ότι $\Delta(G) \leq 4$.

Μια ακολουθία φυσικών αριθμών $(d_1, d_2, \ldots, d_n)$ ονομάζεται **γραφική** (graphical) αν υπάρχει γράφημα G με ακολουθία βαθμών την ακολουθία $(d_1, d_2, \ldots, d_n)$.

Η μηδενική ακολουθία $(0, 0, \ldots, 0)$ μήκους n αντιστοιχεί στο μηδενικό γράφημα με n κορυφές.

Λήμμα χειραψίας

Πρόταση 1 (Λήμμα χειραψίας)

Σε κάθε απλό γράφημα δεσμών ισχύει ότι $\sum_{i=1}^{|V|} d(v_i) = 2|E|$.

Απόδειξη: Κάθε δεσμός του γραφήματος συνεισφέρει κατά 2 στο άθροισμα των βαθμών (λόγω των άκρων του). Άρα, οι $|E|$ δεσμοί που περιέχει το γράφημα θα δημιουργούν συνολικό άθροισμα βαθμών $2|E|$.

Πόρισμα 1

Σε κάθε απλό γράφημα δεσμών ο αριθμός των κόμβων με περιττό βαθμό είναι άρτιος.

Απόδειξη: Έστω $v_1, v_2, \ldots, v_{2k+1}$ οι (περιττοί σε πλήθος) κόμβοι με περιττό βαθμό. Τότε το άθροισμα των βαθμών των κόμβων αυτών θα ήταν περιττό (έστω Π) ως άθροισμα περιττού πλήθους περιττών αριθμών. Δεδομένου ότι το άθροισμα των βαθμών των κόμβων με άρτιο βαθμό είναι άρτιο (έστω A) ως άθροισμα άρτιων αριθμών, το συνολικό άθροισμα των βαθμών του γραφήματος θα ήταν $\Pi + A$: περιττός, το οποιο σύμφωνα με την Πρόταση 1 είναι άτοπο. Άρα το πλήθος των κόμβων με περιττό βαθμό είναι άρτιο.

Η επόμενη πρόταση δίνει μια αναγκαία και ικανή συνθήκη για το πότε μια ακολουθία είναι γραφική.

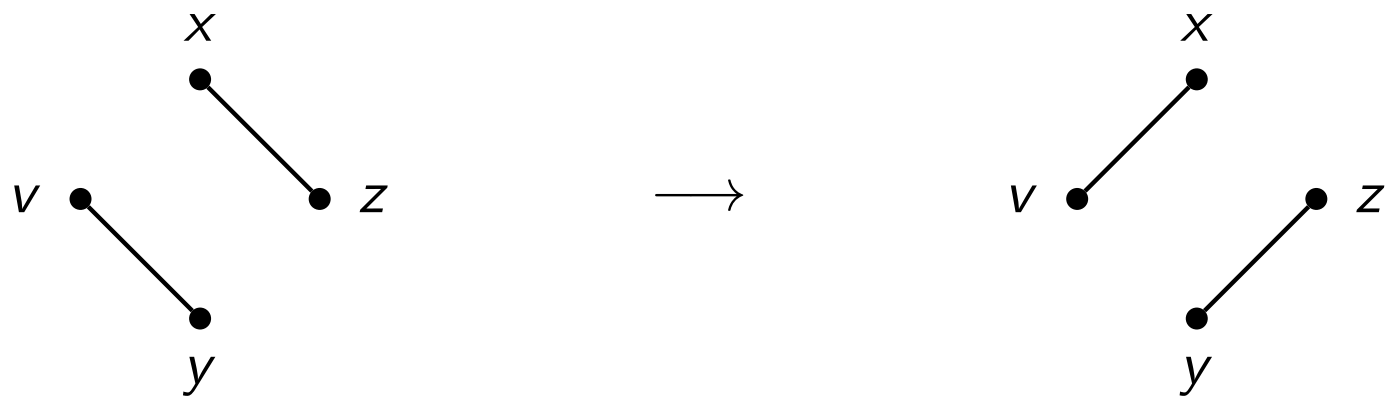
Πρόταση 2 (Θεώρημα Havel - Hakimi)

Η φθίνουσα ακολουθία $d = (d_1, d_2, \ldots, d_n)$ *είναι γραφική αν και μόνο αν η ακολουθία* $d' = (d_2 - 1, d_3 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, d_{d_1+3}, \ldots, d_n)$ *είναι γραφική.*

Απόδειξη. Το αντίστροφο είναι προφανές. Πράγματι, δοθέντος ενός γραφήματος με ακολουθία βαθμών d' προσθέτουμε σε αυτό μια νέα κορυφή v η οποία ενώνεται με τις d_1 κορυφές με τον μεγαλύτερο βαθμό, παίρνοντας έτσι ένα γράφημα με ακολουθία βαθμών d.

Για το ευθύ, θεωρούμε ένα γράφημα $G = (V, E)$ με ακολουθία βαθμών d και μια οποιαδήποτε κορυφή v, έστω βαθμού k. Η απόδειξη βασίζεται στην παρατήρηση ότι μπορούμε να εναλλάξουμε τους δεσμούς του G χωρίς να αλλάξουν οι βαθμοί των κορυφών του, έτσι ώστε η v να συνδέεται με τις (υπόλοιπες) κορυφές που έχουν τους k μεγαλύτερους βαθμούς.

Πράγματι, έστω ότι $x, y \in V$ με $d(x) > d(y)$ και η v έχει γείτονα τον y αλλά όχι τον x.



Επειδή $d(x) > d(y)$ υπάρχει γείτονας z της x που δεν είναι γείτονας της y.
Διαγράφοντας τις ακμές $\{v, y\}$ και $\{x, z\}$ και προσθέτοντας τις ακμές $\{v, x\}$ και $\{y, z\}$ προκύπτει ένα γράφημα G' στο οποίο οι κορυφές διατηρούν τους ίδιους βαθμούς που είχαν στο G και η v συνδέεται με την x.
Επαναλαμβάνοντας αυτόν τον μετασχηματισμό μπορούμε να πετύχουμε την ζητούμενη ιδιότητα.
Τέλος, διαγράφοντας την κορυφή v με τον μέγιστο βαθμό παίρνουμε ένα γράφημα με ακολουθία βαθμών την d'.

Παράδειγμα : Σύμφωνα με το θεώρημα Havel - Hakimi η ακολουθία $(6, 6, 4, 4, 2, 2, 2, 2)$ είναι γραφική αν και μόνο αν η ακολουθία

$$(6-1, 4-1, 4-1, 2-1, 2-1, 2-1, 2) = (5, 3, 3, 1, 1, 1, 2)$$

είναι γραφική. Η ακολουθία $(5, 3, 3, 2, 1, 1, 1)$ είναι γραφική ανν η ακολουθία

$$(3-1, 3-1, 2-1, 1-1, 1-1, 1) = (2, 2, 1, 0, 0, 1)$$

είναι γραφική. Η ακολουθία $(2, 2, 1, 1, 0, 0)$ είναι γραφική ανν η ακολουθία

$$(2-1, 1-1, 1, 0, 0, 0) = (1, 0, 1, 0, 0, 0)$$

είναι γραφική. Η ακολουθία $(1, 1, 0, 0, 0, 0)$ είναι πράγματι γραφική, αφού αντιστοιχεί στο γράφημα,

άρα και η ακολουθία $(6, 6, 4, 4, 2, 2, 2, 2)$ είναι επίσης γραφική.

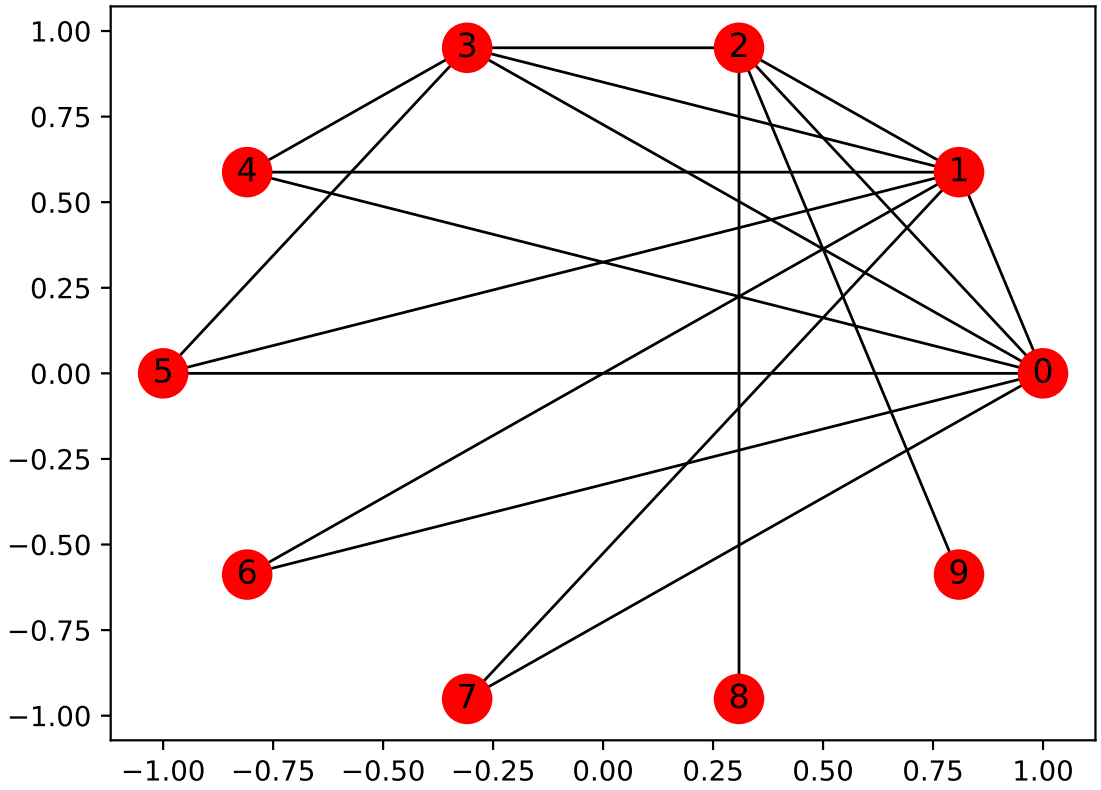
Το θεώρημα Havel - Hakimi δίνει ένα αναδρομικό κριτήριο για τον έλεγχο του κατά πόσο μια ακολουθία είναι γραφική, και ανάγει το πρόβλημα στον έλεγχο μια ακολουθίας με μήκος ένα λιγότερο από την αρχική. Μπορούμε να εφαρμόσουμε ξανά το θεώρημα στην ακολουθία $(d_2 - 1, d_3 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, d_{d_1+3}, \ldots, d_n)$ (αρκεί πρώτα να την διατάξουμε σε φθίνουσα σειρά) και να προκύψει μια ακολουθία με μικρότερο μήκος, μέχρις ότου να καταλήξουμε σε μία ακολουθία για την οποία μπορούμε εύκολα να συμπεράνουμε αν είναι γραφική ή όχι. Οι ακολουθίες που προκύπτουν κατά την αναδρομική εφαρμογή του θεωρήματος Havel - Hakimi είτε είναι όλες γραφικές είτε καμία γραφική.

Για τον έλεγχο ύπαρξης και κατασκευής ενός γραφήματος με συγκεκριμένη ακολουθία βαθμών μπορούμε να χρησιμοποιήσουμε τις μεθόδους is_graphical και havel_hakimi_graph της βιβλιοθήκης networkx.

```
import networkx as nx
import matplotlib.pyplot as plt

def Graph_Check(Seq):
    if nx.is_graphical(Seq):
        print("The sequence",Seq,"is graphical")
        G = nx.havel_hakimi_graph(Seq)
        nx.draw_circular(G,with_labels=True)
        plt.show()
    else:
        print("The sequence",Seq,"is not graphical")
    print("")

Seq1 = [7,7,5,5,3,3,2,2,1,1]
Seq2 = [7,7,7,3,3,2,1,1,1]

Graph_Check(Seq1)
Graph_Check(Seq2)
```

Output:

```
The sequence [7, 7, 5, 5, 3, 3, 2, 2, 1, 1] is graphical
```



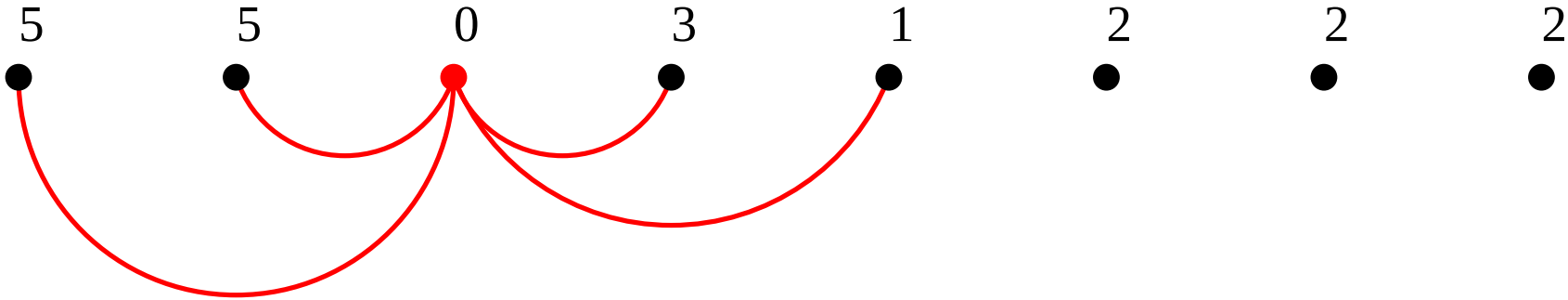

```
The sequence [7, 7, 7, 3, 3, 2, 1, 1, 1] is not graphical
```

Ο αλγόριθμος Havel - Hakimi

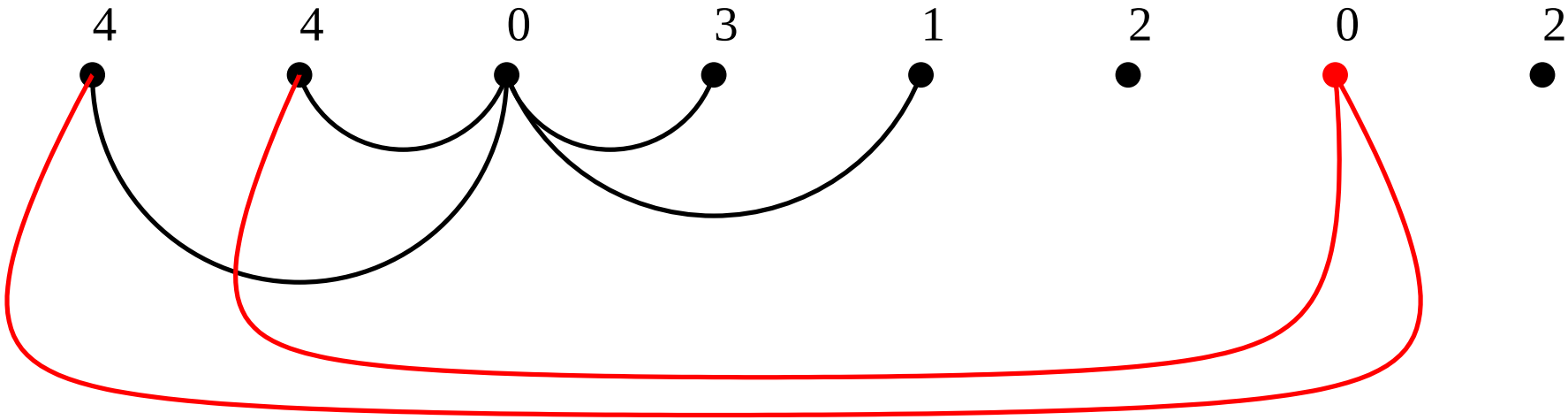
Επίσης, το θεώρημα Havel - Hakimi μας προσφέρει μια απλή επαναληπτική μέθοδο για να κατασκευάζουμε γραφήματα με συγκεκριμένη ακολουθία βαθμών.
Συγκεκριμένα, αν $(d_1, d_2, \ldots, d_n)$ είναι μια γραφική ακολουθία ο αλγόριθμος κατασκευής χρησιμοποιεί μια βοηθητική ακολουθία $(a_1, a_2, \ldots, a_n)$ και λειτουργεί ως εξής:
Αρχικά θεωρούμε n κορυφές $\{v_1, v_2, \ldots, v_n\}$ βαθμού 0 και για κάθε κορυφή σημειώνουμε τον αριθμό a_i των δεσμών που απαιτούνται για να αποκτήσει βαθμό d_i.
Σε κάθε βήμα επιλέγουμε οποιαδήποτε κορυφή, έστω την v_k, με $a_k > 0$, και την συνδέουμε με a_k σε πλήθος κορυφές που έχουν τις μεγαλύτερες δυνατές θετικές τιμές στην ακολουθία $(a_1, a_2, \ldots, a_n)$. Ενημερώνουμε την ακολουθία $(a_1, a_2, \ldots, a_k)$ και επαναλαμβάνουμε το βήμα αυτό μέχρις ότου όλα τα a_k γίνουν μηδενικά.

Παράδειγμα : Για να κατασκευάσουμε ένα γράφημα με ακολουθία βαθμών $(6, 6, 4, 4, 2, 2, 2, 2)$ αρχικά θεωρούμε 8 κορυφές βαθμού 0. (Στο σχήμα οι κορυφές $v_1, v_2, \ldots, v_8$ αναπαρίστανται με την σειρά από τα αριστερά προς τα δεξιά και σημειώνονται μόνο οι τιμές της ακολουθίας a_i).

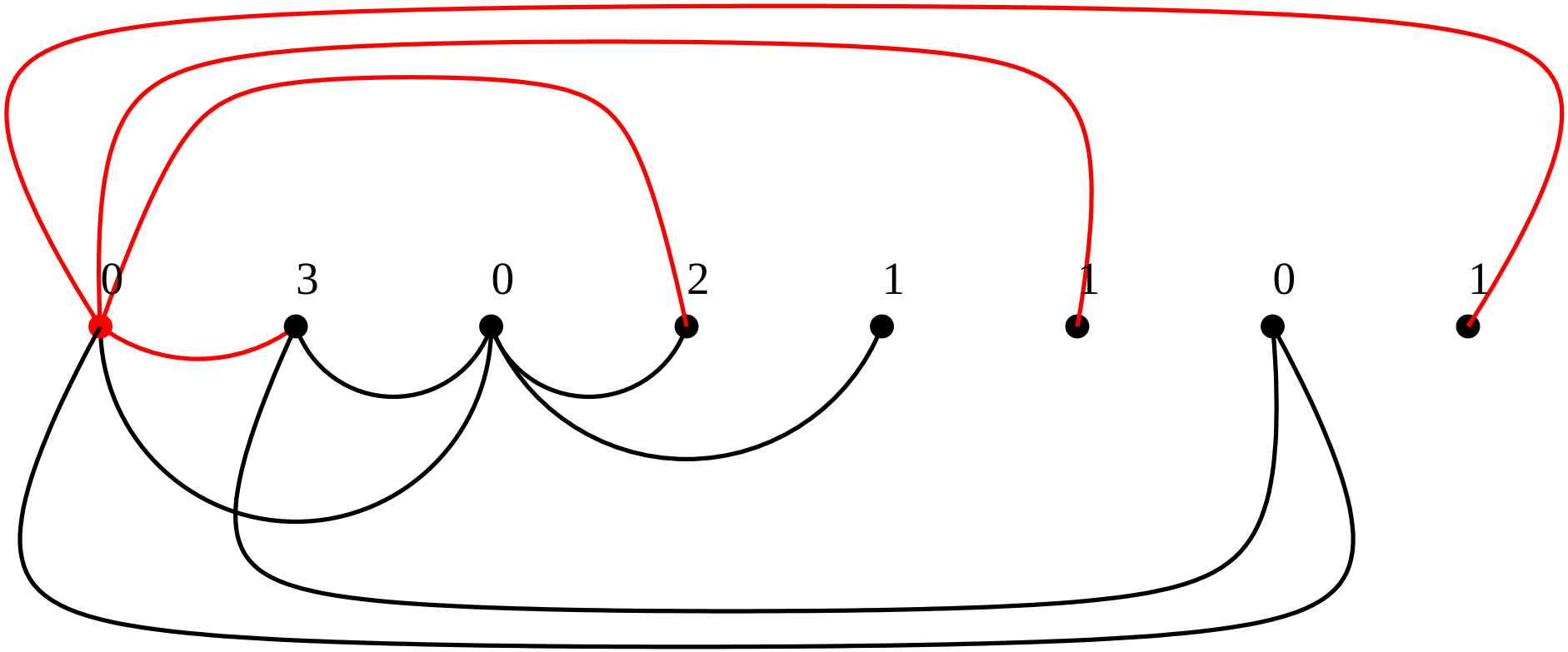
Επιλέγουμε (αυθαίρετα) την κορυφή v_3 (που είναι μαρκαρισμένη με κόκκινο). Η v_3 έχει $a_3 = 4$ οπότε την συνδέουμε με τις 4 κορυφές οι οποίες έχουν τις μεγαλύτερες δυνατές τιμές της ακολουθίας a_i, εδώ είναι οι κορυφές v_1, v_2, v_4 και v_5, οπότε προκύπτει το επόμενο γράφημα στο οποίο έχουμε ενημερώσει τις αντίστοιχες τιμές των a_i.

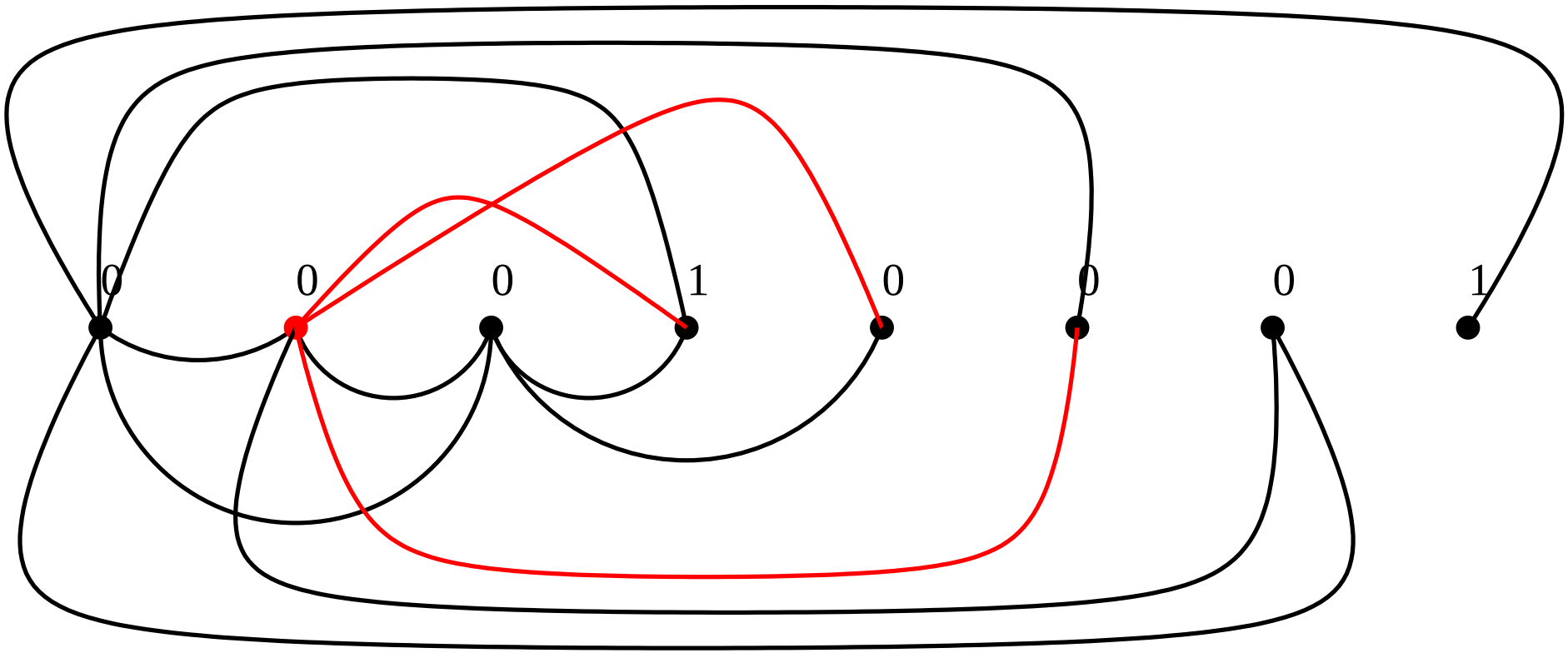


Στην συνέχεια, επιλέγουμε (αυθαίρετα) την κορυφή v_7 για την οποία $a_7 = 2$ και την συνδέουμε με τις 2 κορυφές οι οποίες έχουν τις μεγαλύτερες δυνατές τιμές της ακολουθίας a_i, εδώ είναι οι κορυφές v_1, v_2, οπότε προκύπτει το επόμενο γράφημα στο οποίο έχουμε ενημερώσει τις αντίστοιχες τιμές των a_i.

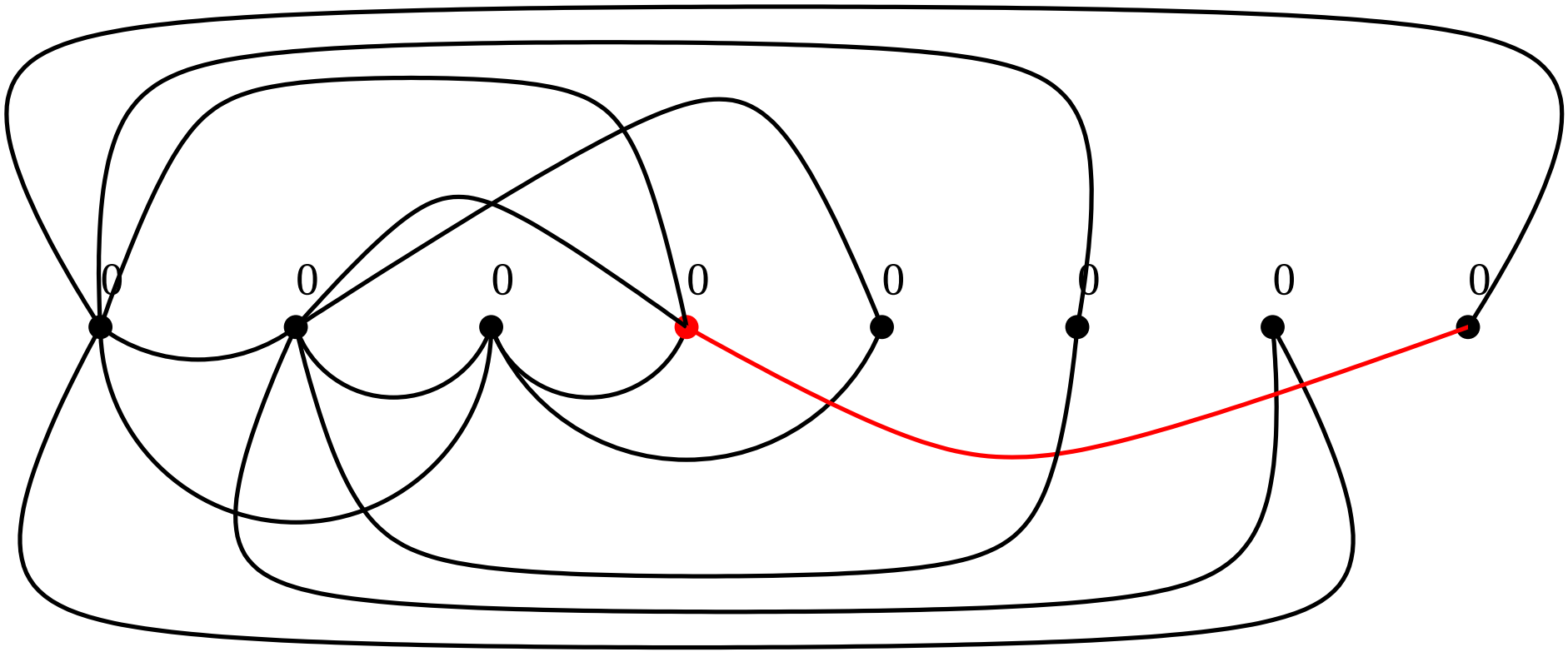


Συνεχίζουμε με τον ίδιο τρόπο, επιλέγοντας κορυφές v_k με $a_k > 0$, οπότε προκύπτουν διαδοχικά τα γραφήματα:




0
0
0
1
0
0
0
1

και τέλος το γράφημα



το οποίο έχει ακολουθία βαθμών $(6, 6, 4, 4, 2, 2, 2, 2)$.

Παρατηρησεις : Στην περίπτωση όπου η ακολουθία $(d_1, d_2, \ldots, d_n)$ δεν είναι γραφική ο αλγόριθμος θα αποτύχει διότι θα υπάρχουν θετικά a_k αλλά δεν θα υπάρχουν αρκετές διαθέσιμες κορυφές για να συνδεθεί η κορυφή v_k.

2η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

- **Μήτρα γραφήματος και λίστα γειτονικότητας**
- **Γραφήματα με συνάρτηση κόστους**
- **Ισόμορφα γραφήματα**

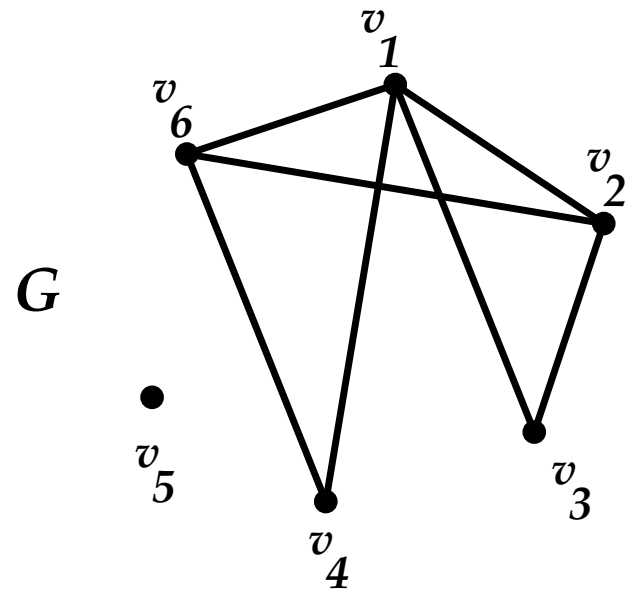
Μήτρα γραφήματος και λίστα γειτονικότητας

Έστω $G = (V, E)$. Ορίζουμε την $|V| \times |V|$ μήτρα M_G ή M του G ως εξής:

$$M = [m_{ij}], \text{ με } m_{ij} = \begin{cases} 1, & \text{αν } \{v_i, v_j\} \in E \\ 0, & \text{αν } \{v_i, v_j\} \notin E. \end{cases}$$

Η μήτρα αυτή ονομάζεται **μήτρα (γειτονικότητας) του γραφήματος δεσμών**.

Παράδειγμα Στο γράφημα G



αντιστοιχεί η μήτρα

$$M = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Παρατήρησεις

- Η μήτρα M είναι προφανώς συμμετρική.
- Ισχύει ότι

$$\sum_{j=1}^{|V|} m_{ij} = \sum_{j=1}^{|V|} m_{ji} = d(v_i),$$

(δηλαδή το άθροισμα των στοιχείων της i γραμμής ισούται με το άθροισμα των στοιχείων της i στήλης (και με τον βαθμό του κόμβου v_i)).

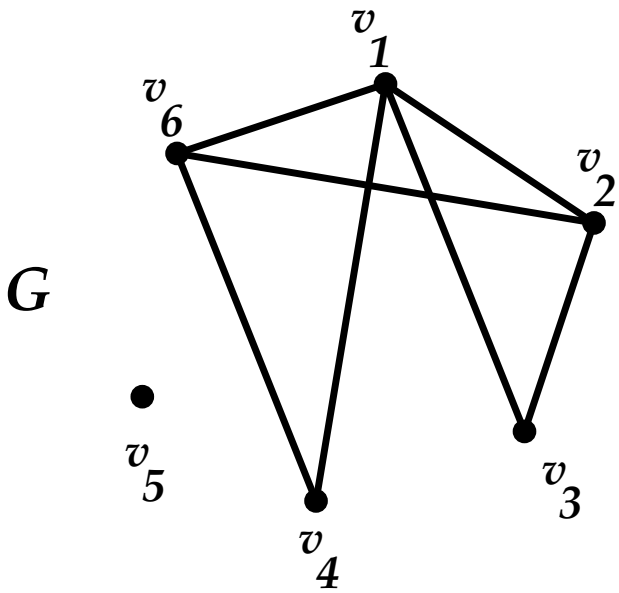
Για να εκφραστούν και να υλοποιηθούν αποδοτικότερα κάποιοι αλγόριθμοι, χρησιμοποιούνται και οι **λίστες γειτονικότητας**.

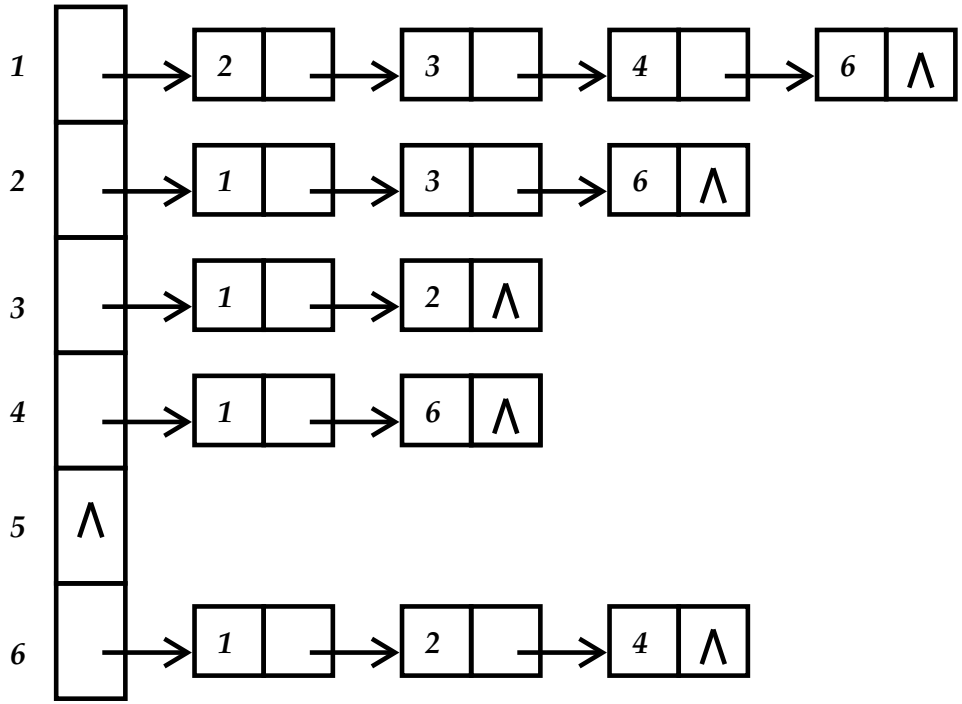
- Σ' αυτές, κάθε γραμμή (λίστα) αντιστοιχεί σε ένα κόμβο $v_i \in V$, ο δείκτης i του οποίου εμφανίζεται ως επικεφαλής δείκτης της λίστας.
- Τα υπόλοιπα στοιχεία της λίστας έχουν τη μορφή

κόμβος	δείκτης

όπου στην πρώτη θέση εμφανίζεται ο δείκτης ενός κόμβου που συνδέεται με τον v_i, ενώ η δεύτερη θέση συνδέεται με το επόμενο στοιχείο της λίστας (αν υπάρχει και άλλος κόμβος που συνδέεται με τον v_i) ή περιέχει ένα σύμβολο $\wedge$ (αν δεν υπάρχει άλλος κόμβος που συνδέεται με τον v_i).

Παράδειγμα Για το γράφημα G





η λίστα γειτονικότητας είναι:

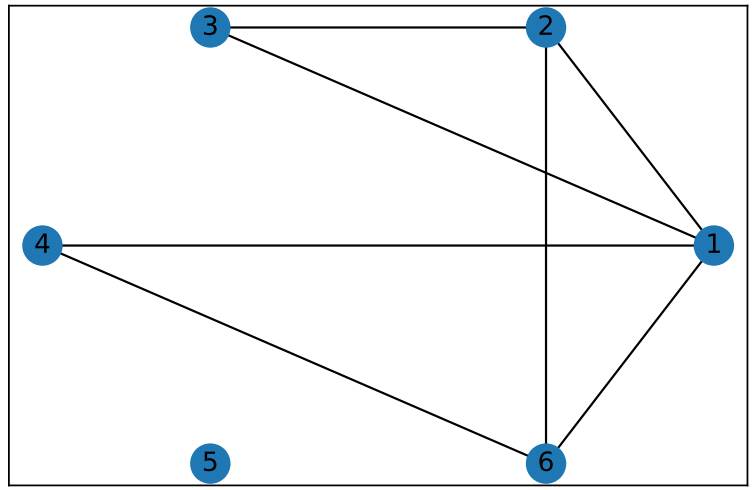
Παρατήρηση. Σε κάθε γραμμή εμφανίζονται όλοι οι κόμβοι που συνδέονται με τον επικεφαλής κόμβο (και όχι κατ' ανάγκη μεταξύ τους).

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()
V = [1,2,3,4,5,6]
E = [[1,2],[1,3],[1,4],[1,6],[2,3],[2,6],[4,6]]
G.add_nodes_from(V)
G.add_edges_from(E)

print("The (sparse) adjacency matrix of graph G is:")
print(nx.adjacency_matrix(G))
print("The adjacency matrix of graph G is:")
print(nx.adjacency_matrix(G).todense())

pos = nx.circular_layout(G)
nx.draw_networkx(G,pos)
plt.show()
```

Output:

```
The (sparse) adjacency matrix of
      graph G is:
  (0, 1)  1
  (0, 2)  1
  (0, 3)  1
  (0, 5)  1
  (1, 0)  1
  (1, 2)  1
  (1, 5)  1
  (2, 0)  1
  (2, 1)  1
  (3, 0)  1
  (3, 5)  1
  (5, 0)  1
  (5, 1)  1
  (5, 3)  1

The adjacency matrix of graph G
    is:
```

```
[[0 1 1 1 0 1]
 [1 0 1 0 0 1]
 [1 1 0 0 0 0]
 [1 0 0 0 0 1]
 [0 0 0 0 0 0]
 [1 1 0 1 0 0]]
```

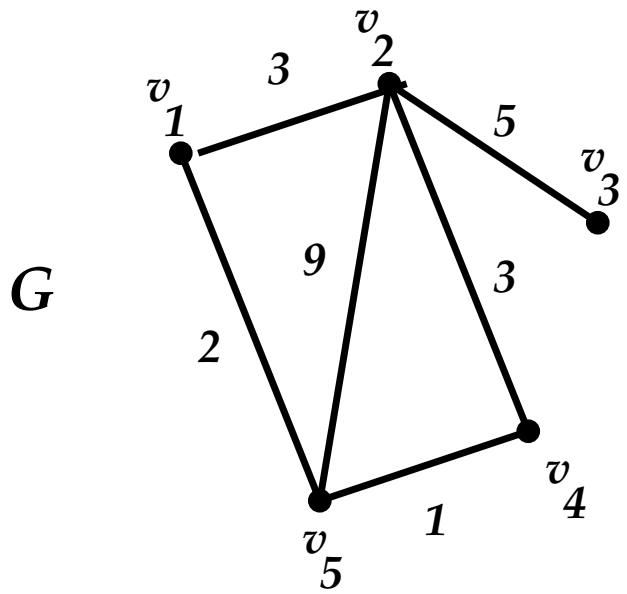


Γραφήματα με συνάρτηση κόστους

Με ένα γράφημα $G(V, E)$ μπορεί να συσχετισθεί κάποια **συνάρτηση κόστους**, δηλαδή μια συνάρτηση $f : E \to \mathbb{R}$. Οι τιμές $f(\{v_i, v_j\})$ για κάθε $\{v_i, v_j\} \in E$, δίδονται αντίστοιχα και πάνω στο γράφημα.
Βασικές έννοιες, όπως ο ισομορφισμός, τα υπογραφήματα κ.λπ. μεταφέρονται κατά προφανή τρόπο στα γραφήματα με συνάρτηση κόστους.
Προφανώς, μπορούμε επίσης να ορίσουμε και την αντίστοιχη μήτρα γειτονικότητας:

$$M = [m_{ij}], \text{ με } m_{ij} = \begin{cases} f(\{v_i, v_j\}), & \text{αν } \{v_i, v_j\} \in E \\ 0, & \text{αν } \{v_i, v_j\} \notin E. \end{cases}$$

Παράδειγμα Στο γράφημα G



αντιστοιχεί η μήτρα

$$M = \begin{bmatrix} 0 & 3 & 0 & 0 & 2 \\ 3 & 0 & 5 & 3 & 9 \\ 0 & 5 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 1 \\ 2 & 9 & 0 & 1 & 0 \end{bmatrix}.$$

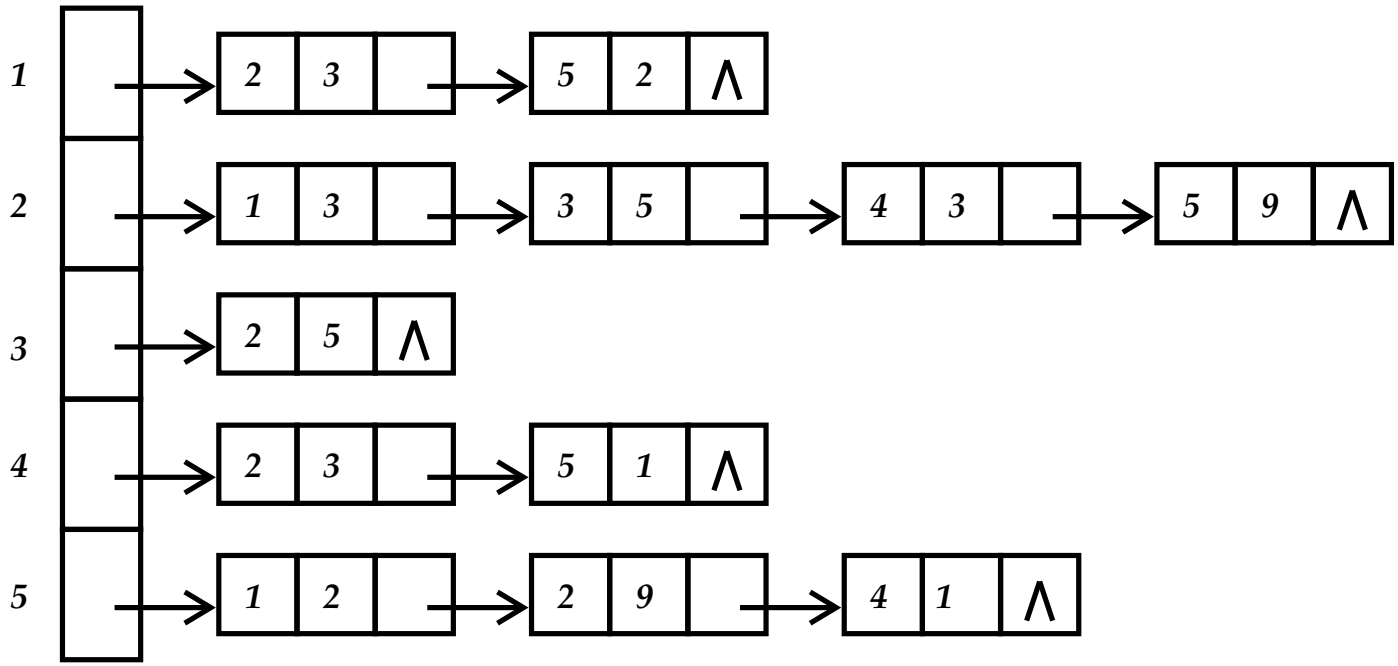
Παρατήρηση. Προφανώς τα γραφήματα χωρίς συνάρτηση κόστους μπορούν να θεωρηθούν σαν ειδική περίπτωση, όπου $f(\{v_i, v_j\}) = 1$, για κάθε $\{v_i, v_j\} \in E$.

Στην περίπτωση που το γράφημα έχει συνάρτηση κόστους, τα στοιχεία της λίστας έχουν την μορφή

κόμβος	κόστος	δείκτης

όπου στη δεύτερη θέση εμφανίζεται το κόστος του αντίστοιχου δεσμού.

Παράδειγμα Για το γράφημα G η λίστα γειτονικότητας είναι:



Ισόμορφα γραφήματα

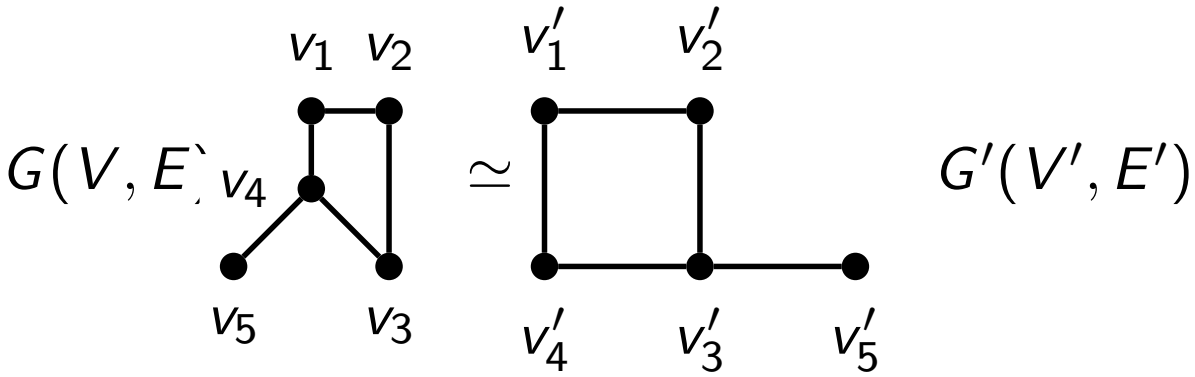
Τα γραφήματα δεσμών $G = (V, E)$ και $G' = (V', E')$ ονομάζονται **ισόμορφα** αν και μόνο αν

- υπάρχει αμφιμονοσήμαντη απεικόνιση $f : V \to V'$,
- με $\{v, u\} \in E \Leftrightarrow \{f(v), f(u)\} \in E'$.

Αν δύο γραφήματα G και G' είναι ισόμορφα, θα γράφουμε $G \simeq G'$.

Παραδείγματα

Τα επόμενα γραφήματα είναι ισόμορφα



διότι για την $f : V \to V'$ με

$$\begin{aligned} f(v_1) &= v'_4, \\ f(v_2) &= v'_1, \\ f(v_3) &= v'_2, \\ f(v_4) &= v'_3, \\ f(v_5) &= v'_5, \end{aligned}$$

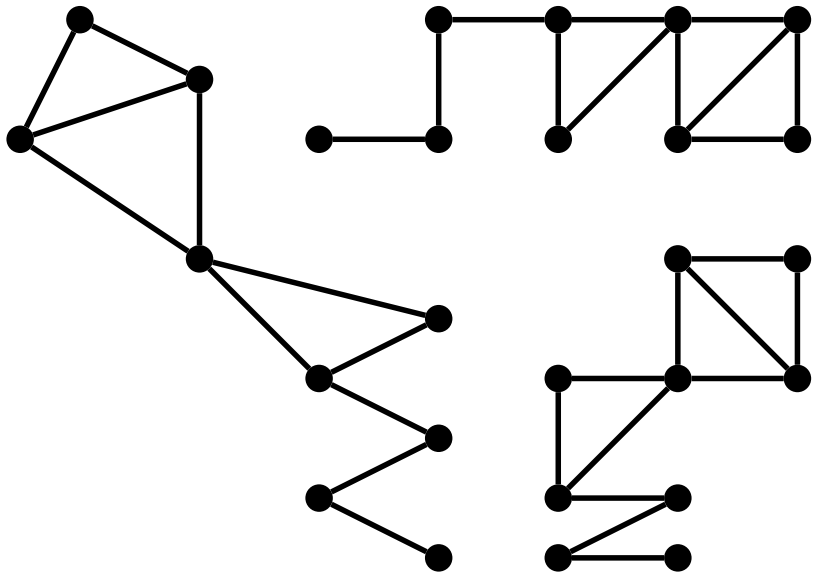
έχουμε πράγματι ότι

$$\{v_i, v_j\} \in E \Leftrightarrow \{f(v_i), f(v_j)\} \in E',$$

(για παράδειγμα :

$$\begin{aligned} &\{v_1, v_2\} \in E \text{ και } \{v'_4, v'_1\} \in E', \\ &\{v_2, v_4\} \notin E \text{ και } \{v'_1, v'_3\} \notin E', \\ &\{v_4, v_5\} \in E \text{ και } \{v'_3, v'_5\} \in E, \quad \text{κ.ο.κ.}). \end{aligned}$$

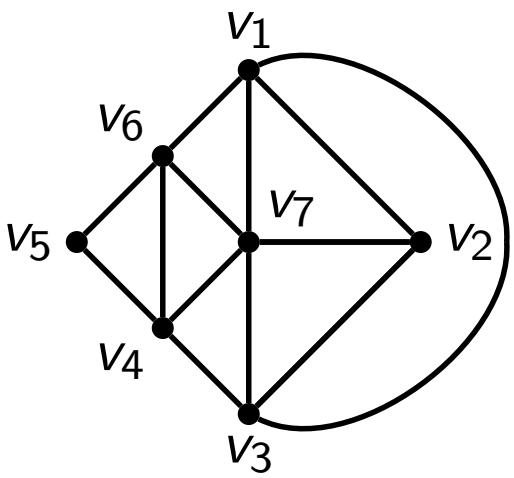
Τα επόμενα γραφήματα είναι όλα ισόμορφα:



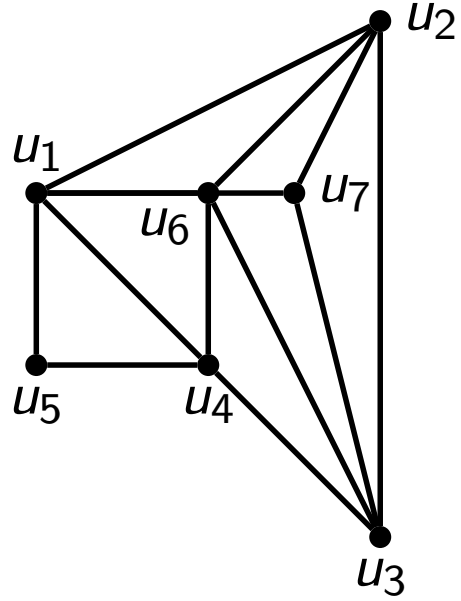
Αντίθετα τα επόμενα δύο γραφήματα δεν είναι ισόμορφα:

Παράδειγμα Να εξετασθεί αν τα γραφήματα G_1 και G_2 είναι ισόμορφα.

G_1



G_2



Τα G_1, G_2 είναι ισόμορφα. Ένας ισομορφισμός f είναι ο εξής:

$f(v_1) = u_2$
$f(v_2) = u_7$
$f(v_3) = u_3$
$f(v_4) = u_4$
$f(v_5) = u_5$
$f(v_6) = u_1$
$f(v_7) = u_6$

Μπορούμε να ελέγξουμε αν τα G_1, G_2 είναι ισόμορφα χρησιμοποιώντας την μέθοδο GraphMatcher της βιβλιοθήκης networkx.

```
import networkx as nx
G1 = nx.Graph()
G1.add_nodes_from(range(1,8))
G1.add_edges_from([[1,2],[1,3],[1,6],[1,7],[2,3],[2,7],
                   [3,4],[3,7],[4,5],[4,6],[4,7],[5,6],[6,7]])
G2 = nx.Graph()
G2.add_nodes_from(range(1,8))
G2.add_edges_from([[1,2],[1,5],[1,4],[1,6],[2,3],[2,6],
                   [2,7],[3,4],[3,6],[3,7],[4,5],[4,6],[6,7]])
#Test whether the graphs are isomorphic
GM = nx.isomorphism.GraphMatcher(G1,G2)
if GM.is_isomorphic():  #If G1, G2 are isomorphic
    print("The graphs are isomorphic")
    print("An isomorphism between them is the following:")
    for i in G1:
        print("v",i,"->","u",GM.mapping[i])
    print(GM.mapping)
else:
    print("The graphs are not isomorphic")
```

Output:

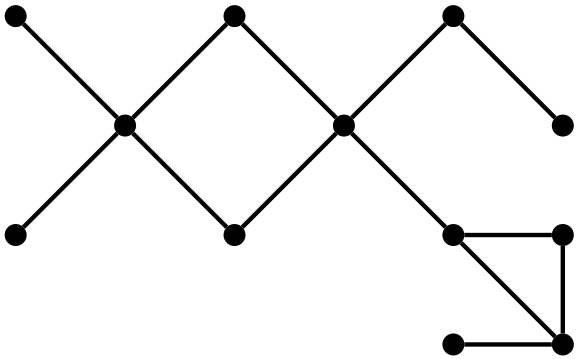
```
The graphs are isomorphic
An isomorphism between them is the following:
v 1 -> u 3
v 2 -> u 7
v 3 -> u 2
v 4 -> u 1
v 5 -> u 5
v 6 -> u 4
v 7 -> u 6
{4: 1, 3: 2, 1: 3, 6: 4, 5: 5, 7: 6, 2: 7}
```

Παρατήρηση Παρατηρήστε ότι ο αλγόριθμος ανακάλυψε έναν διαφορετικό ισομορφισμό από αυτόν που δόθηκε αρχικά.

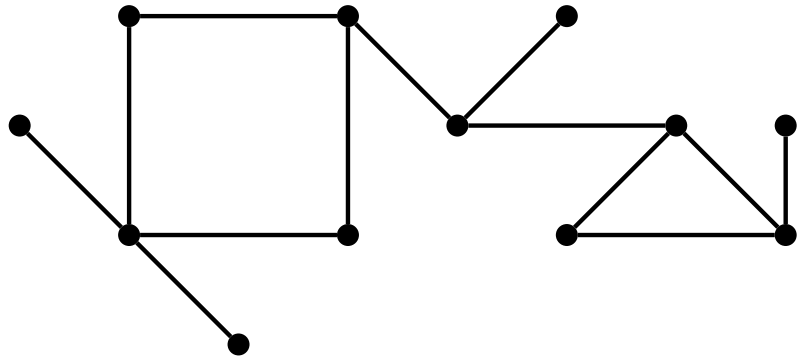
Πρόταση 3

Αν δύο γραφήματα G, H είναι ισόμορφα, τότε:
i) Έχουν την ίδια ακολουθία βαθμών, και μάλιστα ισχύει ότι $d_G(v) = d_H(f(v))$, για κάθε $v \in V(G)$.
ii) Έχουν ισόμορφα υπογραφήματα.

Συνήθως η Πρόταση 3 χρησιμοποιείται αρνητικά, δηλαδή αν δύο γραφήματα δεν έχουν την ίδια ακολουθία βαθμών, ή/και δεν έχουν τα ίδια υπογραφήματα, τότε δεν είναι ισόμορφα. Για παράδειγμα,

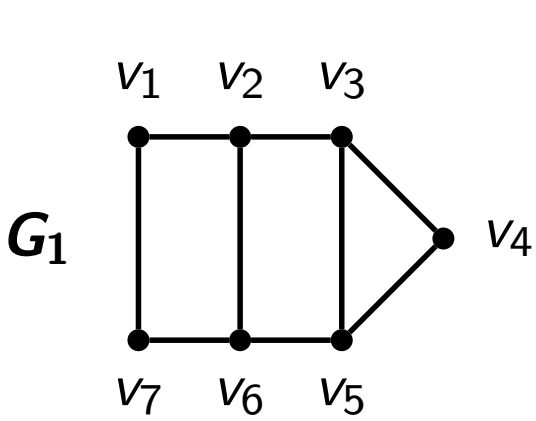
- G_1 

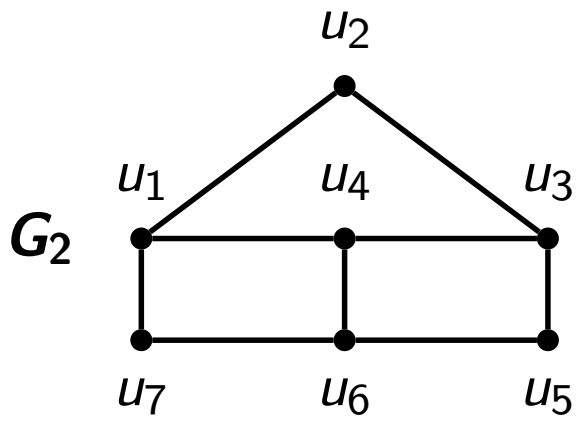
$(4, 4, 3, 3, 2, 2, 2, 2, 1, 1, 1, 1)$

G_2 

$(4, 3, 3, 3, 3, 2, 2, 2, 1, 1, 1, 1)$

$G_1 \not\cong G_2$, διότι έχουν διαφορετικές ακολουθίες βαθμών.





Δεν είναι ισόμορφα: Το G_1 περιέχει κύκλο μήκους 3 ενώ το G_2 όχι.

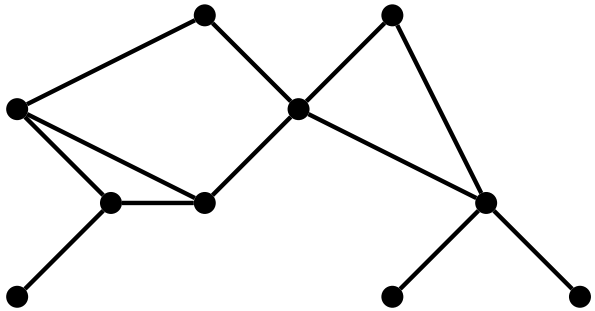
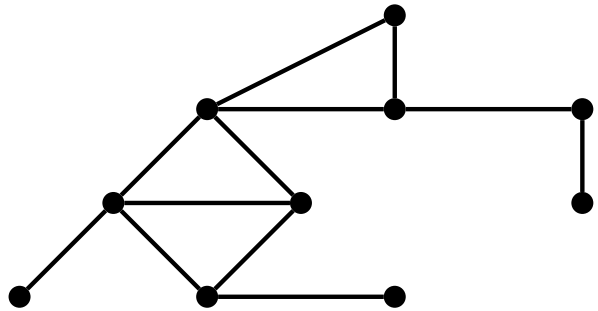
Πάλι, μπορούμε να ελέγξουμε αν τα G_1, G_2 είναι ισόμορφα χρησιμοποιώντας την μέθοδο GraphMatcher της βιβλιοθήκης networkx.

```
import networkx as nx
G1 = nx.Graph()
G1.add_nodes_from(range(1,8))
G1.add_edges_from
    ([[1,2],[1,7],[2,3],[2,6],[3,4],[3,5],[4,5],[5,6],[6,7]])
G2 = nx.Graph()
G2.add_nodes_from(range(1,8))
G2.add_edges_from
    ([[1,2],[1,4],[1,7],[2,3],[3,4],[3,5],[4,6],[5,6],[6,7]])
#Test whether the graphs are isomorphic
GM = nx.isomorphism.GraphMatcher(G1,G2)
if GM.is_isomorphic():  #If G1, G2 isomorphic? then
    print("The graphs are isomorphic")
    print("An isomorphism between them is the following:")
    for i in G1:
        print("v",i,"->","u",GM.mapping[i])
    print(GM.mapping)
else:
    print("The graphs are not isomorphic")
    print("Degree sequence of G1:",sorted((d for n, d in G1.degree
    ()),reverse=True))
    print("Degree sequence of G2:",sorted((d for n, d in G2.degree
    ()),reverse=True))
```

Output:

```
The graphs are not isomorphic
Degree sequence of G1: [3, 3, 3, 3, 2, 2, 2]
Degree sequence of G2: [3, 3, 3, 3, 2, 2, 2]
```

Παρατήρηση Παρατηρήστε ότι παρόλο που τα γραφήματα είναι μη ισόμορφα έχουν την ίδια ακολουθία βαθμών. Στην περίπτωση όπου τα δύο γραφήματα δεν είναι ισόμορφα, η μέθοδος `GraphMatcher` δεν μας δίνει κάποια εξήγηση γιατί συμβαίνει αυτό.

G_3  G_4 

$(4, 4, 3, 3, 3, 2, 2, 1, 1, 1)$ $(4, 4, 3, 3, 3, 2, 2, 1, 1, 1)$

$G_3 \not\simeq G_4$, διότι, ενώ έχουν την ίδια ακολουθία βαθμών, έχουν διαφορετικά υπογραφήματα, (για παράδειγμα, το G_3 περιέχει δύο K_3, ενώ το G_4 περιέχει τρία K_3).

Παρατήρηση. Ο έλεγχος αν δύο μεγάλα γραφήματα είναι ισόμορφα ή όχι είναι υπολογιστικά δύσκολος μέχρι σήμερα. Παρόλα αυτά υπάρχει η άποψη το πρόβλημα μπορεί να λυθεί γρηγορότερα αλλά δεν έχει βρεθεί ακόμα η κατάλληλη ιδέα.

3η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

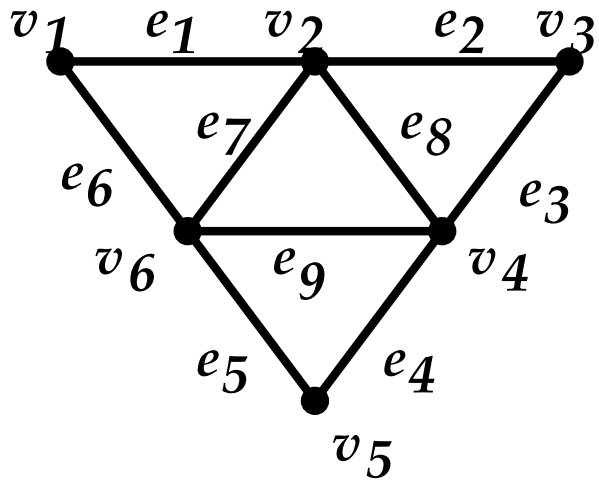
- **Συνεκτικότητα**
  - Διαδρομές
  - Δρόμοι
  - Μονοπάτια
  - Κύκλοι
- **Απαρίθμηση διαδρομών**
- **Απόσταση**
  - Διάμετρος
  - Ακτίνα
  - Εκκεντρότητα

Συνεκτικότητα

- **Διαδρομή** (walk) που ενώνει τους κόμβους v_i, v_j ενός γραφήματος G (ή $v_i - v_j$ **διαδρομή**) είναι μια ακολουθία της μορφής $(v_i, e_{ik}, v_k, e_{kl}, v_l, \ldots, v_r, e_{rj}, v_j)$, όπου e_{st} είναι ο δεσμός του γραφήματος που ενώνει τους κόμβους v_s και v_t. (Συνήθως περιγράφουμε μια διαδρομή μόνο με τους διαδοχικούς κόμβους της: $(v_i, v_k, v_l, \ldots, v_r, v_j)$). **Μήκος** μιας διαδρομής ονομάζεται το πλήθος των δεσμών της.
- Αν σε μια $v_i - v_j$ διαδρομή του G κάθε δεσμός εμφανίζεται μια μόνο φορά, η διαδρομή λέγεται $v_i - v_j$ **δρόμος** (trail) του G.
- Αν επιπλέον, σε ένα $v_i - v_j$ δρόμο του G κάθε κόμβος εμφανίζεται μια μόνο φορά, ο δρόμος λέγεται $v_i - v_j$ **μονοπάτι** (path) του G.
- Μια $v_i - v_j$ διαδρομή, ή ένας $v_i - v_j$ δρόμος του G, με $v_i = v_j$ λέγεται **κλειστή διαδρομή** του G ή **κλειστός δρόμος** του G.
- Τέλος, ένας κλειστός δρόμος του G, μήκους n, με n διακεκριμένους κόμβους, λέγεται **κύκλος** του G.

Παράδειγμα

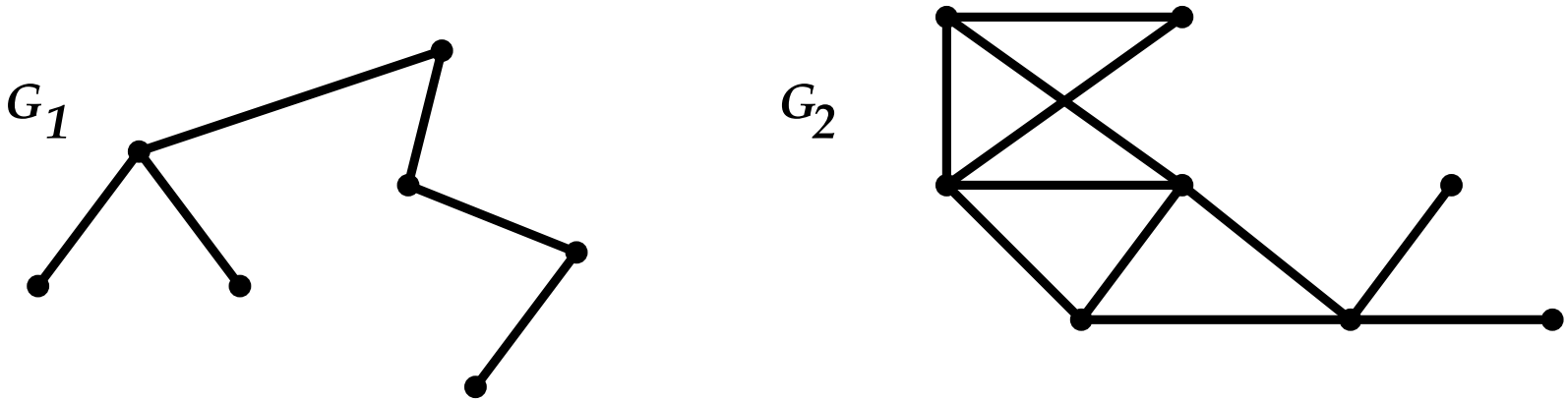
Για το γράφημα G έχουμε :



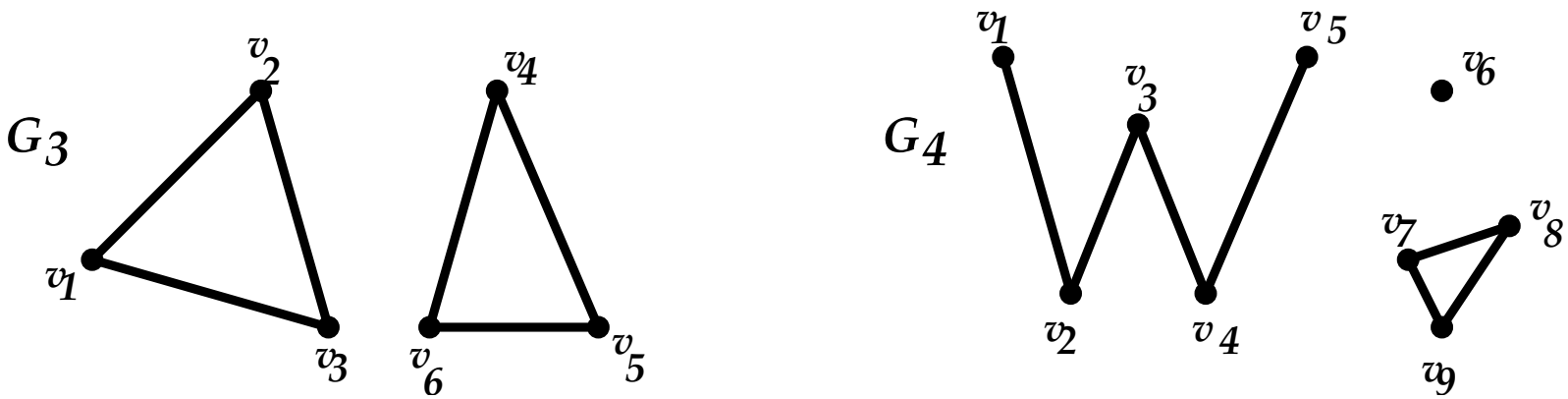
$v_1 - v_5$ διαδρομή του G (μήκους 6):
$(v_1, e_1, v_2, e_7, v_6, e_9, v_4, e_8, v_2, e_7, v_6, e_5, v_5)$, ή συντομότερα
$(v_1, v_2, v_6, v_4, v_2, v_6, v_5)$.
$v_1 - v_5$ δρόμος του G (μήκους 6): $(v_1, v_2, v_3, v_4, v_2, v_6, v_5)$.
$v_1 - v_5$ μονοπάτι του G (μήκους 5): $(v_1, v_2, v_3, v_4, v_6, v_5)$.
Κλειστή διαδρομή του G (μήκους 7): $(v_1, v_2, v_6, v_4, v_3, v_2, v_6, v_1)$.
Κλειστός δρόμος του G (μήκους 6): $(v_1, v_2, v_3, v_4, v_2, v_6, v_1)$.
Κύκλος του G (μήκους 4): $(v_1, v_2, v_4, v_6, v_1)$.

Ένα γράφημα λέγεται **συνεκτικό** αν για οποιουσδήποτε δύο κόμβους του, υπάρχει μονοπάτι που τους ενώνει.

Παραδείγματα



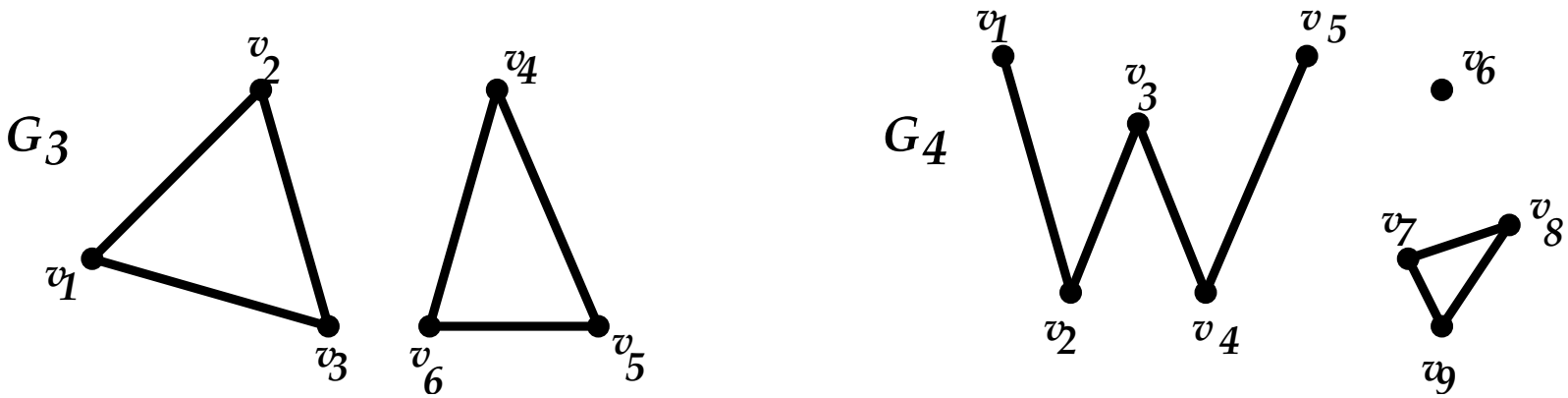
Συνεκτικά γραφήματα



Μη συνεκτικά γραφήματα

Συνιστώσα ενός γραφήματος G ονομάζεται κάθε μεγιστικό (maximal) συνεκτικό υπογράφημά του (δηλαδή κάθε συνεκτικό υπογράφημά του που δεν είναι υπογράφημα κάποιου άλλου συνεκτικού υπογραφήματος του G).
Προφανώς τα συνεκτικά γραφήματα αποτελούνται από μια μόνο συνιστώσα : τον εαυτό τους.

Παραδείγματα



Μη συνεκτικά γραφήματα

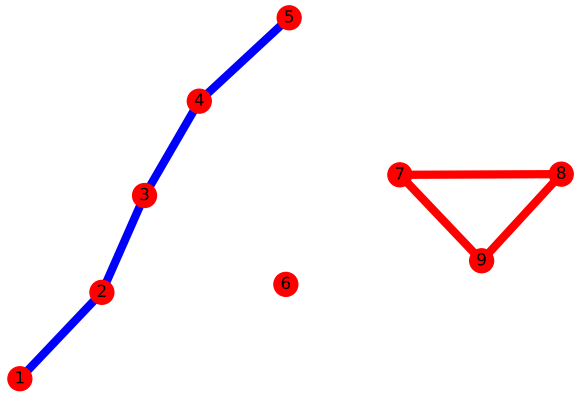
Στο προηγούμενο σχήμα οι συνιστώσες του G_3 είναι τα δύο τρίγωνα $G_{3,1}$, $G_{3,2}$ με $V(G_{3,1}) = \{v_1, v_2, v_3\}$ και $V(G_{3,2}) = \{v_4, v_5, v_6\}$ αντίστοιχα, ενώ το G_4 έχει προφανώς τρεις συνιστώσες.

Μπορούμε να βρούμε τις συνεκτικές συνιστώσες ενός γραφήματος με την βοήθεια της μεθόδου connected_components(G). Στο επόμενο πρόγραμμα σχεδιάζουμε με χρώματα τους δεσμούς τους ανάλογα με το μέγεθος κάθε συνιστώσας.

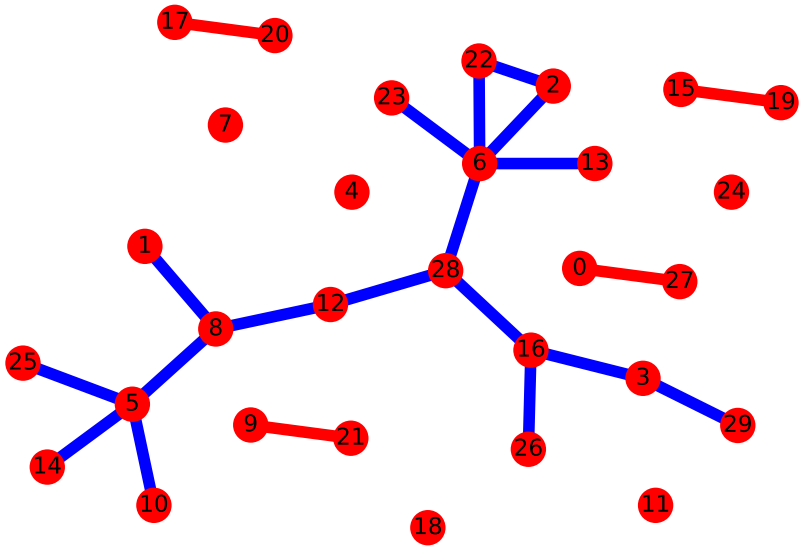
```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()
V = [v for v in range(1,10)]
E = [[1,2],[2,3],[3,4],[4,5],[7,8],[7,9],[8,9]]
G.add_nodes_from(V)
G.add_edges_from(E)

#draw the graph using graphviz layout positioning algorithm
#keep the positions of nodes in order to redraw
pos = nx.drawing.nx_agraph.graphviz_layout(G)
nx.draw(G,pos,with_labels=True)
```

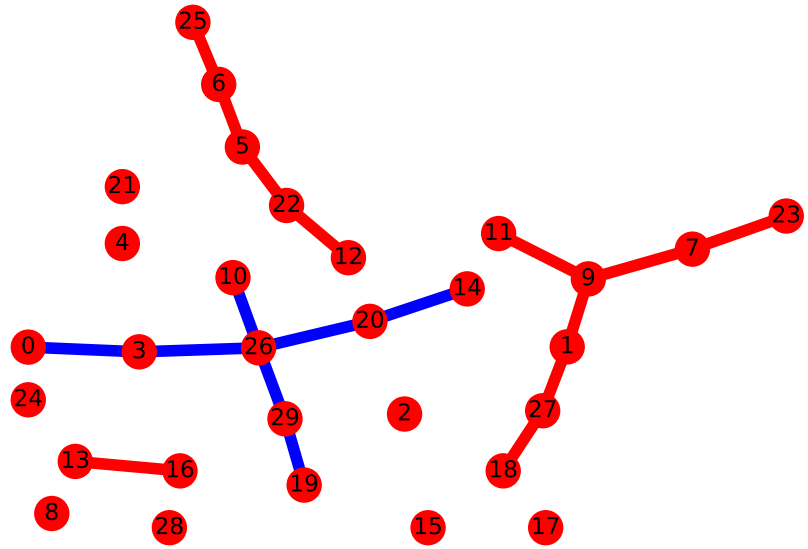
```
#find the connected components of G
#sort the list from the largest to smaller
Gcc = sorted(nx.connected_components(G), key=len, reverse=True)

#G0 the largest connected components
G0 = G.subgraph(Gcc[0])
#Draw the edges of the largest component
nx.draw_networkx_edges(G0,pos,edge_color='b',width=6.0)

#for every connected components of size > 1 draw its edges
for cc in Gcc[1:]:
    if len(cc) > 1:
        G1 = G.subgraph(cc)
        nx.draw_networkx_edges(G1,pos,edge_color='r', width=6.0)

plt.savefig("lect02a.eps")
plt.show()
```

Output:



Δύο επιπλέον παραδείγματα όπου το γράφημα G έχει κατασκευασθεί με την μέθοδο `nx.gnp_random_graph(30,0.054)`

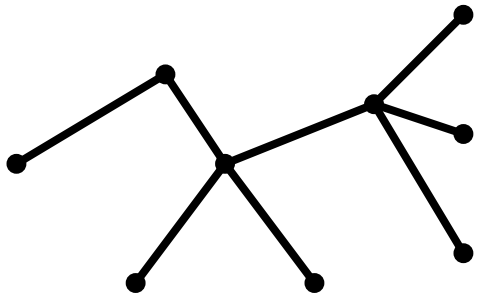




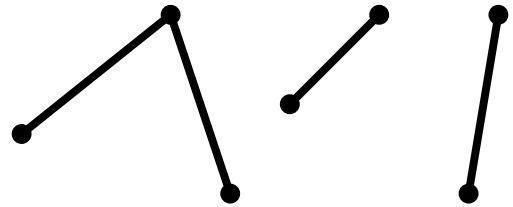
Άκυκλο ονομάζεται ένα γράφημα που δεν έχει κύκλους.

Παραδείγματα

G_1

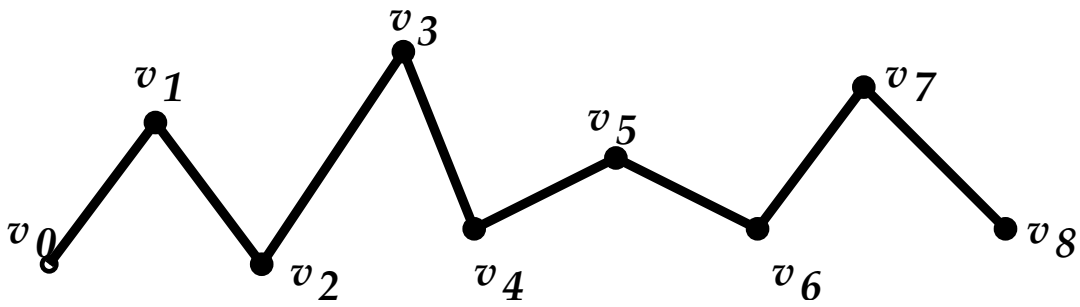


G_2



Ένα συνεκτικό γράφημα $G = (V, E)$ με $V = \{v_0, v_1, \ldots, v_n\}$ λέγεται $v_0 - v_n$ **μονοπάτι** (ή απλά **μονοπάτι**) μήκους n, αν $d(v_0) = d(v_n) = 1$ και $d(v_i) = 2$, για κάθε $i \in [n-1]$.

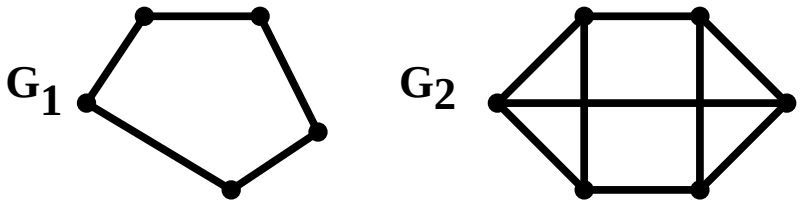
Παράδειγμα



Μονοπάτι μήκους 8

Ένα συνεκτικό 2-κανονικό γράφημα λέγεται **κύκλος**, ενώ ένα 3-κανονικό γράφημα λέγεται **κυβικό γράφημα**. Ένας κύκλος μήκους n, δηλαδή με n κόμβους, συμβολίζεται με C_n.

Παραδείγματα



Τα γραφήματα G_1, G_2 είναι κύκλος και κυβικό γράφημα αντίστοιχα.

Παρατήρηση. Παρατηρείστε τη διαφορά ανάμεσα στους ορισμούς «μονοπάτι γραφήματος» και «κύκλος γραφήματος» που δόθηκαν νωρίτερα και στους ορισμούς των γραφημάτων «μονοπάτι» και «κύκλος» που δίνονται εδώ.

Απαρίθμηση διαδρομών

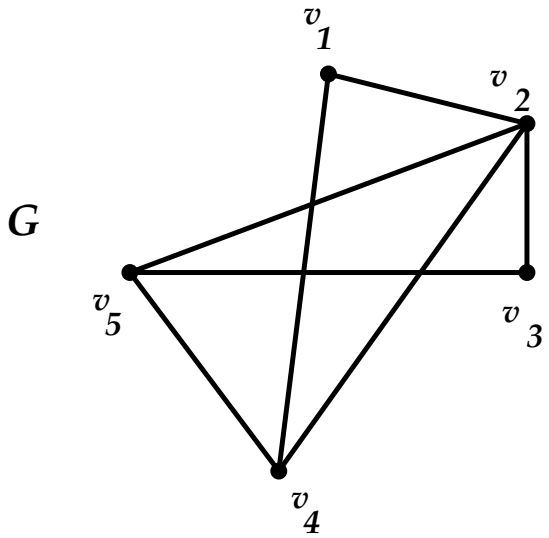
Έστω $G = (V, E)$ ένα γράφημα δεσμών με $|V| = n$ και μήτρα γειτνίασης $M = [m_{ij}]$.
Το στοιχείο p_{ij} της μήτρας M^2 δίδεται από τον τύπο:

$$\begin{aligned} p_{ij} &= \sum_{r=1}^{n} m_{ir} m_{rj} \\ &= m_{i1} m_{1j} + m_{i2} m_{2j} + \cdots + m_{in} m_{nj} \end{aligned}$$

Για κάθε $r \in [n]$, το γινόμενο $m_{ir} m_{rj}$ ισούται με 1 ανν $m_{ir} = m_{rj} = 1$, δηλαδή όταν υπάρχουν οι δεσμοί $\{v_i, v_r\}$ και $\{v_r, v_j\}$. Ισοδύναμα, $m_{ir} m_{rj} = 1$ ανν υπάρχει η διαδρομή μήκους 2 μεταξύ των v_i και v_j που διέρχεται από το v_r.
Επομένως, το p_{ij} ισούται με το άθροισμα όλων των διαδρομών μήκους 2 μεταξύ των v_i και v_j που διέρχονται από κάθε μια από τις υπόλοιπες κορυφές $v_1, v_2, \ldots, v_n$.

Απαρίθμηση διαδρομών

Παράδειγμα Για το γράφημα



$$M = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}, \; M^2 = \begin{bmatrix} 2 & 1 & 1 & 1 & 2 \\ 1 & 4 & 1 & 2 & 2 \\ 1 & 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 3 & 1 \\ 2 & 2 & 1 & 1 & 3 \end{bmatrix},$$

Η ιδιότητα αυτή γενικεύεται επαγωγικά στην επόμενη πρόταση.

Απαρίθμηση διαδρομών

Πρόταση 4 (Απαρίθμηση διαδρομών μήκους n)

Έστω $G = (V, E)$ γράφημα δεσμών με μήτρα γειτνίασης M. Ο αριθμός των διαδρομών μήκους ν, από την κορυφή v_i στην κορυφή v_j ισούται με το στοιχείο p_{ij} της μήτρας $M^\nu = [p_{ij}]$.

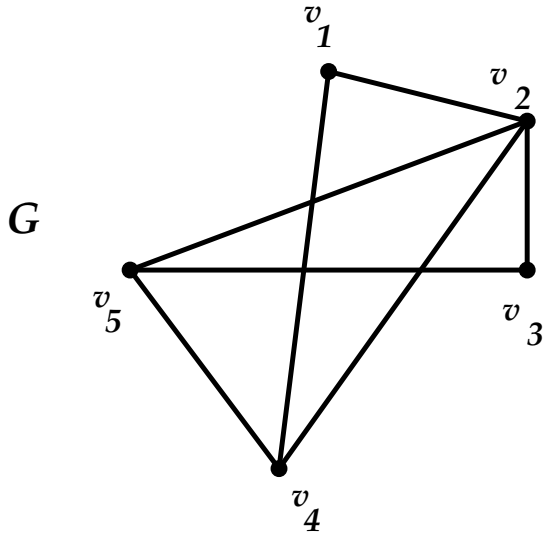
Απόδειξη. Για $\nu = 1$ προφανώς ισχύει.
Έστω ότι ισχύει για $\nu = k$ (και έστω ότι $M = [m_{ij}]$ και $M^k = [p_{ij}]$).
Για $\nu = k + 1$ θα είναι $M^{k+1} = M^k M = [q_{ij}]$, όπου

$$q_{ij} = \sum_{r=1}^{n} p_{ir} m_{rj} \tag{1}$$

(όπου $n = |V|$).
Από την υπόθεση της επαγωγής, p_{ir} είναι ο αριθμός των διαδρομών μήκους k από την v_i στην v_r. Επίσης, m_{rj} είναι ο αριθμός των δεσμών (δηλαδή των διαδρομών μήκους 1) από την v_r στην v_j.

Τότε το $p_{ir}\, m_{rj}$ είναι ο αριθμός των διαδρομών μήκους $k+1$ από την v_i στην v_j, με προτελευταία κορυφή την v_r. Άρα το $\sum_{r=1}^{n} p_{ir}\, m_{rj}$ (δηλαδή, λόγω της (1), το q_{ij}) θα είναι ο αριθμός όλων των διαδρομών μήκους $k+1$ από την v_i στην v_j με προτελευταία κορυφή μιας από τις $v_1, v_2, \ldots, v_n$, δηλαδή θα είναι πράγματι ο αριθμός όλων των διαδρομών μήκους $k+1$ από την v_i στην v_j.

Παράδειγμα Για το γράφημα



$$M = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix},$$

$$M^2 = \begin{bmatrix} 2 & 1 & 1 & 1 & 2 \\ 1 & 4 & 1 & 2 & 2 \\ 1 & 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 3 & 1 \\ 2 & 2 & 1 & 1 & 3 \end{bmatrix},$$

$$M^3 = \begin{bmatrix} 2 & 6 & 3 & 5 & 3 \\ 6 & 6 & 6 & 7 & 7 \\ 3 & 6 & 2 & 3 & 5 \\ 5 & 7 & 3 & 4 & 7 \\ 3 & 7 & 5 & 7 & 4 \end{bmatrix},$$

$$M^4 = \begin{bmatrix} 11 & 13 & 9 & 11 & 14 \\ 13 & 26 & 13 & 19 & 19 \\ 9 & 13 & 11 & 14 & 11 \\ 11 & 19 & 14 & 19 & 14 \\ 14 & 19 & 11 & 14 & 19 \end{bmatrix},$$

$$M^5 = \begin{bmatrix} 24 & 45 & 27 & 38 & 33 \\ 45 & 64 & 45 & 58 & 58 \\ 27 & 45 & 24 & 33 & 38 \\ 38 & 58 & 33 & 44 & 52 \\ 33 & 58 & 38 & 52 & 44 \end{bmatrix},$$

κ.ο.κ.

Στην $M^3 = [p_{ij}]$ έχουμε $p_{13} = 3$, άρα υπάρχουν τρεις διαδρομές μήκους 3 από την v_1 ως την v_3. (Πράγματι, είναι οι (v_1, v_2, v_5, v_3), (v_1, v_4, v_2, v_3)), (v_1, v_4, v_5, v_3)).
Στην $M^4 = [p_{ij}]$ έχουμε $p_{13} = 9$, άρα υπάρχουν εννέα διαδρομές μήκους 4 από την v_1 ως την v_3.

Παρατήρηση. Ο αριθμός των διαδρομών μήκους το πολύ ν από την κορυφή v_i στην κορυφή v_j είναι ίσο με το στοιχείο a_{ij} της μήτρας $A = \sum_{k=0}^{\nu} M^k = I + M^1 + \cdots + M^\nu$. (Η μήτρα M^0 είναι η ταυτοτική μήτρα I, και την συμπεριλαμβάνουμε στο άθροισμα διότι κάθε κορυφή θεωρείται διαδρομή μήκους 0 με αρχή και τέλος τον εαυτό της.

Απόσταση

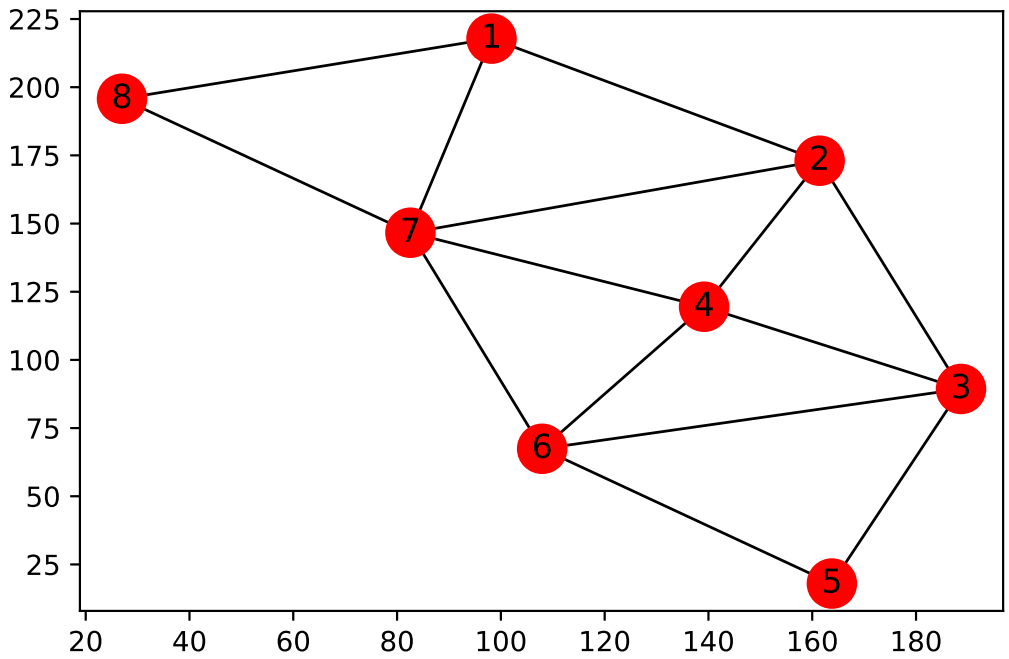
Απόσταση $d(u, v)$ μεταξύ δύο κορυφών u, v μιας συνιστώσας του G ονομάζεται το ελάχιστο μήκος μεταξύ όλων των διαδρομών που τους συνδέουν.
Μερικοί συγγραφείς επεκτείνουν τον παραπάνω ορισμό, ορίζοντας ως απόσταση μεταξύ δύο κορυφών u, v οι οποίες ανήκουν σε διαφορετικές συνιστώσες ενός μη συνεκτικού γραφήματος το ∞.
Γεωδαιτικό ή **γεωδεσικό** ή **συντομότερο** λέγεται κάθε $u - v$ μονοπάτι ενός γραφήματος G, με μήκος ίσο με $d(u, v)$.
Διάμετρος $d(G)$ ενός συνεκτικού γραφήματος G λέγεται το μήκος του μεγαλύτερου γεωδεσικού του, (δηλαδή η μεγαλύτερη απόσταση ανάμεσα σε όλα τα δυνατά ζεύγη κορυφών). Αν το γράφημα δεν είναι συνεκτικό τότε η διάμετρος του ισούται με ∞.

Η εύρεση της απόστασης ανάμεσα σε δύο κόμβους u, v ενός γραφήματος μπορεί να γίνει χρησιμοποιώντας την αναζήτηση σε πλάτος, ή τους αλγορίθμους του Dijkstra, ή των Bellman - Ford. (Οι δύο τελευταίοι αλγόριθμοι μπορούν να δώσουν απαντήσεις και σε γραφήματα που έχουν βάρη πάνω στους δεσμούς τους.)
Η βιβλιοθήκη `networkx` έχει τις μεθόδους `shortest_path(G,v,u)` και `shortest_path_length(G,v,u)` που υπολογίζουν ένα γεωδαιτικό μονοπάτι μεταξύ των κορυφών v και u και την απόσταση των v και u αντίστοιχα.
Επιπρόσθετα, υπάρχουν οι μέθοδοι `all_pairs_shortest_path(G)` και `all_pairs_shortest_path_length(G)` που υπολογίζουν ένα γεωδαιτικό μονοπάτι ανάμεσα σε όλα τα ζεύγη κορυφών και τα αντίστοιχα μήκη τους.

```
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np

G = nx.Graph()
V = [1,2,3,4,5,6,7,8]
E = [[1,2],[1,7],[1,8],[2,3],[2,4],[2,7],[3,4],[3,5],[3,6],
     [4,6],[4,7],[5,6],[6,7],[7,8]]

G.add_nodes_from(V)
G.add_edges_from(E)
pos = nx.nx_agraph.graphviz_layout(G)
nx.draw_networkx(G,pos)

v, u = 1, 4
print("The distance between nodes",v,"and",u,"is:",nx.
    shortest_path_length(G,v,u))

shortestpaths = dict(nx.all_pairs_shortest_path(G))
for v in G:
    print("A shortest path between",v)
    for u in G:
        print("and",u,"is:",shortestpaths[v][u])

alldistances = dict(nx.all_pairs_shortest_path_length(G))
```

```
n = G.order()
D = np.zeros((n,n)) #create a n x n matrix with zero entries
for v in G:
    for u in G:
        #D has numbering 0...7 while nodes are numbered 1...8
        D[u-1][v-1] = alldistances[v][u]
print("The distances between all nodes pairs of G are:")
print(D)

plt.show()
```

Output:



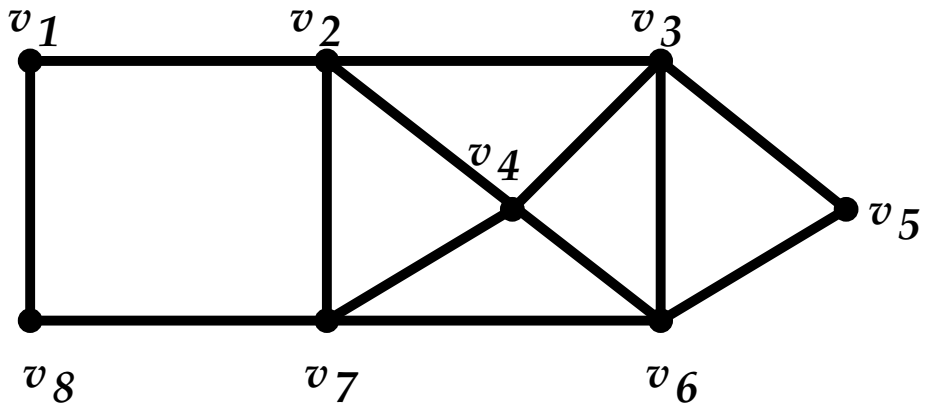
```
The distance between nodes 1
      and 4 is: 2
A shortest path between 1
and 1 is: [1]
and 2 is: [1, 2]
and 3 is: [1, 2, 3]
and 4 is: [1, 2, 4]
and 5 is: [1, 2, 3, 5]
and 6 is: [1, 7, 6]
and 7 is: [1, 7]
and 8 is: [1, 8]
A shortest path between 2
and 1 is: [2, 1]
and 2 is: [2]
and 3 is: [2, 3]
and 4 is: [2, 4]
and 5 is: [2, 3, 5]
and 6 is: [2, 3, 6]
and 7 is: [2, 7]
and 8 is: [2, 1, 8]
A shortest path between 3
and 1 is: [3, 2, 1]
and 2 is: [3, 2]
and 3 is: [3]
and 4 is: [3, 4]
and 5 is: [3, 5]
and 6 is: [3, 6]
and 7 is: [3, 2, 7]
and 8 is: [3, 2, 1, 8]
A shortest path between 4
and 1 is: [4, 2, 1]
and 2 is: [4, 2]
and 3 is: [4, 3]
and 4 is: [4]
and 5 is: [4, 3, 5]
and 6 is: [4, 6]
and 7 is: [4, 7]
and 8 is: [4, 7, 8]
A shortest path between 5
and 1 is: [5, 3, 2, 1]
and 2 is: [5, 3, 2]
and 3 is: [5, 3]
and 4 is: [5, 3, 4]
and 5 is: [5]
and 6 is: [5, 6]
and 7 is: [5, 6, 7]
and 8 is: [5, 6, 7, 8]
A shortest path between 6
and 1 is: [6, 7, 1]
and 2 is: [6, 3, 2]
and 3 is: [6, 3]
and 4 is: [6, 4]
and 5 is: [6, 5]
and 6 is: [6]
and 7 is: [6, 7]
and 8 is: [6, 7, 8]
A shortest path between 7
and 1 is: [7, 1]
and 2 is: [7, 2]
and 3 is: [7, 2, 3]
and 4 is: [7, 4]
and 5 is: [7, 6, 5]
and 6 is: [7, 6]
and 7 is: [7]
and 8 is: [7, 8]
A shortest path between 8
and 1 is: [8, 1]
and 2 is: [8, 1, 2]
and 3 is: [8, 1, 2, 3]
and 4 is: [8, 7, 4]
and 5 is: [8, 7, 6, 5]
and 6 is: [8, 7, 6]
and 7 is: [8, 7]
and 8 is: [8]
```

```
The distances between all nodes pairs of G are:
[[0. 1. 2. 2. 3. 2. 1. 1.]
 [1. 0. 1. 1. 2. 2. 1. 2.]
 [2. 1. 0. 1. 1. 1. 2. 3.]
 [2. 1. 1. 0. 2. 1. 1. 2.]
 [3. 2. 1. 2. 0. 1. 2. 3.]
 [2. 2. 1. 1. 1. 0. 1. 2.]
 [1. 1. 2. 1. 2. 1. 0. 1.]
 [1. 2. 3. 2. 3. 2. 1. 0.]]
```

Εκκεντρότητα

Μερικές φορές θέλουμε να αποκτήσουμε μια συνολική εικόνα για τις πιθανές αποστάσεις που έχουν τα ζεύγη κόμβων του γραφήματος.
Εκκεντρότητα $e(v)$ μιας κορυφή v ενός συνεκτικού γραφήματος G είναι η μέγιστη απόσταση της v από κάθε άλλη κορυφή u, δηλαδή
$e(v) = \max_{u \in V(G)} d(u, v)$
Παρατήρηση : Προφανώς $d(G) = \max_{v \in V(G)} e(v)$ και $e(v) \leq d(G)$.
Ακτίνα $r(G)$ ενός συνεκτικού γραφήματος G είναι η ελάχιστη εκκεντρότητα, ανάμεσα σε όλους τους κόμβους του G, δηλαδή
$r(G) = \min_{v \in V(G)} e(v)$.
Ο v λέγεται **κεντρικός κόμβος** του συνεκτικού γραφήματος G, αν $e(v) = r(G)$. **Κέντρο** του συνεκτικού γραφήματος ονομάζεται το σύνολο των κεντρικών του κόμβων.
Ο v λέγεται **περιφερειακός κόμβος** του συνεκτικού γραφήματος G, αν $e(v) = d(G)$. **Περιφερειακό σύνολο** του συνεκτικού γραφήματος ονομάζεται το σύνολο των περιφερειακών του κόμβων.

Παράδειγμα



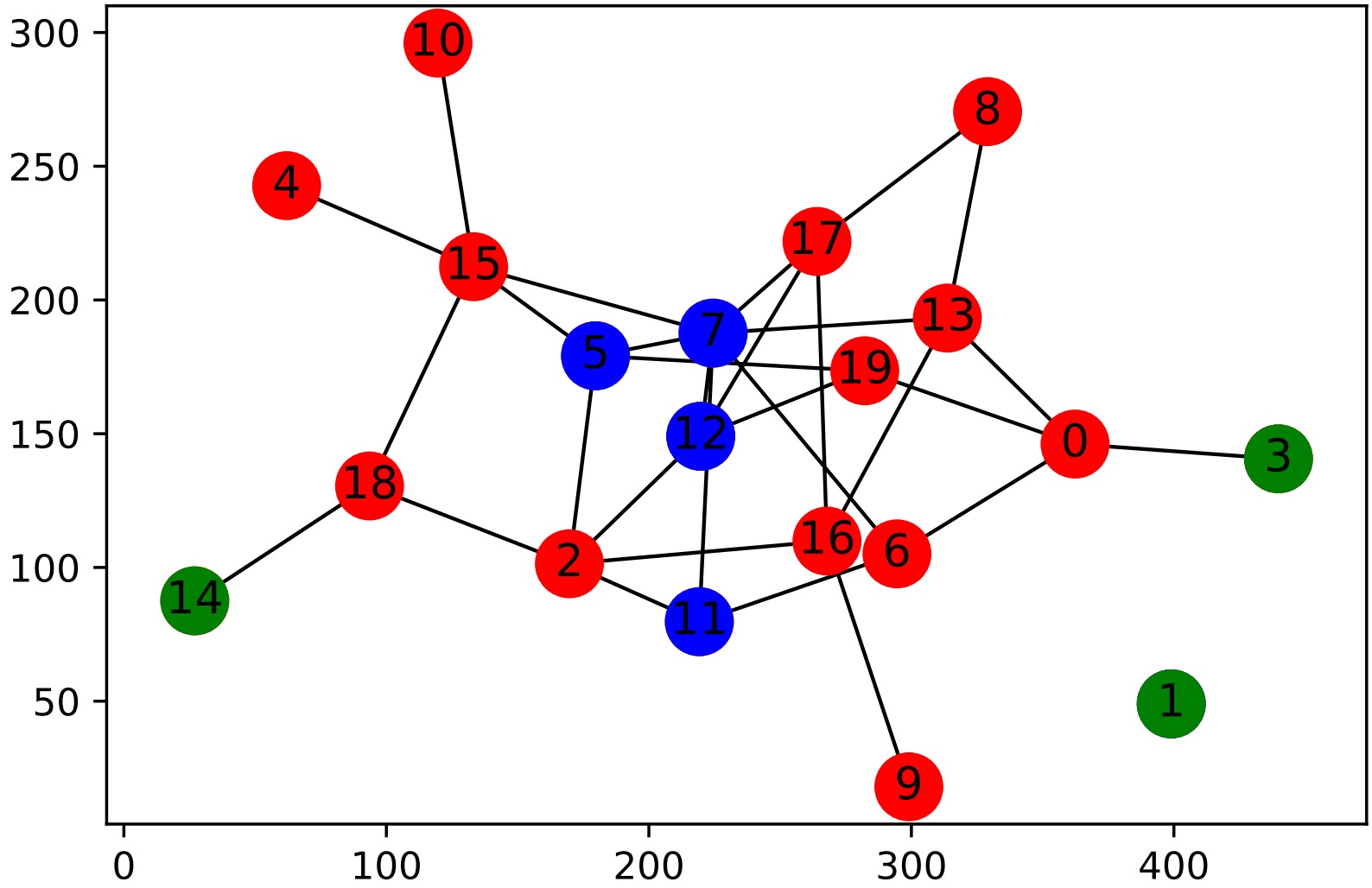
Οι αποστάσεις μεταξύ της κορυφής v_1 και των κορυφών του γραφήματος G είναι: $d(v_1, v_1) = 0$, $d(v_1, v_2) = 1$, $d(v_1, v_3) = 2$, $d(v_1, v_4) = 2$, $d(v_1, v_5) = 3$, $d(v_1, v_6) = 3$, $d(v_1, v_7) = 2$, $d(v_1, v_8) = 1$.
(v_1, v_2, v_4, v_6): γεωδεσικό, (v_1, v_2, v_7, v_6): γεωδεσικό
$(v_1, v_2, v_4, v_7, v_6)$: όχι γεωδεσικό
$d(G) = 3$
$e(v_1) = e(v_3) = e(v_5) = e(v_6) = e(v_8) = 3$
$e(v_2) = e(v_4) = e(v_7) = 2$
$r(G) = 2$
Κέντρο του $G = \{v_2, v_4, v_7\}$.
Περιφερειακό σύνολο του $G = \{v_1, v_3, v_5, v_8\}$.

Η βιβλιοθήκη networkx διαθέτει τις μεθόδους radius(G), diameter(G), center(G) και periphery(G) για τον υπολογισμό των αντίστοιχων εννοιών. Υπολογιστικά όμως συμφέρει να υπολογίσουμε μια φορά τις εκκεντρότητες των κορυφών του G με την μέθοδο eccentricity(G) και έπειτα, με βάση αυτές, τα υπόλοιπα στατιστικά.

```
import networkx as nx
import matplotlib.pyplot as plt

#create a random graph with n nodes and m edges
n,m = 20,30
G = nx.gnm_random_graph(n,m)
pos = nx.nx_agraph.graphviz_layout(G)
nx.draw_networkx(G,pos)

print("G has",nx.number_connected_components(G),"connected
    component(s)")

#for every connected component compute the eccentricities of its
    nodes and then its radius, diameter and center
Gcc = nx.connected_components(G)
```

```
for cc in Gcc:
    G1 = G.subgraph(cc)        #G1 is a connected component
    #find the eccentricities every node in G1
    eccdict = nx.eccentricity(G1)
    for v in G1:  #print the eccentricities of every node in G1
        print("The eccentricity of node",v,"is",eccdict[v])
    all_values = eccdict.values()
    G1rad = min(all_values) #compute the radius of G1
    G1diam = max(all_values) #compute the diameter of G1
    G1center = [] #find center and periphery of G1
    G1periphery = []
    for node in eccdict:
        eccnode = eccdict[node]
        if eccnode == G1rad:
            G1center.append(node)
        if eccnode == G1diam:
            G1periphery.append(node)
    print("The connected component",cc,"has radius",G1rad,"diameter
    ",G1diam,"center",G1center,"and periphery",G1periphery)
    #color blue the central and peripheral nodes of G1
    G2 = G.subgraph(G1center)  #G2 is the induced subgraph of
    Gcenter
    nx.draw_networkx_nodes(G2,pos,node_color='blue',width=3.0)
    G3 = G.subgraph(G1periphery)  #G3 is the induced subgraph of
    Gperiphery
    nx.draw_networkx_nodes(G3,pos,node_color='green',width=3.0)
plt.show()
```

Output:


300
250
200
150
100
50
0
100
200
300
400
10
8
4
17
15
13
7
5
19
12
0
3
18
16
6
2
14
11
1
9

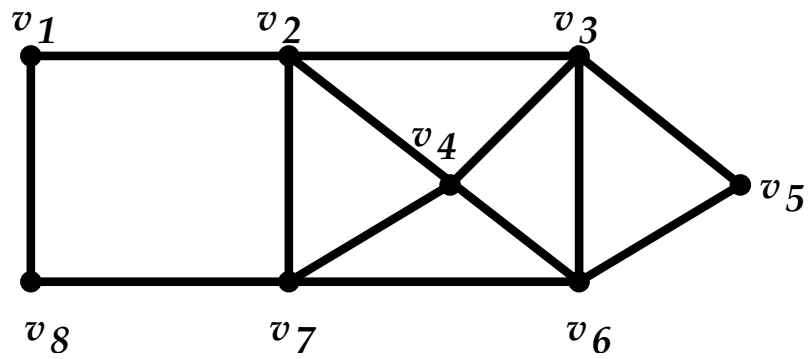
```
G has 2 connected component(s)
The eccentricity of node 0 is 5
The eccentricity of node 2 is 4
The eccentricity of node 3 is 6
The eccentricity of node 4 is 5
The eccentricity of node 5 is 3
The eccentricity of node 6 is 4
The eccentricity of node 7 is 3
The eccentricity of node 8 is 5
The eccentricity of node 9 is 5
The eccentricity of node 10 is 5
The eccentricity of node 11 is 3
The eccentricity of node 12 is 3
The eccentricity of node 13 is 4
The eccentricity of node 14 is 6
The eccentricity of node 15 is 4
The eccentricity of node 16 is 4
The eccentricity of node 17 is 4
The eccentricity of node 18 is 5
The eccentricity of node 19 is 4
The connected component {0, 2,
    3, 4, 5, 6, 7, 8, 9, 10, 11,
     12, 13, 14, 15, 16, 17, 18,
     19} has radius 3 diameter 6
     center [5, 7, 11, 12] and
    periphery [3, 14]
The eccentricity of node 1 is 0
The connected component {1} has
    radius 0 diameter 0 center
    [1] and periphery [1]
```

Παρατήρηση. Αν $d(v_i, v_j) = d$ τότε το στοιχείο (i, j) της μήτρας M^ν ισούται με 0 για κάθε ν με $0 \leq \nu < d$. Άρα, αν ένα γράφημα έχει διάμετρο $d(G)$ τότε για κάθε ν με $0 \leq \nu < d(G)$ θα υπάρχει τουλάχιστον ένα μη μηδενικό στοιχείο στην μήτρα M^ν.

Πρόταση 5 (Υπολογισμός διαμέτρου γραφήματος με χρήση μήτρας)

Η διάμετρος ενός γραφήματος G με μήτρα γειτνίασης M ισούται με τον ελάχιστο φυσικό αριθμό d για τον οποίο όλα τα στοιχεία της μήτρας $A = \sum_{\nu=0}^{d} M^\nu$ είναι μη μηδενικά. Για κάθε συνεκτικό γράφημα με n κορυφές η τιμή του d είναι το πολύ $n - 1$.

Παράδειγμα Η διάμετρος του γραφήματος G



μπορεί να υπολογισθεί χρησιμοποιώντας την παραπάνω πρόταση.

```
import numpy as np

M = np.array([[0, 1, 0, 0, 0, 0, 0, 1],
              [1, 0, 1, 1, 0, 0, 1, 0],
              [0, 1, 0, 1, 1, 1, 0, 0],
              [0, 1, 1, 0, 0, 1, 1, 0],
              [0, 0, 1, 0, 0, 1, 0, 0],
              [0, 0, 1, 0, 1, 0, 1, 0],
              [0, 1, 0, 1, 0, 1, 0, 1],
              [1, 0, 0, 0, 0, 0, 1, 0]])
rows,cols = M.shape
A = np.zeros((rows,cols))

for i in range(rows):
    A += np.linalg.matrix_power(M,i)
    print("A = sum from M**0 to M**",i)
    print(A)
    print("")
    if(np.count_nonzero(A) == rows*cols): #A does not contain any
    zeros
        print("The diameter of G is:", i)
        break
```

Output:

```
A = sum from M**0 to M** 0
[[1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1.]]

A = sum from M**0 to M** 1
[[1. 1. 0. 0. 0. 0. 0. 1.]
 [1. 1. 1. 1. 0. 0. 1. 0.]
 [0. 1. 1. 1. 1. 1. 0. 0.]
 [0. 1. 1. 1. 0. 1. 1. 0.]
 [0. 0. 1. 0. 1. 1. 0. 0.]
 [0. 0. 1. 0. 1. 1. 1. 0.]
 [0. 1. 0. 1. 0. 1. 1. 1.]
 [1. 0. 0. 0. 0. 0. 1. 1.]]
```

```
A = sum from M**0 to M** 2
[[3. 1. 1. 1. 0. 0. 2. 1.]
 [1. 5. 2. 3. 1. 3. 2. 2.]
 [1. 2. 5. 2. 2. 3. 3. 0.]
 [1. 3. 3. 4. 2. 3. 3. 1.]
 [0. 1. 2. 1. 3. 2. 1. 0.]
 [0. 2. 2. 2. 2. 4. 1. 1.]
 [2. 2. 3. 2. 1. 2. 5. 1.]
 [1. 2. 0. 1. 0. 1. 1. 3.]]

A = sum from M**0 to M** 3
[[ 3.  7.  2.  4.  1.  4.  3.  5.]
 [ 7.  9. 12.  9.  5.  8. 13.  3.]
 [ 2. 11. 10. 10.  8. 12.  7.  4.]
 [ 4. 11. 12. 10.  6. 12. 11.  4.]
 [ 1.  4.  7.  4.  5.  7.  4.  1.]
 [ 3.  5. 10.  5.  6.  8.  9.  1.]
 [ 3. 12.  7. 10.  5. 11.  8.  7.]
 [ 5.  3.  4.  3.  1.  2.  7.  3.]]

The diameter of G is: 3
```

Επομένως, η διάμετρος $d(G)$ του G ισούται με 3.

Επειδή όλα τα συνεκτικά γραφήματα με n κορυφές έχουν διάμετρο το πολύ $n-1$, αυτό δίνει ένα απλό κριτήριο για να ελέγξουμε την συνεκτικότητα ενός γραφήματος χρησιμοποιώντας τις δυνάμεις της μήτρας γειτνίασης του.

Πρόταση 6 (Έλεγχος συνεκτικότητας γραφήματος με χρήση μήτρας)

Ένα γράφημα δεσμών G με n κορυφές και μήτρα γειτνίασης M είναι συνεκτικό αν και μόνο αν κάθε στοιχείο της μήτρας $A = \sum_{\nu=0}^{n-1} M^{\nu}$ είναι μη μηδενικό.

Παρατήρηση. Οι δυνάμεις της μήτρας γειτνίασης ενός γραφήματος μπορούν να χρησιμοποιηθούν και για τον υπολογισμό της εκκεντρότητας των κορυφών του και της ακτίνας του.

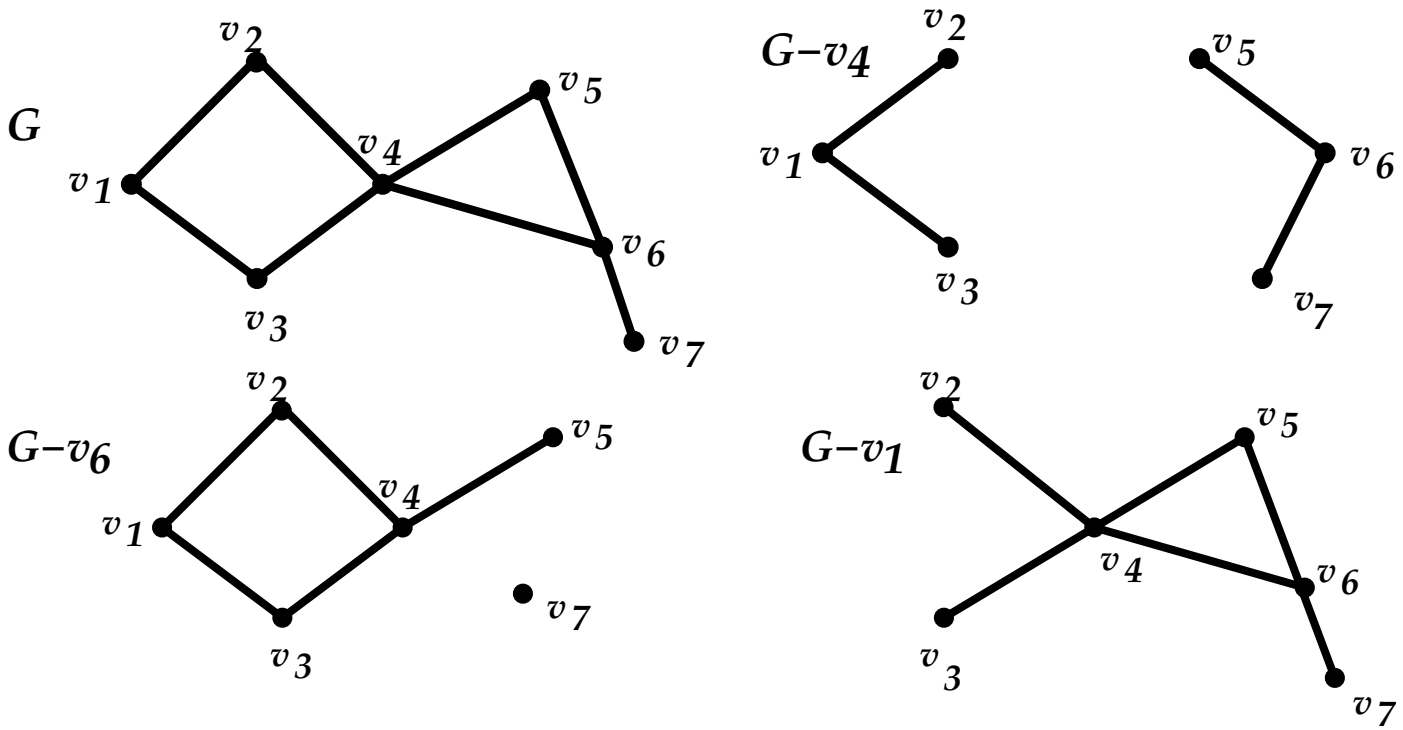
- Η εκκεντρότητα μιας κορυφής v_i ισούται με τον ελάχιστο φυσικό αριθμό d για τον οποίο όλα τα στοιχεία της γραμμής i της μήτρας $A = \sum_{\nu=0}^{d} M^{\nu}$ είναι μη μηδενικά.
- Η ακτίνα ενός γραφήματος G ισούται με τον ελάχιστο φυσικό αριθμό r για τον οποίο όλα τα στοιχεία κάποιας γραμμής της μήτρας $A = \sum_{\nu=0}^{r} M^{\nu}$ είναι μη μηδενικά.

4η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

- **Κλειδώσεις**
- **Γέφυρες**
- **Συμπαγή γραφήματα - Μπλόκ**
- **k-συνεκτικότητα**
- **Γραφήματα Euler**
  - **Αλγόριθμος του Hierholzer**
- **Γραφήματα Hamilton**
- **Διμερή γραφήματα**

Κλειδώσεις

Κλείδωση (ή **σημείο κοπής**) ενός συνεκτικού γραφήματος $G = (V, E)$ λέγεται κάθε $v \in V$ τέτοιο ώστε το $G - v$ είναι μη συνεκτικό.
Παραδείγματα Οι κόμβοι v_4, v_6 του παρακάτω γραφήματος G είναι κλειδώσεις, ενώ ο v_1 δεν είναι.

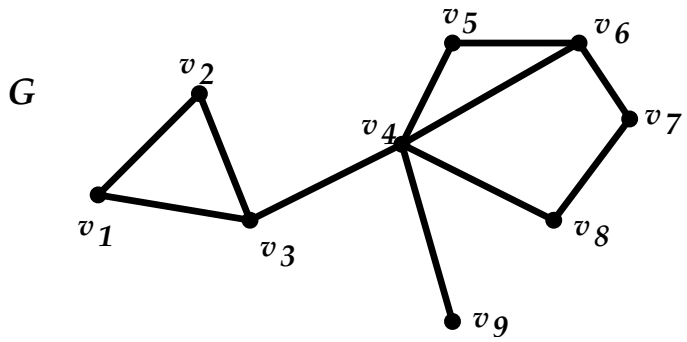


Προφανώς, κλειδώσεις ενός μη συνεκτικού γραφήματος G ονομάζονται οι κλειδώσεις των συνιστωσών του G.

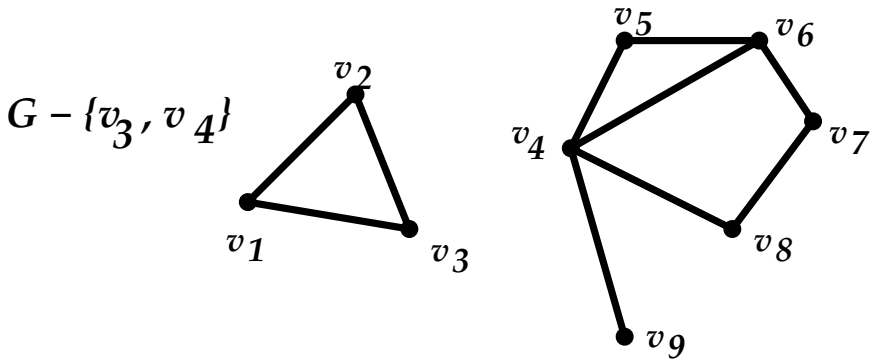
Γέφυρες

Γέφυρα (ή **ισθμός**) ενός συνεκτικού γραφήματος $G = (V, E)$ λέγεται κάθε δεσμός $e \in E$ τέτοιος ώστε το $G - e$ είναι μη συνεκτικό.

Παράδειγμα Για το γράφημα



ο $\{v_3, v_4\}$ είναι γέφυρα, αφού το γράφημα



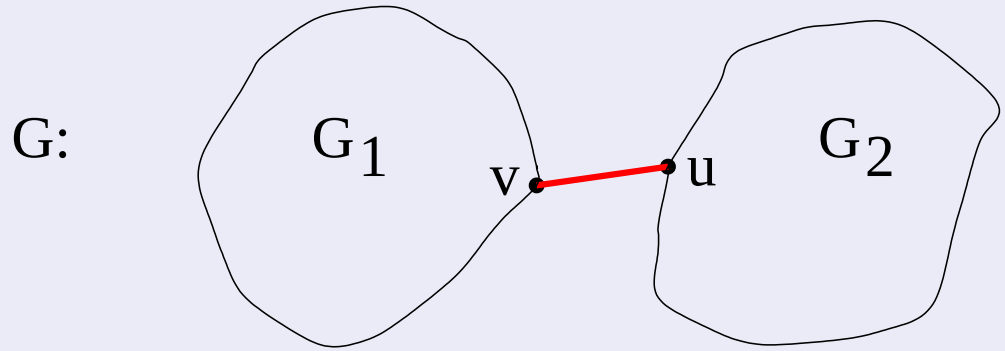
είναι μη συνεκτικό. Προφανώς, γέφυρες ενός μη συνεκτικού γραφήματος G ονομάζονται οι γέφυρες των συνιστωσών του G.

Πρόταση 7

Αν ένα συνεκτικό γράφημα δεσμών G περιέχει μόνο κορυφές με άρτιο βαθμό, τότε δεν περιέχει γέφυρες.

Απόδειξη.

Έστω ότι το G περιέχει την γέφυρα $e = \{v, u\}$.



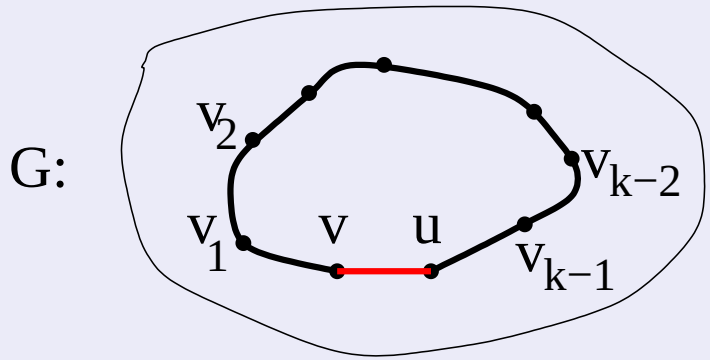
Αν διαγράψουμε από το G την γέφυρα e, στο νέο γράφημα $G - e$ οι κορυφές v, u θα έχουν περιττούς βαθμούς και θα προκύψουν δύο συνεκτικές συνιστώσες G_1, G_2 στις οποίες θα ανήκουν αντίστοιχα τα v, u. Τότε όμως όλες οι υπόλοιπες κορυφές της G_1 θα έχουν άρτιο βαθμό, επομένως το άθροισμα των βαθμών των κορυφών της G_1 θα είναι περιττό, άτοπο. Άρα, το G δεν περιέχει γέφυρα. □

Πρόταση 8

Αν ένα συνεκτικό γράφημα δεσμών G δεν περιέχει γέφυρα, τότε κάθε κορυφή του ανήκει πάνω σε κάποιο κύκλο.

Απόδειξη.

Έστω $e = \{v, u\}$ ένας δεσμός του γραφήματος.



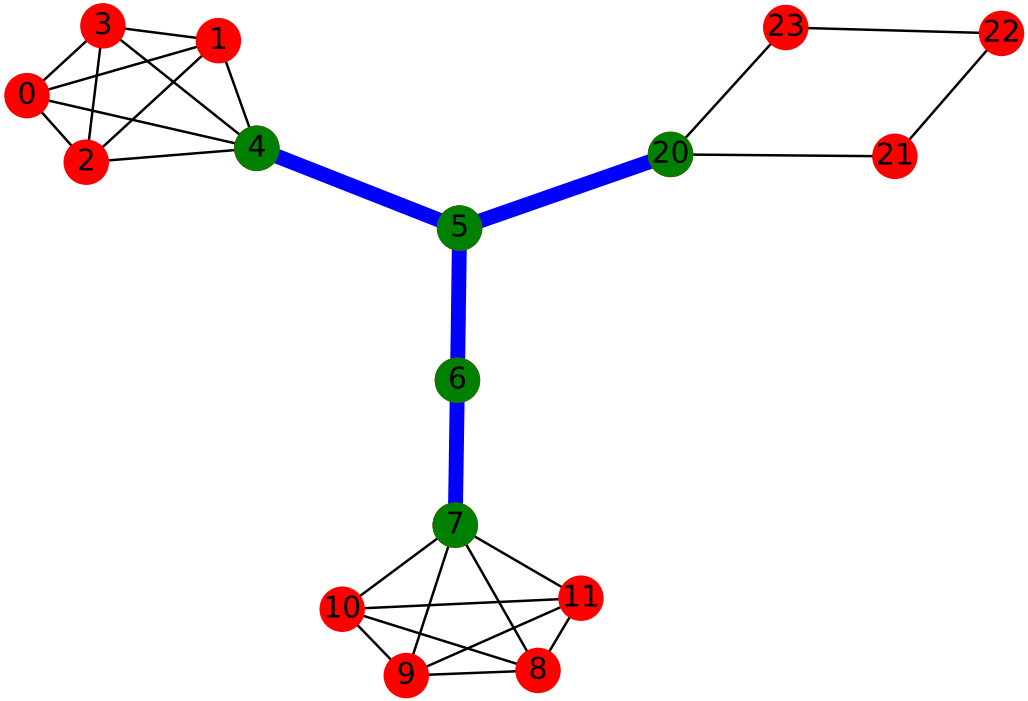
Αφού ο δεσμός e δεν είναι γέφυρα το γράφημα $G - e$ είναι συνεκτικό, άρα υπάρχει μονοπάτι $P = (v, v_1, v_2, \ldots, v_k = u$ στο $G - e$ που συνδέει τις κορυφές v και u. Επομένως, στο G η v ανήκει στον κύκλο $(v, v_1, v_2, \ldots, u, v)$. □

Μπορούμε να εντοπίσουμε τις γέφυρες και τις κλειδώσεις ενός γραφήματος χρησιμοποιώντας τις μεθόδους bridges(G) και articulation_points(G).

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.barbell_graph(5,2)
G.add_edges_from([[5,20],[20,21],[21,22],[22,23],[23,20]])

pos = nx.drawing.nx_agraph.graphviz_layout(G)
nx.draw(G,pos,with_labels=True)
#nx.draw_networkx(G)
B = list(nx.bridges(G))
print("Bridges of G:",B)
for e in B:
    G1 = G.subgraph(e)
    nx.draw_networkx_edges(G1,pos,edge_color='blue',width=6.0)
C = list(nx.articulation_points(G))
print("Cut points of G:",C)
for v in C:
    G1 = G.subgraph(v)
    nx.draw_networkx_nodes(G1,pos,node_color='green',width=6.0)
plt.savefig("barbell52.eps")
plt.show()
```

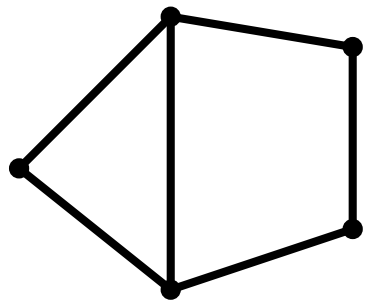
Output:

```
Bridges of G: [(4, 5), (5, 6), (5, 20), (6, 7)]
Cut points of G: [7, 6, 5, 20, 4]
```



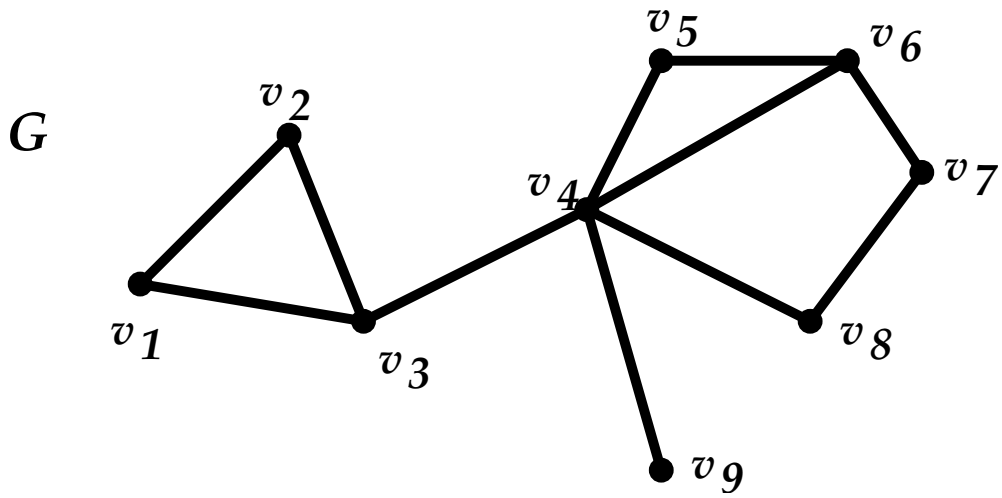
Συμπαγή γραφήματα - Μπλοκ

Ένα μη τετριμμένο, συνεκτικό γράφημα χωρίς κλειδώσεις λέγεται **μη διαχωρίσιμο** (ή **συμπαγές**, ή **δισυνεκτικό**).
Παράδειγμα Το παρακάτω γράφημα είναι μη διαχωρίσιμο:

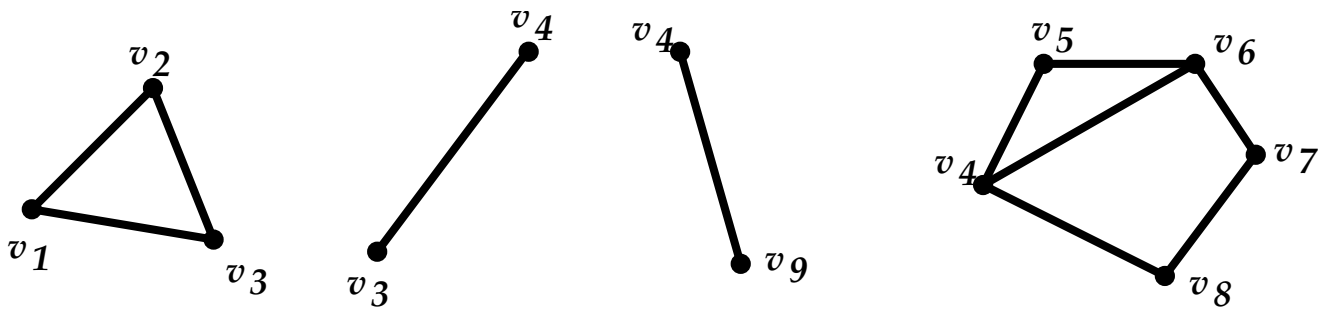


Αν το H είναι ένα μεγιστικό μη διαχωρίσιμο υπογράφημα του G (δηλαδή το H δεν είναι υπογράφημα κάποιου άλλου μη διαχωρίσιμου υπογραφήματος του G) τότε λέγεται **μπλοκ** (ή **δισυνεκτική συνιστώσα**) του G.

Παράδειγμα Τα μπλοκ του γραφήματος



είναι τα

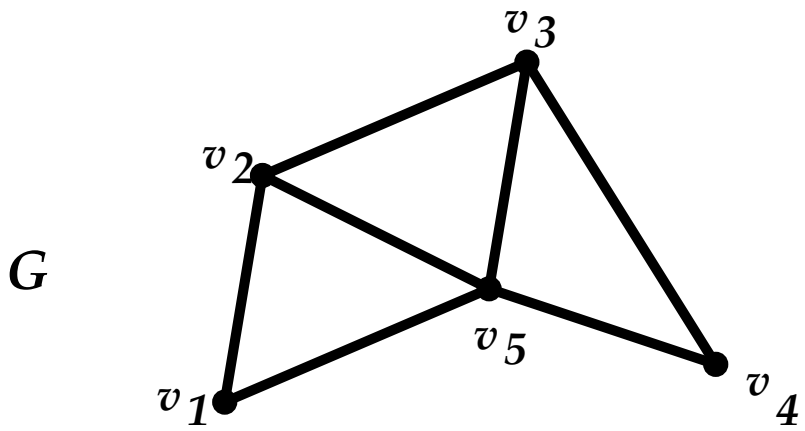


k-συνεκτικότητα

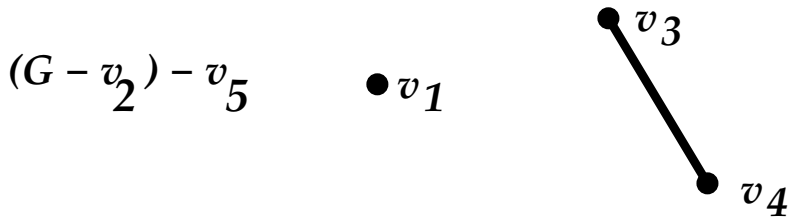
Σύνολο κλειδώσεων ενός συνεκτικού γραφήματος G λέγεται κάθε $\{v_1, v_2, \ldots, v_n\} \subseteq V$ τέτοιο ώστε το $(((G - v_1) - v_2) - \cdots) - v_n$ να είναι μη συνεκτικό.

Ένα γράφημα ονομάζεται k-**συνεκτικό** (k-connected) αν κάθε σύνολο κλειδώσεων του περιέχει τουλάχιστον k κορυφές. Με άλλα λόγια, αν το G είναι k-συνεκτικό τότε το γράφημα που προκύπτει από την διαγραφή οποιουδήποτε συνόλου $k - 1$ κορυφών του G είναι επίσης συνεκτικό.

Παράδειγμα: Το γράφημα



είναι 2-συνεκτικό, αφού το σύνολο $\{v_2, v_5\}$ είναι ένα ελάχιστο σύνολο κλειδώσεων:



Παρατήρηση. Αν ένα γράφημα G είναι $(k+1)$-συνεκτικό, τότε είναι και k-συνεκτικό. Πράγματι, αφού το G είναι $(k+1)$-συνεκτικό η διαγραφή οποιονδήποτε k κορυφών του, δεν το κάνει μη συνεκτικό, άρα ούτε και η διαγραφή $k-1$ κορυφών οδηγεί σε μη συνεκτικό γράφημα, οπότε το G είναι και k-συνεκτικό. Δεν ισχύει το αντίστροφο, αν ένα γράφημα είναι k-συνεκτικό τότε δεν είναι $(k+1)$-συνεκτικό.

Στην επόμενη πρόταση δίδεται μια απλή αναγκαία και ικανή συνθήκη ώστε ένα γράφημα να είναι 2-συνεκτικό.

Πρόταση 9

Ένα γράφημα είναι 2-συνεκτικό αν και μόνο αν για κάθε ζεύγος κορυφών του υπάρχει τουλάχιστον ένα κύκλος που τις περιέχει.

Παράδειγμα : Στο προηγούμενο 2-συνεκτικό γράφημα οι κορυφές v_2 και v_4 ανήκουν στον κύκλο $(v_2 v_5 v_4 v_3 v_2)$, οι κορυφές v_2 και v_1 ανήκουν στον κύκλο (v_1, v_2, v_5, v_1), οι κορυφές v_2 και v_3 ανήκουν στον κύκλο $(v_2, v_1, v_5, v_4, v_3, v_3)$, κ.ο.κ.

Παρατήρηση. Αν ένα γράφημα είναι 2-συνεκτικό τότε για κάθε ζεύγος κορυφών του υπάρχουν 2 τουλάχιστον διαφορετικά μονοπάτια που τους συνδέουν τα οποία, εκτός από τα άκρα τους, περιέχουν διαφορετικές κορυφές το καθένα.

Γενικότερα έχει αποδειχθεί η παρακάτω πρόταση.

Πρόταση 10 (Whitney, 1932)

Ένα γράφημα G είναι k-συνεκτικό αν και μόνο αν κάθε ζεύγος κορυφών του u, v υπάρχουν k τουλάχιστον διαφορετικά μονοπάτια που τους συνδέουν τα οποία, εκτός από τα άκρα τους, περιέχουν διαφορετικές κορυφές το καθένα.

Γραφήματα Euler

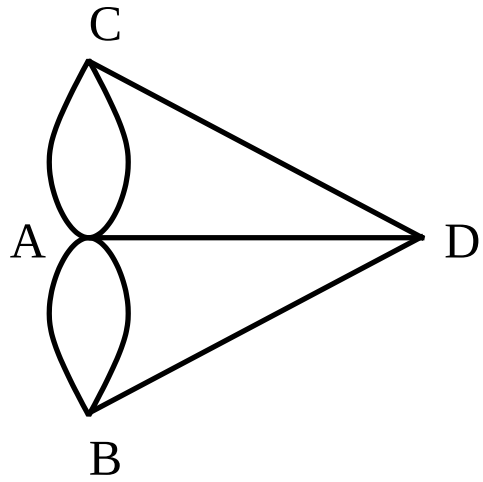
Αν υπάρχει (τουλάχιστον) ένας δρόμος του γραφήματος G, ο οποίος χρησιμοποιεί όλους τους δεσμούς του G, λέγεται **δρόμος Euler**. Αν το G περιέχει ένα κλειστό δρόμο Euler, τότε λέγεται **γράφημα Euler**.

Πρόταση 11

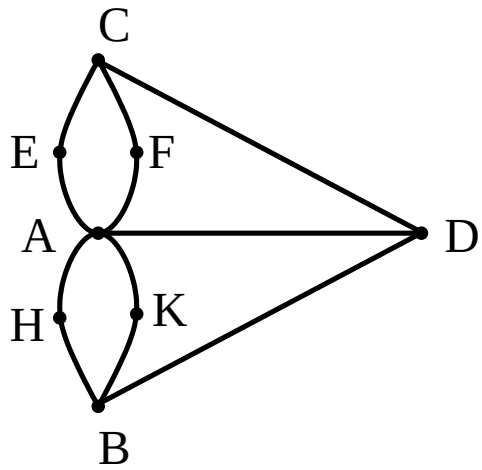
i) *Έστω G ένα συνεκτικό γράφημα που περιέχει τουλάχιστον ένα δρόμο Euler. Τότε περιέχει το πολύ δύο κόμβους περιττού βαθμού. Αν περιέχει δύο τέτοιους κόμβους v_1, v_2, τότε όλοι οι δρόμοι Euler του G είναι $v_1 - v_2$ δρόμοι.*

ii) *Ένα συνεκτικό γράφημα G είναι γράφημα Euler αν και μόνο αν όλοι οι κόμβοι του έχουν άρτιο βαθμό. Στην περίπτωση αυτή, όλοι οι δρόμοι Euler του G είναι κλειστοί.*

Παρατήρηση. Η Πρόταση 11 i) δίνει και την (αρνητική) απάντηση στο πρόβλημα των γεφυρών του Königsberg, που παρουσιάστηκε στην εισαγωγή, αφού το γράφημα

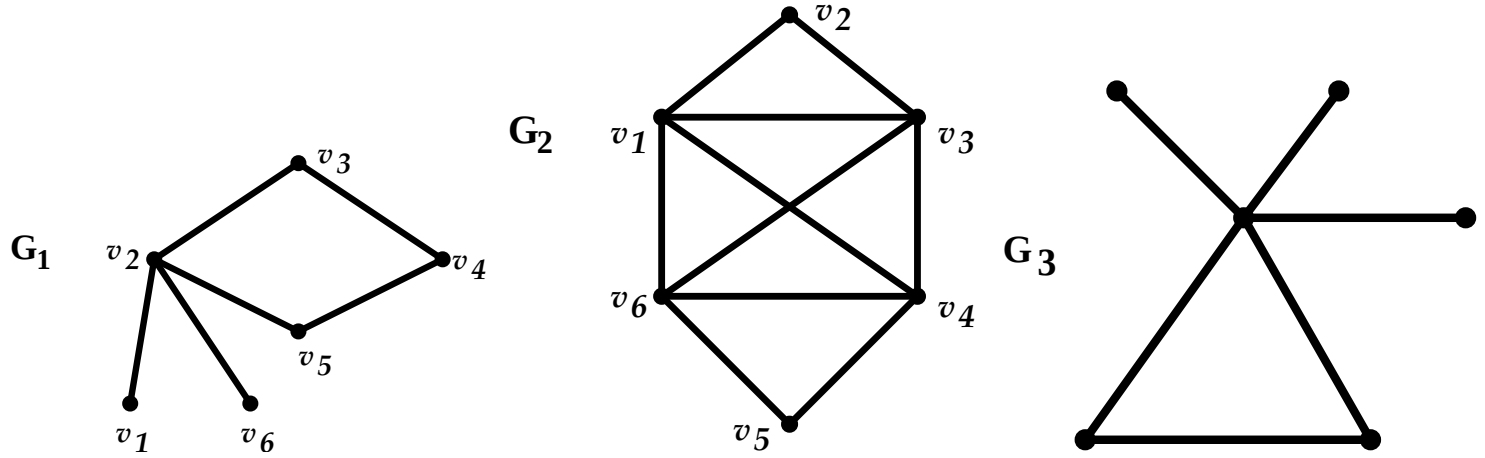


ή, αυστηρότερα, το γράφημα



έχει πάνω από δύο κόμβους περιττού βαθμού.

Παραδείγματα



Το γράφημα G_1 περιέχει το δρόμο Euler

$$(v_1, v_2, v_3, v_4, v_5, v_2, v_6)$$

αλλά δεν είναι γράφημα Euler, (αφού περιέχει και κόμβους περιττού βαθμού).

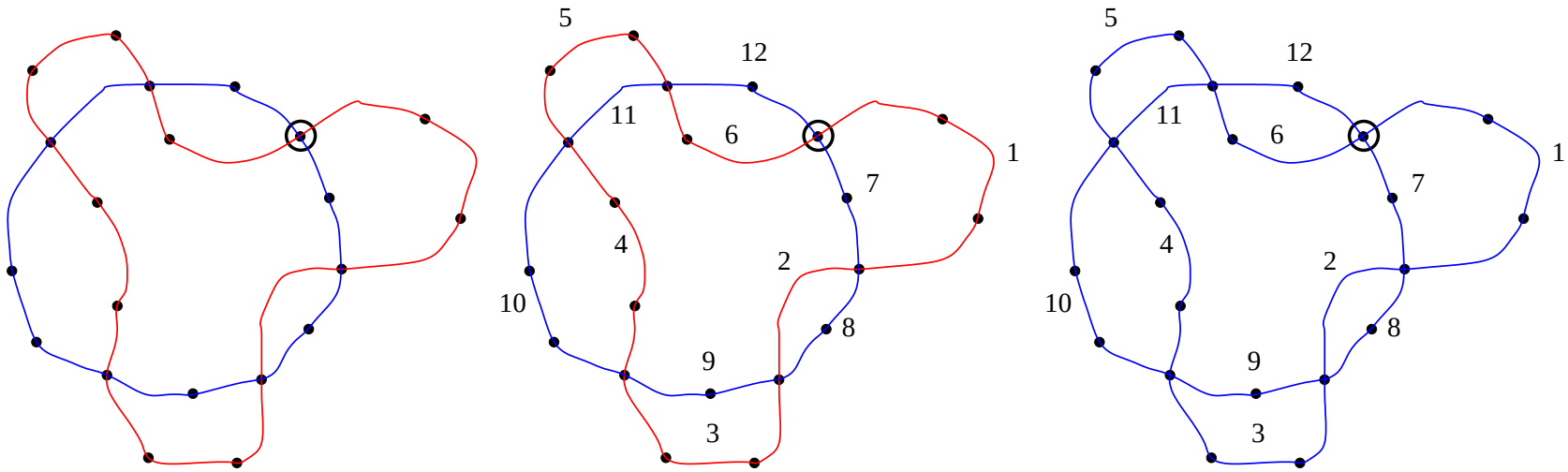
Το γράφημα G_2 (του οποίου όλοι οι κόμβοι έχουν άρτιο βαθμό) είναι γράφημα Euler, και περιέχει για παράδειγμα τον κλειστό δρόμο Euler

$$(v_6, v_1, v_2, v_3, v_4, v_1, v_3, v_6, v_4, v_5, v_6).$$

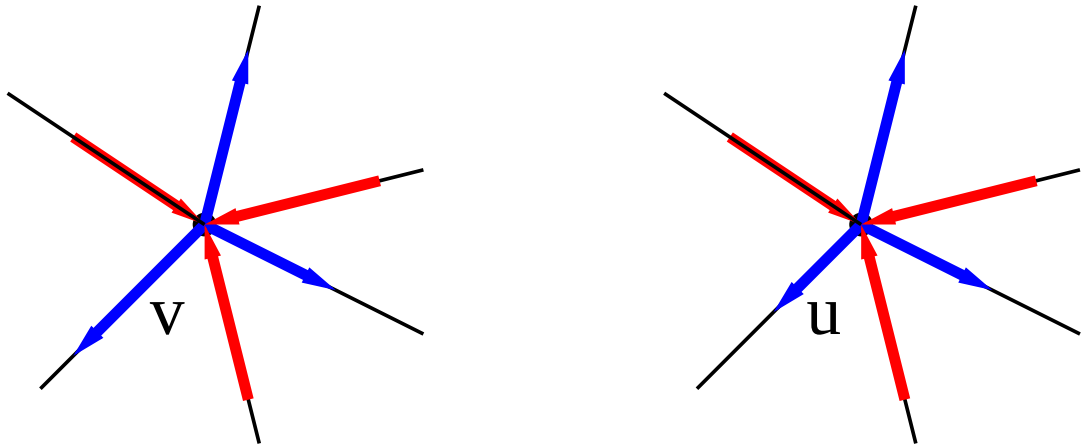
Το γράφημα G_3 δεν περιέχει δρόμο Euler, (αφου περιέχει πάνω από δύο κόμβους περιττού βαθμού.

Για την εύρεση ενός κλειστού δρόμου Euler μπορεί να χρησιμοποιηθεί ο επόμενος αλγόριθμος. Ο αλγόριθμος στηρίζεται σε δύο παρατηρήσεις:

- Αν έχουμε δύο κλειστούς δρόμους που διέρχονται από μια ή περισσότερες κοινές κορυφές και χρησιμοποιούν διαφορετικούς δεσμούς, τότε μπορούμε να τους ενώσουμε σε ένα μεγαλύτερο κλειστό δρόμο.



- Επειδή όλες οι κορυφές έχουν άρτιο βαθμό, κάθε δρόμος που ξεκινά από μια οποιαδήποτε κορυφή v μπορεί πάντα να επιστρέψει στην αρχική κορυφή v ανεξάρτητα από τις επιλογές που γίνονται κατά την διάτρεξή του. Ο λόγος είναι ότι για κάθε δεσμό που μας απομακρύνει από την αρχική v υπάρχει τουλάχιστον ένας δεσμός που μας οδηγεί πάλι σ᾽ αυτή. Αντίθετα σε οποιαδήποτε άλλη κορυφή u δεν μπορούμε να φτάσουμε σε αδιέξοδο αφού για κάθε δεσμό που μας οδηγεί στην u υπάρχει τουλάχιστον ένας δεσμός που μας απομακρύνει από την u.



Αλγόριθμος του Hierholze

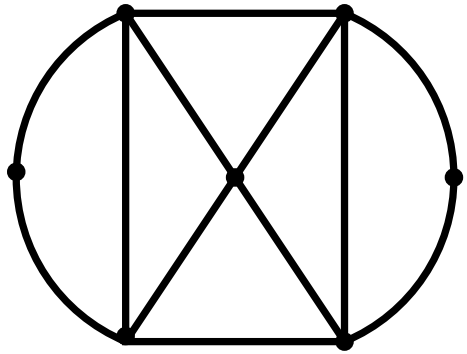
Αλγόριθμος του Hierholzer
Είσοδος: Ένα γράφημα Euler G.
Έξοδος: Ένας κλειστός δρόμος Euler.

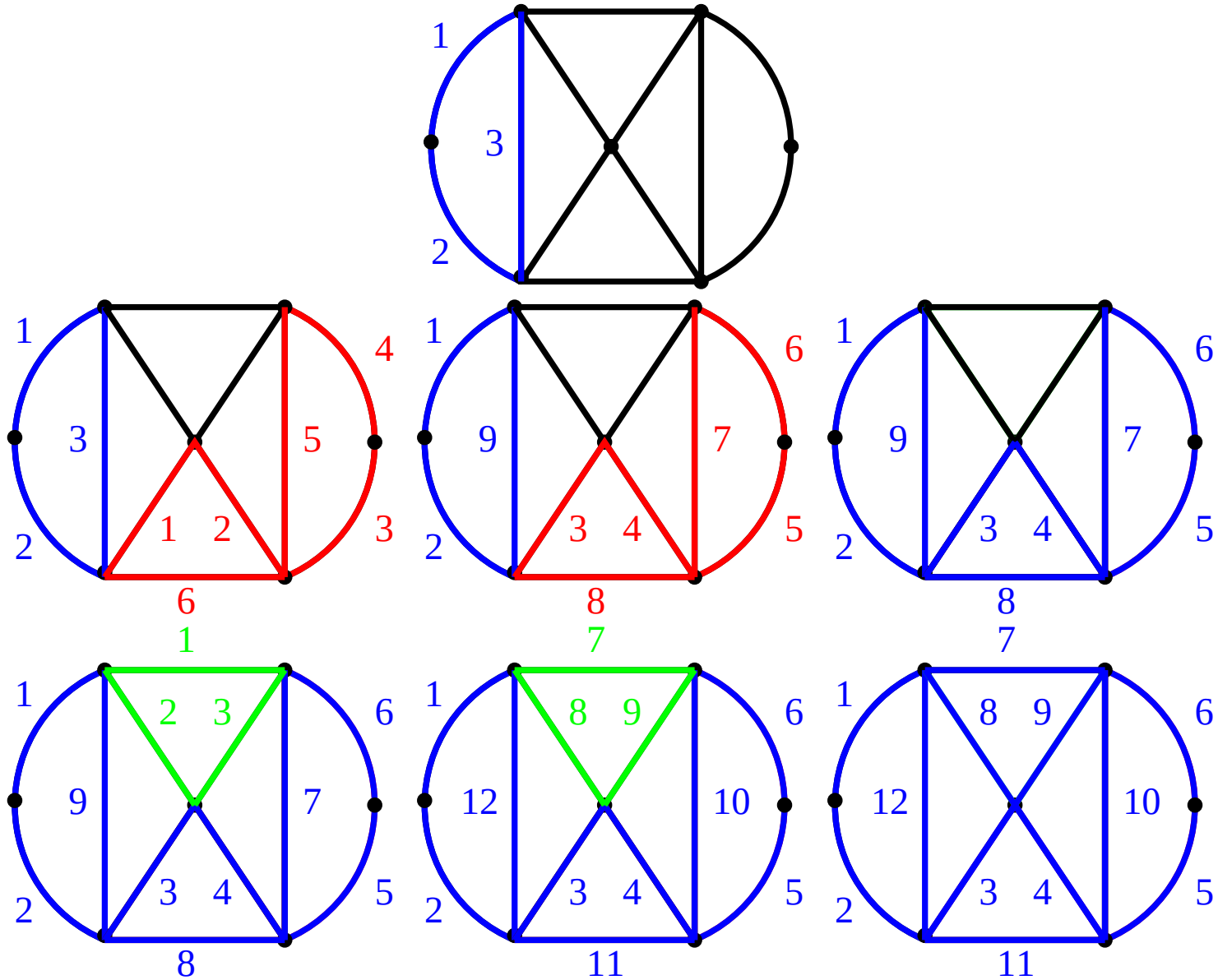
Βήμα 1 Επιλέγουμε οποιαδήποτε κορυφή v η οποία είναι άκρο δεσμού που δεν έχουμε διασχίσει. Κατασκευάζουμε ένα κλειστό δρόμο που περιέχει την v επιλέγοντας αυθαίρετα ένα οποιοδήποτε από δεσμούς που δεν έχουμε ήδη διασχίσει μέχρι να να επιστρέψουμε και πάλι στην v.

Βήμα 2 Αν υπάρχει κορυφή u η οποία είναι άκρο δεσμού που δεν έχουμε συμπεριλάβει στον κλειστό δρόμο W, επαναλαμβάνουμε το Βήμα 1 για την κορυφή u και **ενώνουμε** τους δύο κλειστούς δρόμους που προκύπτουν.

Αφού το G είναι συνεκτικό επαναλαμβάνοντας τα βήματα 1 και 2 θα εξαντλήσουμε όλους τους δεσμούς του γραφήματος και θα δημιουργήσουμε ένα κλειστό δρόμο Euler για το G.

Παράδειγμα Να βρεθεί ένας κλειστός δρόμος Euler για το γράφημα



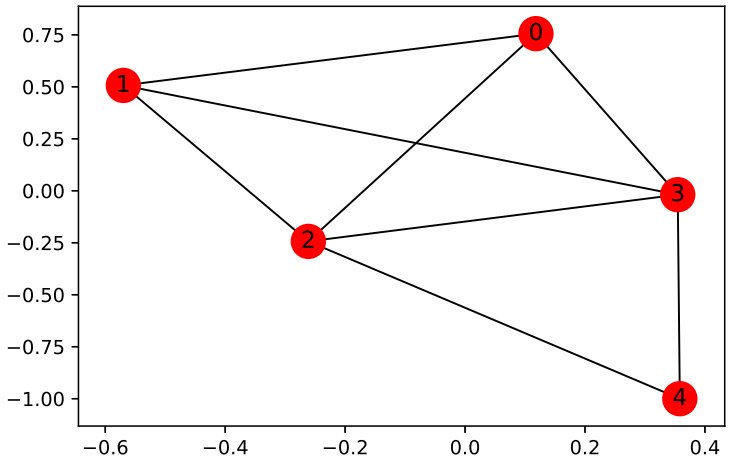

1
3
2
1
4
3
5
1 2
3
6
1
6
9
7
3 4
5
8
1
6
9
7
3 4
5
8
1
1
2 3
6
9
7
3 4
5
8
7
1
8 9
6
12
10
3 4
5
11
7
1
8 9
6
12
10
3 4
5
11

Μπορούμε να βρούμε ένα κλειστό δρόμο Euler σε ένα (συνεκτικό) γράφημα με άρτιους βαθμούς κορυφών χρησιμοποιώντας την μέθοδο `eulerian_circuit(G)`:

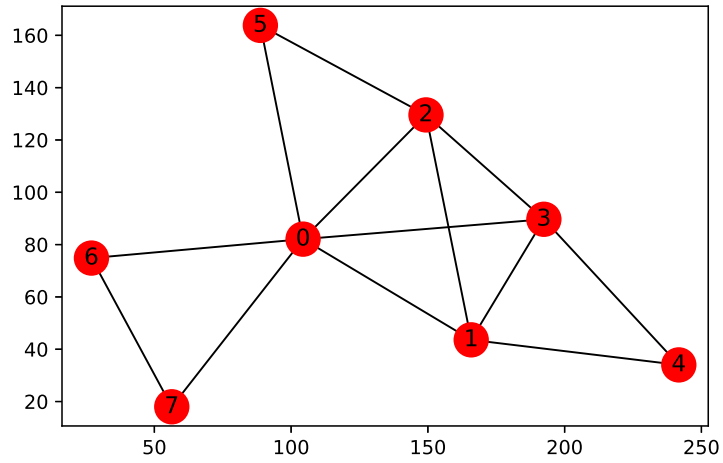
```
import networkx as nx
import matplotlib.pyplot as plt

H = nx.house_x_graph()
nx.draw_networkx(H)
plt.show()
if nx.is_eulerian(H):
    W1 = nx.eulerian_circuit(H)
    print("An Eulerian circuit for H:", list(W1))
else:
    print("H is not Eulerian graph")
```

```
Seq = [6,4,4,4,2,2,2,2]
G = nx.havel_hakimi_graph(Seq)
pos = nx.drawing.nx_agraph.graphviz_layout(G)
nx.draw_networkx(G,pos)
plt.savefig("euler_example0.eps")
if nx.is_eulerian(G):
    #W is an Eulerian circuit for G
    W = nx.eulerian_circuit(G)
    print("An Eulerian circuit for G:", list(W))
    #print("An Eulerian circuit for G:", [u for u, v in W])
    i = 1
    for e in W:
        G1 = G.subgraph(e)
        nx.draw_networkx_edges(G1,pos,edge_color='blue',width=3.0)
        nx.draw_networkx_nodes(G1,pos,node_color='blue',width=3.0)
        plt.savefig("euler_example"+str(i)+".eps")
        nx.draw_networkx_edges(G1,pos,edge_color='red',width=3.0)
        nx.draw_networkx_nodes(G1,pos,node_color='red',width=3.0)
        i += 1
    plt.savefig("euler_example"+str(i)+".eps")
else:
    print("G is not Eulerian graphs")
plt.show()
```

Output:




```
H is not Eulerian graph
```

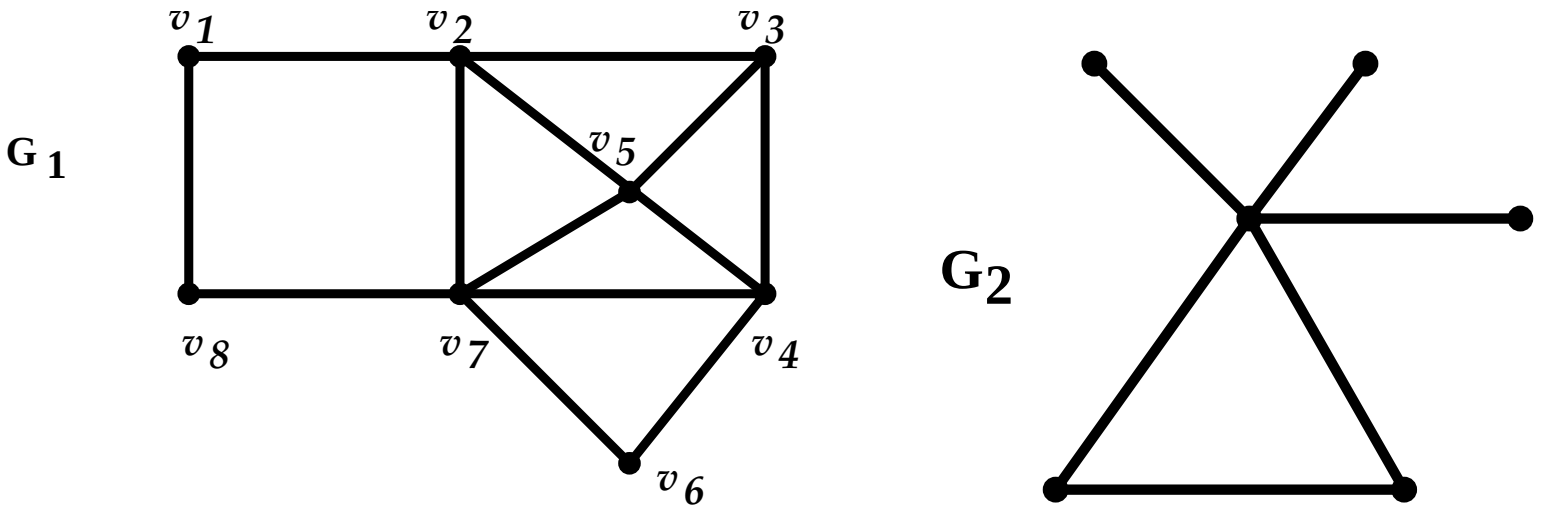



```
An Eulerian circuit for G: [(0,
    6), (6, 7), (7, 0), (0, 5),
    (5, 2), (2, 3), (3, 4), (4,
    1), (1, 2), (2, 0), (0, 1),
    (1, 3), (3, 0)]
```

Γραφήματα Hamilton

Ένας κύκλος του G ο οποίος διέρχεται από όλους τους κόμβους του G λέγεται **κύκλος Hamilton**. Αν το G περιέχει ένα κύκλο Hamilton, λέγεται **γράφημα Hamilton**.

Παραδείγματα



Το γράφημα G_1 είναι γράφημα Hamilton, αφού περιέχει τον κύκλο Hamilton $(v_1, v_2, v_5, v_3, v_4, v_6, v_7, v_8, v_1)$, ενώ το γράφημα G_2 δεν είναι γράφημα Hamilton, αφού προφανώς δεν περιέχει ένα κύκλο Hamilton.

Πρόταση 12

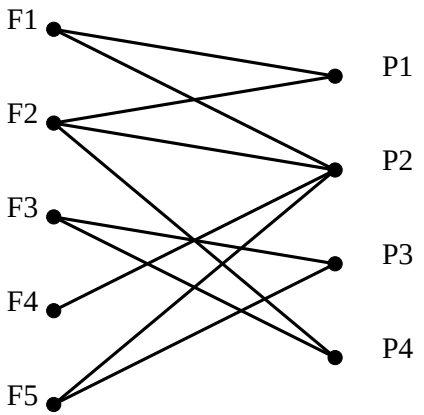
Αν σε ένα απλό γράφημα G *με* $|V(G)| = n \geq 3$, *για κάθε ζεύγος* v, u *μη γειτονικών κορυφών ισχύει ότι*

$$d(v) + d(u) \geq n,$$

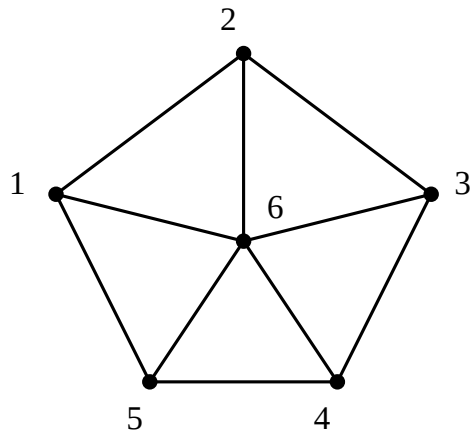
τότε είναι γράφημα Hamilton.

Διμερή γραφήματα

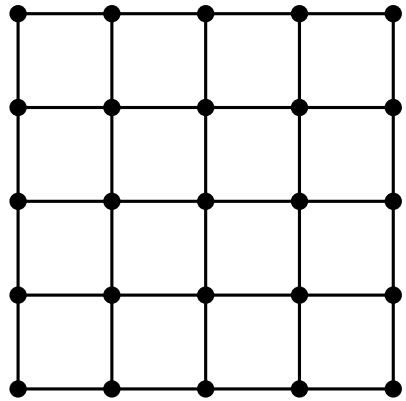
Ένα γράφημα $G = (V, E)$ λέγεται **διμερές** αν το V μπορεί να διαμεριστεί σε δύο υποσύνολα V_1, V_2 τέτοια ώστε κάθε $e \in E$ ενώνει ένα κόμβο του V_1 με ένα κόμβο του V_2.



διμερές



όχι διμερές



διμερές

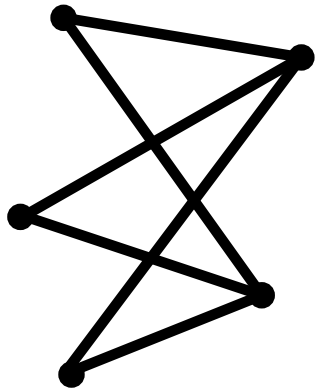
Πρόταση 13

Ένα γράφημα είναι διμερές αν και μόνο αν όλοι οι κύκλοι του είναι άρτιου μήκους.

Πλήρες διμερές γράφημα

Χρησιμοποιήσαμε ήδη το συμβολισμό K_n για το πλήρες γράφημα με n κόμβους.
Με $K_{n,m}$ συμβολίζουμε ένα διμερές γράφημα $G = (X, E)$ με $X = X_1 \cup X_2$, $X_1 \cap X_2 = \emptyset$, $|X_1| = n$, $|X_2| = m$ και τέτοιο ώστε για κάθε $v \in X_1$ και για κάθε $u \in X_2$ να ισχύει ότι $\{v, u\} \in E$. Το $K_{m,n}$ ονομάζεται **πλήρες διμερές γράφημα** (complete bipartite graph).
Παράδειγμα :

$K_{3,2}$



Πρόταση 14

Για κάθε $m, n \in \mathbb{N}^*$ ισχύει ότι $|E(K_{n,m})| = n \cdot m$.

Απόδειξη Έστω ότι το $K_{m,n}$ έχει διαμέριση κορυφών Q_1, Q_2 όπου $|Q_1| = n$ και $|X_2| = m$. Κάθε δεσμός του $K_{m,n}$ έχει ακριβώς ένα άκρο του στο σύνολο X_1. Επομένως, μπορούμε να μετρήσουμε τους δεσμούς με βάση τα άκρα τους στο X_1. Κάθε κορυφή $v \in X_1$ έχει βαθμό m άρα είναι άκρο σε m δεσμούς, άρα συνολικά υπάρχουν $n \cdot m$ δεσμοί.

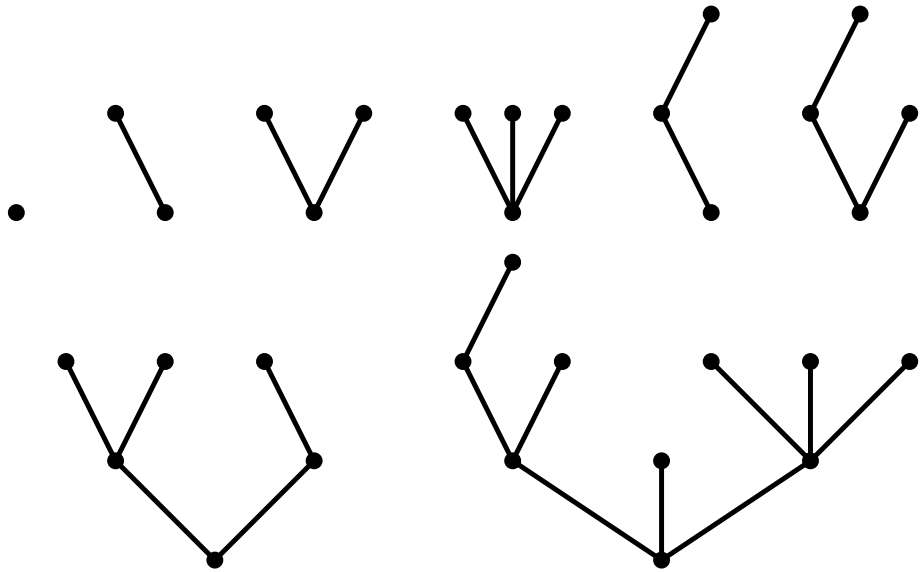
5η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

- **Δένδρα**
- **Δένδρα ζεύξης**
- **Δένδρα με ρίζα**
- **Διατεταγμένα δένδρα**
- **Δυαδικά δένδρα**

Δένδρα

Ένα συνεκτικό άκυκλο γράφημα δεσμών ονομάζεται **δένδρο**.

Παραδείγματα

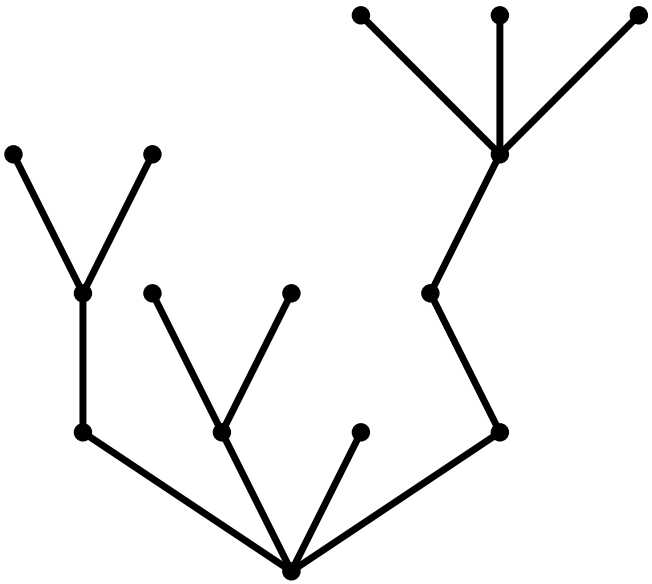


Οι κόμβοι ενός δένδρου με βαθμό 1 λέγονται **φύλλα** του δένδρου.

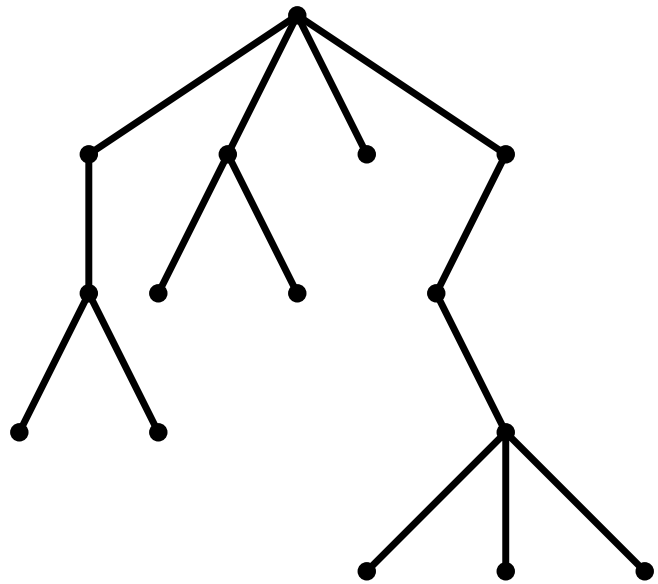
Μέγεθος ενός δένδρου ονομάζεται το πλήθος των δεσμών του.

Ένα άκυκλο γράφημα δεσμών λέγεται **δάσος**.

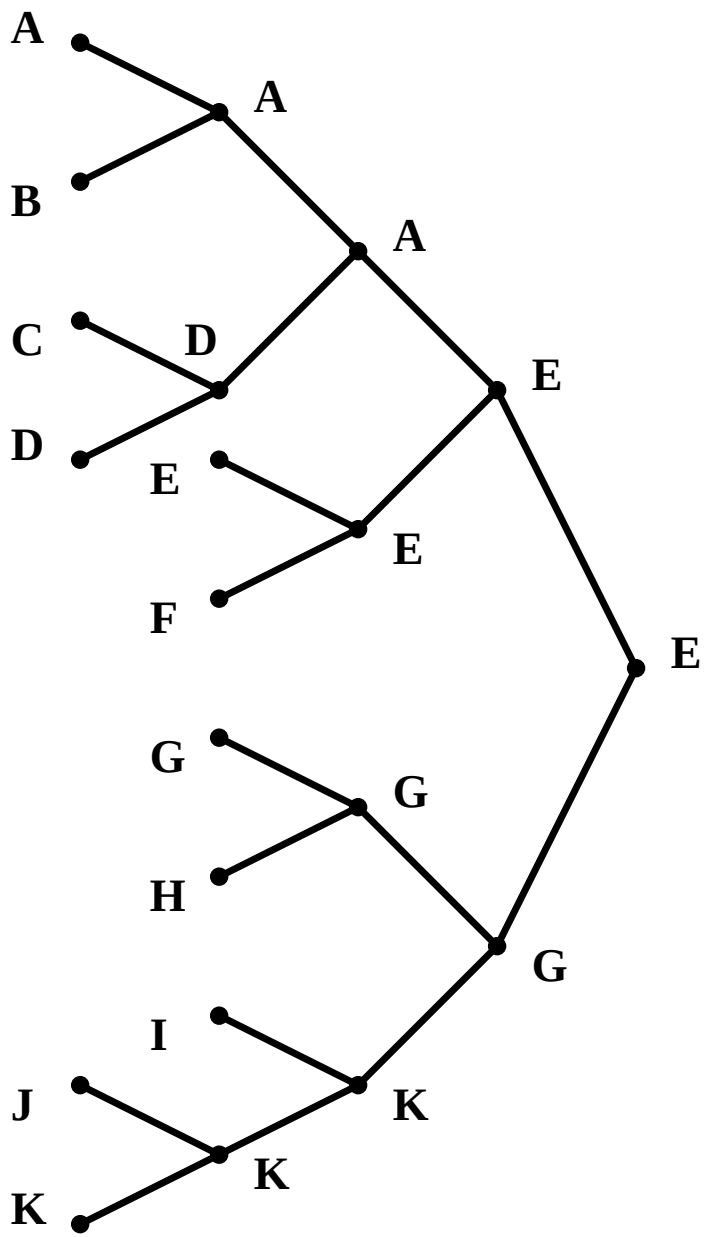
Αναπαράσταση δένδρων



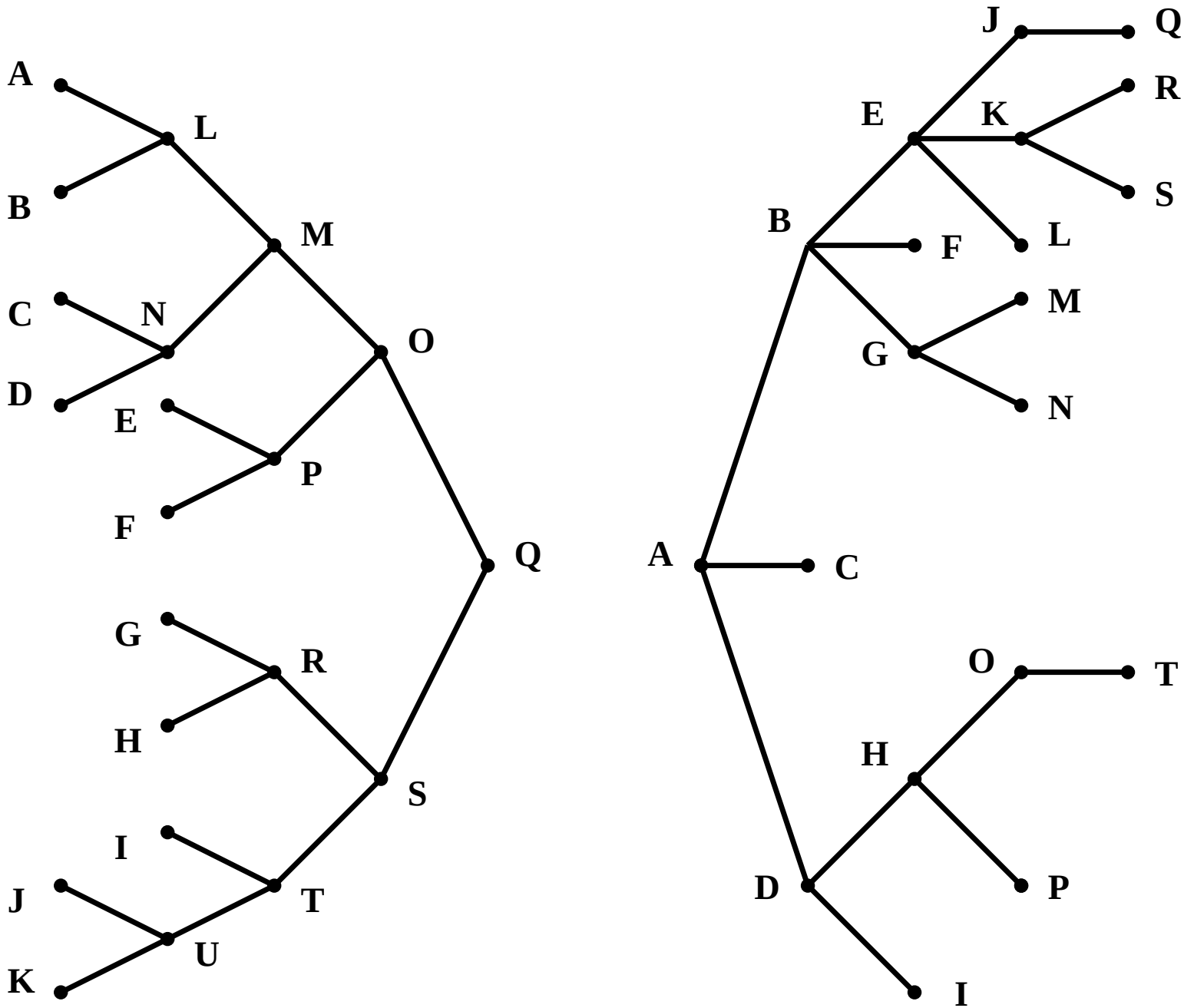
‘‘Φυσικός’’ τρόπος



Στην πληροφορική



Τουρνουά



Γενεαλογικά δένδρα

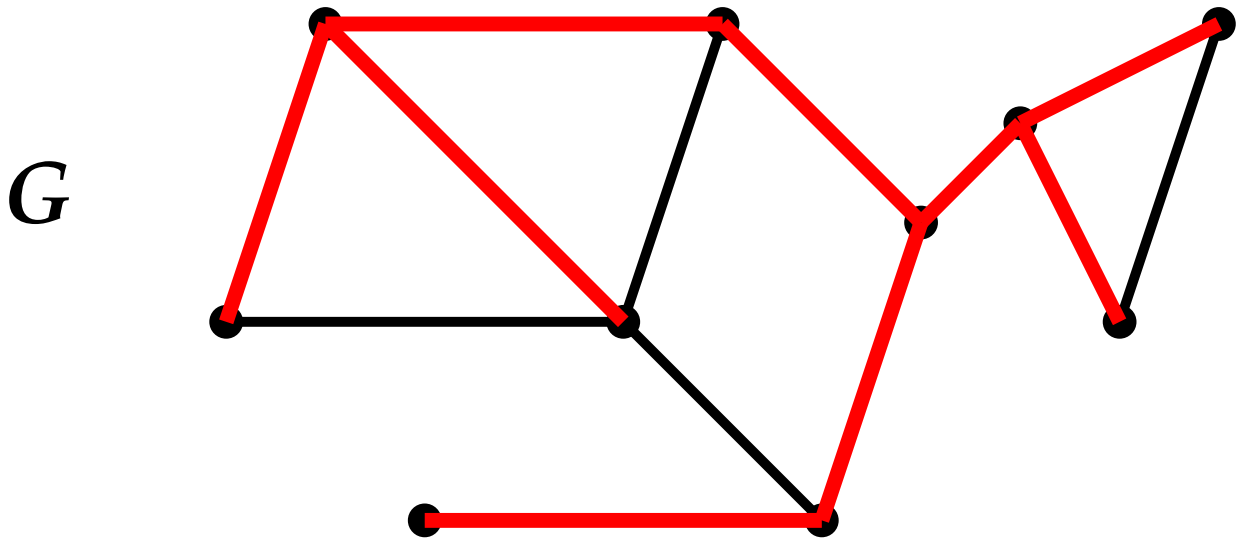
Χαρακτηρισμοί δένδρων

Πρόταση 15

Για ένα γράφημα $G = (V, E)$ οι παρακάτω προτάσεις είναι ισοδύναμες:
1) Το G είναι δένδρο.
2) Κάθε δύο κόμβοι του G ενώνονται με ένα μοναδικό μονοπάτι.
3) Το G είναι συνεκτικό, με $|V| = |E| + 1$.
4) Το G δεν έχει κύκλους και $|V| = |E| + 1$.

Δένδρα ζεύξης

Υπενθυμίζουμε ότι ένα γράφημα $G_1 = (V, E_1)$ λέγεται γενετικό υπογράφημα ενός γραφήματος $G = (V, E)$, αν $E_1 \subseteq E$.
Αν το G_1 είναι δένδρο, τότε έχουμε ένα **γενετικό (ή γεννητικό, ή μερικό) δένδρο**, ή **δένδρο ζεύξης** του G.
Παράδειγμα



Το ''κόκκινο'' γράφημα είναι γενετικό δένδρο του G.

(Φυσικά μπορούμε να βρούμε κι άλλα γενετικά δένδρα για το ίδιο G).

Πρόταση 16

Ένα γράφημα έχει (τουλάχιστον ένα) δένδρο ζεύξης αν και μόνο αν είναι συνεκτικό.

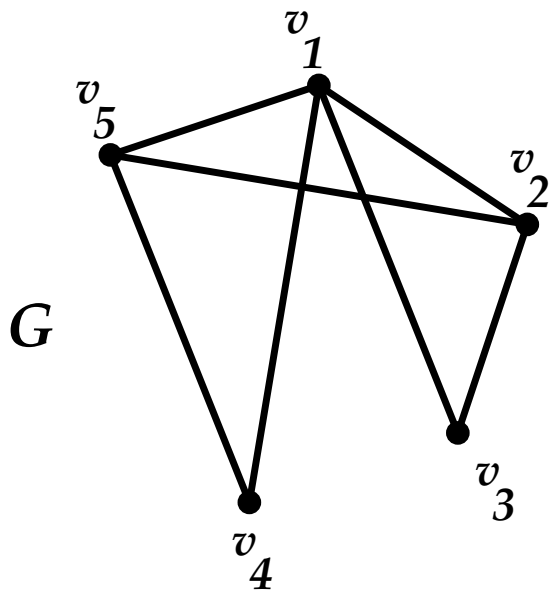
Απαρίθμηση δένδρων ζεύξης

Μήτρα Laplace

Έστω $G = (V, E)$ ένα γράφημα δεσμών με μήτρα γειτνίασης $M(G)$ και η διαγώνια μήτρα $\Delta(G) = \mathrm{diag}(d(v_1), d(v_2), \ldots, d(v_n))$.
Η **Λαπλασιανή μήτρα**, ή **μήτρα Laplace** $L(G)$ του G ορίζεται ως η μήτρα

$$L(G) = \Delta(G) - M(G)$$

Παράδειγμα Για το γράφημα G



έχουμε

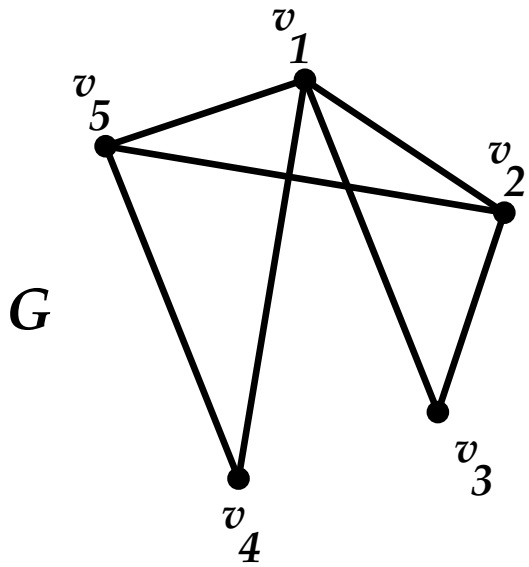
$$L(G) = \Delta(G) - M(G)$$

$$= \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ -1 & -1 & 2 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix}$$

Πρόταση 17 (Θεώρημα μήτρας - δένδρου)

Το πλήθος των δένδρων ζεύξης του G ισούται με την τιμή της ορίζουσας της μήτρας που προκύπτει από την μήτρα Laplace $L(G)$ σβήνοντας την i γραμμή και την i στήλη για οποιοδήποτε $i \in [|V|]$.

Παράδειγμα Το πλήθος των δένδρων ζεύξης του γραφήματος G



ισούται με την τιμή της ορίζουσας $\begin{vmatrix} 3 & -1 & 0 & -1 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 2 & -1 \\ -1 & 0 & -1 & 3 \end{vmatrix}$ που προκύπτει

από την μήτρα $L(G) = \begin{bmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ -1 & -1 & 2 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix}$ σβήνοντας την 1η

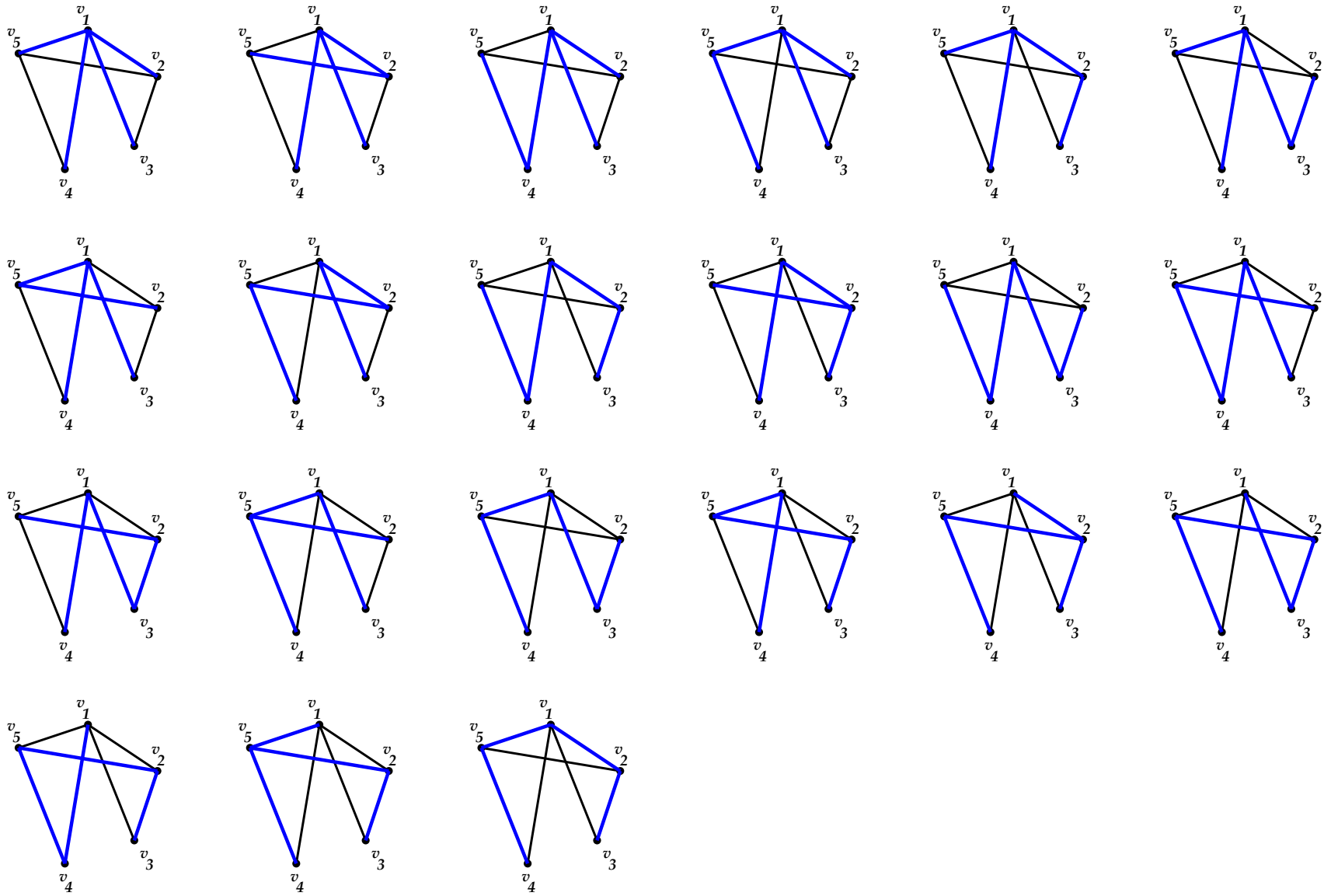
γραμμή και την 1η στήλη της

Συγκεκριμένα έχουμε:

$$\begin{vmatrix} 3 & -1 & 0 & -1 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 2 & -1 \\ -1 & 0 & -1 & 3 \end{vmatrix} \overset{R_3 \to R_3 + 2R_4}{=} \begin{vmatrix} 3 & -1 & 0 & -1 \\ -1 & 2 & 0 & 0 \\ -2 & 0 & 0 & 5 \\ -1 & 0 & -1 & 3 \end{vmatrix} =$$

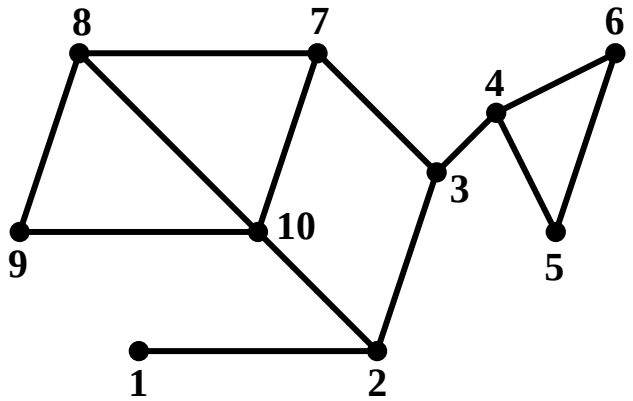
$$(-1)(-1)^{4+3} \begin{vmatrix} 3 & -1 & -1 \\ -1 & 2 & 0 \\ -2 & 0 & 5 \end{vmatrix} \overset{R_3 \to R_3 + 5R_1}{=} \begin{vmatrix} 3 & -1 & -1 \\ -1 & 2 & 0 \\ 13 & -5 & 0 \end{vmatrix} =$$

$$(-1)(-1)^{1+3} \begin{vmatrix} -1 & 2 \\ 13 & -5 \end{vmatrix} = 21 \text{ δένδρα ζεύξης.}$$


v1
v2
v3
v4
v5

Παράδειγμα Για γράφημα

G

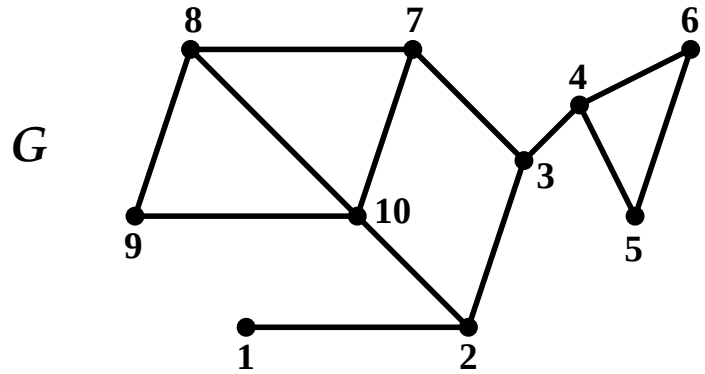


έχουμε $L(G) = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & -1 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 4 \end{bmatrix}$

Το πλήθος των γενετικών δένδρων του G ισούται με την ορίζουσα της μήτρας που προκύπτει από την $L(G)$ σβήνοντας, για παράδειγμα, την 10η γραμμή και 10η στήλη. Οπότε

$$\begin{vmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} = \cdots = 87.$$

Δηλαδή το γράφημα



έχει 87 δένδρα ζεύξης.

Αναλυτικά οι πράξεις υπολογισμού:

$$\begin{vmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} \overset{R_2 \to R_2 + R_1}{=}$$

$$\begin{vmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} =$$

$$1 \begin{vmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} \overset{R_1 \to R_1 + 2 \cdot R_2}{=}$$

$$\begin{vmatrix} 0 & 5 & -2 & 0 & 0 & -2 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} =$$

$$(-1)^{2+1}(-1)\begin{vmatrix} 5 & -2 & 0 & 0 & -2 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{vmatrix} \overset{R_7 \to R_7 + 2 \cdot R_6}{=} \begin{vmatrix} 5 & -2 & 0 & 0 & -2 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & -2 & 5 & 0 \end{vmatrix} =$$

$$(-1)^{6+7}(-1)\begin{vmatrix} 5 & -2 & 0 & 0 & -2 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 2 & 0 & 0 \\ -1 & 0 & 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 & -2 & 5 \end{vmatrix} \overset{R_6 \to R_6 + 5R_5}{=} \begin{vmatrix} 5 & -2 & 0 & 0 & -2 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 2 & 0 & 0 \\ -1 & 0 & 0 & 0 & 3 & -1 \\ -5 & 0 & 0 & 0 & 13 & 0 \end{vmatrix} =$$

$$(-1)^{5+6}(-1)\begin{vmatrix} 5 & -2 & 0 & 0 & -2 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & -1 & -1 & 2 & 0 \\ -5 & 0 & 0 & 0 & 13 \end{vmatrix} \overset{\substack{R_1 \to R_1 + 5R_2 \\ R_5 \to R_5 - 5R_2}}{=} \begin{vmatrix} 0 & 13 & -5 & -5 & -2 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & -1 & -1 & 2 & 0 \\ 0 & -15 & 5 & 5 & 13 \end{vmatrix} =$$

$$(-1)^{2+1}(-1)\begin{vmatrix} 13 & -5 & -5 & -2 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & 0 \\ -15 & 5 & 5 & 13 \end{vmatrix} \overset{\substack{R_1 \to R_1 + 13R_2 \\ R_3 \to R_3 - R_2 \\ R_4 \to R_4 - 15R_2}}{=} \begin{vmatrix} 0 & 21 & -18 & -2 \\ -1 & 2 & -1 & 0 \\ 0 & -3 & 3 & 0 \\ 0 & -25 & 20 & 13 \end{vmatrix} =$$

$$(-1)^{2+1}(-1)\begin{vmatrix} 21 & -18 & -2 \\ -3 & 3 & 0 \\ -25 & 20 & 13 \end{vmatrix} \overset{C_1 \to C_1 + C_2}{=} \begin{vmatrix} 3 & -18 & -2 \\ 0 & 3 & 0 \\ -5 & 10 & 13 \end{vmatrix} = (-1)^{2+2} 3 \begin{vmatrix} 3 & -2 \\ -15 & 13 \end{vmatrix} = 3(3 \cdot 13 - (-2)(-5)) =$$

$3 \cdot 29 = 87$

Πρόταση 18

Ο αριθμός των δένδρων ζεύξης του K_n ισούται με n^{n-2}.

Από το θεώρημα μήτρας-δένδρου έχουμε ότι ο ζητούμενος αριθμός ισούται με την τιμή της ορίζουσας της $(n-1)\times(n-1)$ μήτρας

$$\begin{vmatrix} n-1 & -1 & -1 & \cdots & -1 \\ -1 & n-1 & -1 & \cdots & -1 \\ -1 & -1 & n-1 & \cdots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & -1 & \cdots & n-1 \end{vmatrix} \underset{=}{R_1 = R_1 + \sum_{i=2}^{n-2} R_i}$$

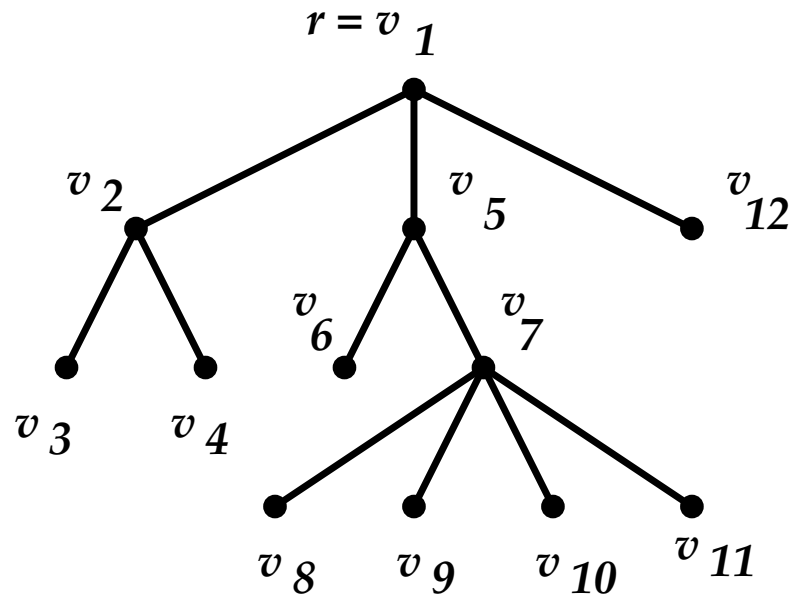
$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ -1 & n-1 & -1 & \cdots & -1 \\ -1 & -1 & n-1 & \cdots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & -1 & \cdots & n-1 \end{vmatrix} \underset{=}{R_i = R_i + R_1, i \geq 2} \begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & n & 0 & \cdots & 0 \\ 0 & 0 & n & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & n \end{vmatrix} = n^{n-2}$$

Παρατήρηση. Υπάρχει εναλλακτικός κομψός τρόπος υπολογισμού του πλήθους των δένδρων ζεύξης του K_n μέσω κωδίκων Prüfer.

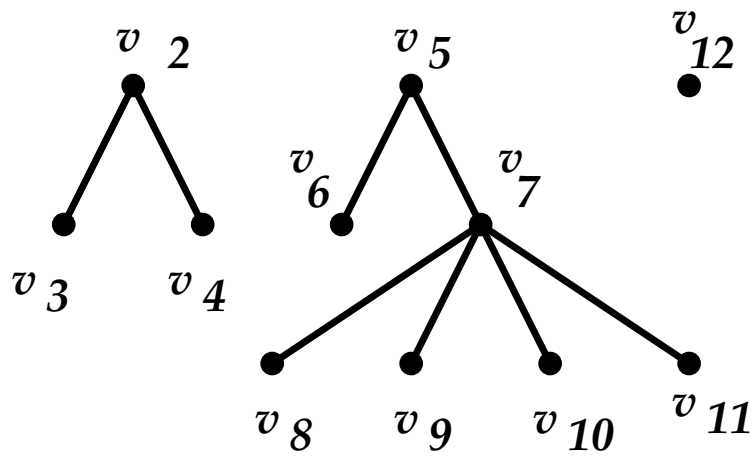
Δένδρα με ρίζα

Δένδρο με ρίζα είναι ένα δένδρο με έναν ειδικά επιλεγμένο κόμβο (τη **ρίζα** του δένδρου).
Έστω r η ρίζα του δένδρου T. Τα δένδρα του δάσους $T - r$ λέγονται **υποδένδρα** (ή **δένδρα–παιδιά**) **της ρίζας** r. Τα δένδρα του δάσους $T - r$ θεωρούνται επίσης δένδρα με ρίζα : Ρίζα καθενός είναι το άλλο άκρο του δεσμού που περιέχει την ρίζα r του T. Η έννοια των υποδένδρων ενός οποιουδήποτε κόμβου (που δεν είναι φύλλο) ορίζεται ανάλογα.

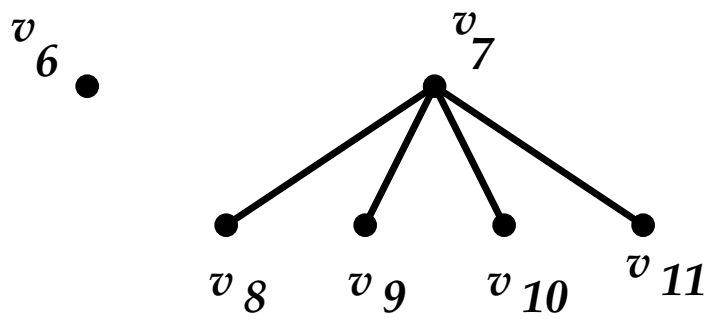
Παραδείγματα



Υποδένδρα της ρίζας v_1 είναι τα



Υποδένδρα του κόμβου v_5 είναι τα



Ορίζουμε το **επίπεδο** $l(v)$ ενός κόμβου v του T ως εξής: $l(r) = 0$ και αν στο (μοναδικό) $r - v$ μονοπάτι $(r, \ldots, u, v)$ έχουμε $l(u) = i$, τότε $l(v) = i + 1$. (Στην περίπτωση αυτή το u λέγεται **γονέας** του v και το v λέγεται **παιδί** του u. Παιδιά του ίδιου γονέα λέγονται **αδέλφια**). Με άλλα λόγια το επίπεδο ενός κόμβου είναι η απόσταση του από την ρίζα του δένδρου.

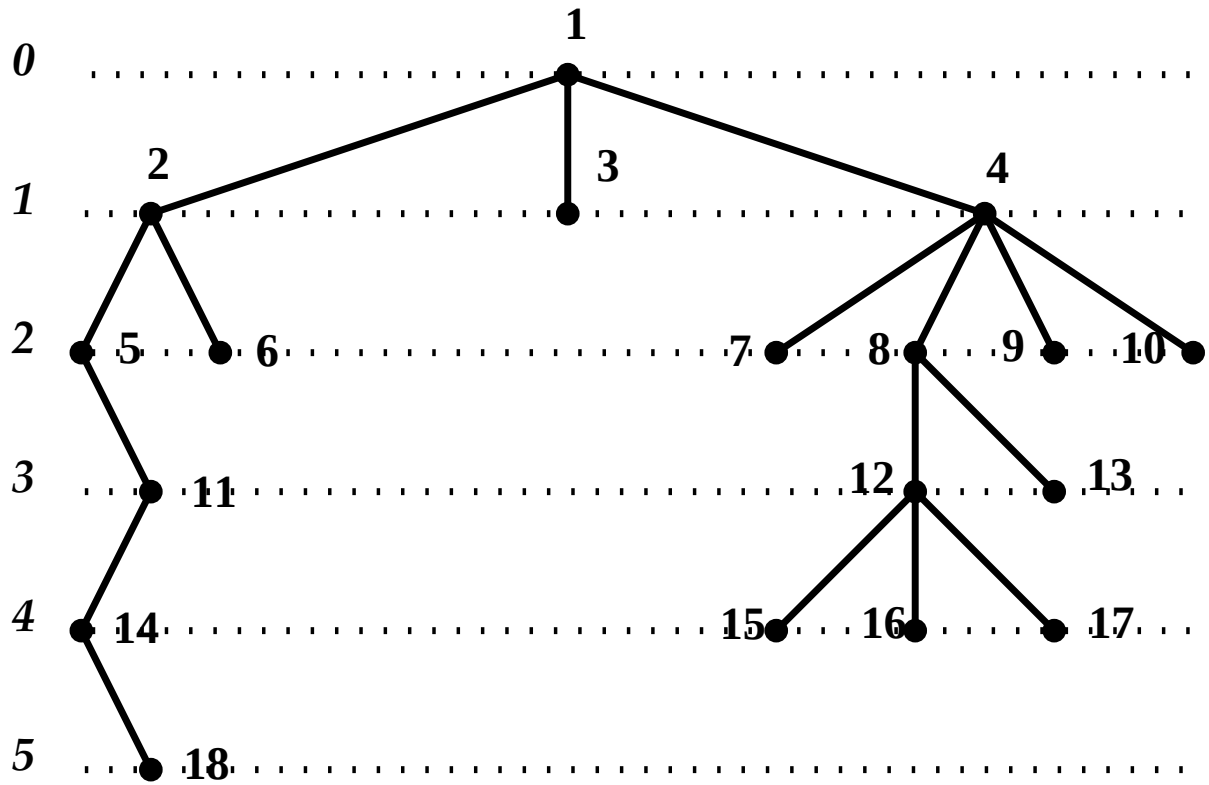
Το πλήθος των παιδιών ενός κόμβου σε ένα δένδρο με ρίζα ονομάζεται **βαθμός** του κόμβου, (οπότε, προφανώς, ο βαθμός κάθε κόμβου, εκτός από τη ρίζα, είναι κατά ένα μικρότερος από το βαθμό του κόμβου, όπως αυτός ορίσθηκε νωρίτερα για ένα τυχαίο γράφημα).

Αν υπάρχει στο T διαδρομή από ένα κόμβο v_1 σε ένα κόμβο v_k, η οποία χρησιμοποιεί κόμβους με επίπεδα που συνεχώς αυξάνουν τότε λέμε ότι το v_1 είναι **πρόγονος** του v_k και το v_k είναι **απόγονος** του v_1.

Ένας κόμβος χωρίς παιδιά (δηλαδή βαθμού 0) λέγεται **φύλλο** (ή **τερματικός κόμβος**). Αλλιώς λέγεται **ενδιάμεσος** κόμβος.

Ύψος (ή **βάθος**) $h(T)$ ενός δένδρου T λέγεται το μεγαλύτερο από τα επίπεδα των κόμβων του.

Παράδειγμα Επίπεδα



1 : ρίζα

5, 6 : παιδιά του 2

7, 8, 9, 10 : παιδιά του 4

2 : πρόγονος των 5,6, 11,14,18

3, 6, 7, 9, 10, 13, 15, 16, 17, 18 : φύλλα

11 : ενδιάμεσος κόμβος

2 : γονέας των 5, 6

4 : γονέας του 7

15, 16, 17 : αδέλφια

15 : απόγονος των 1, 4, 8, 12

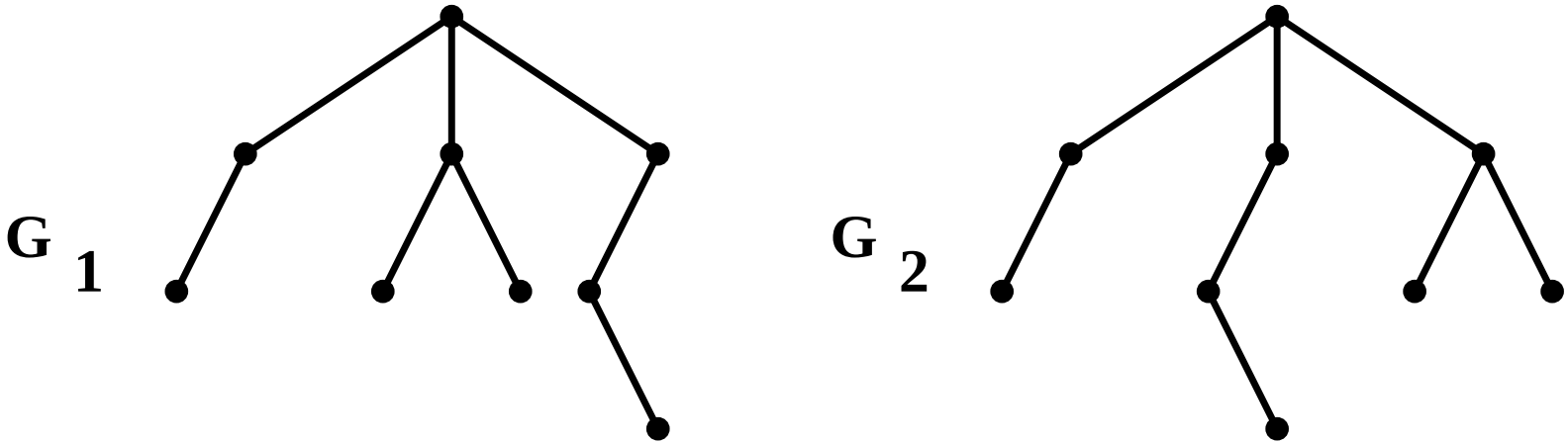
Βαθμός του 12 = 3

$h(T) = 5$.

Διατεταγμένα δένδρα

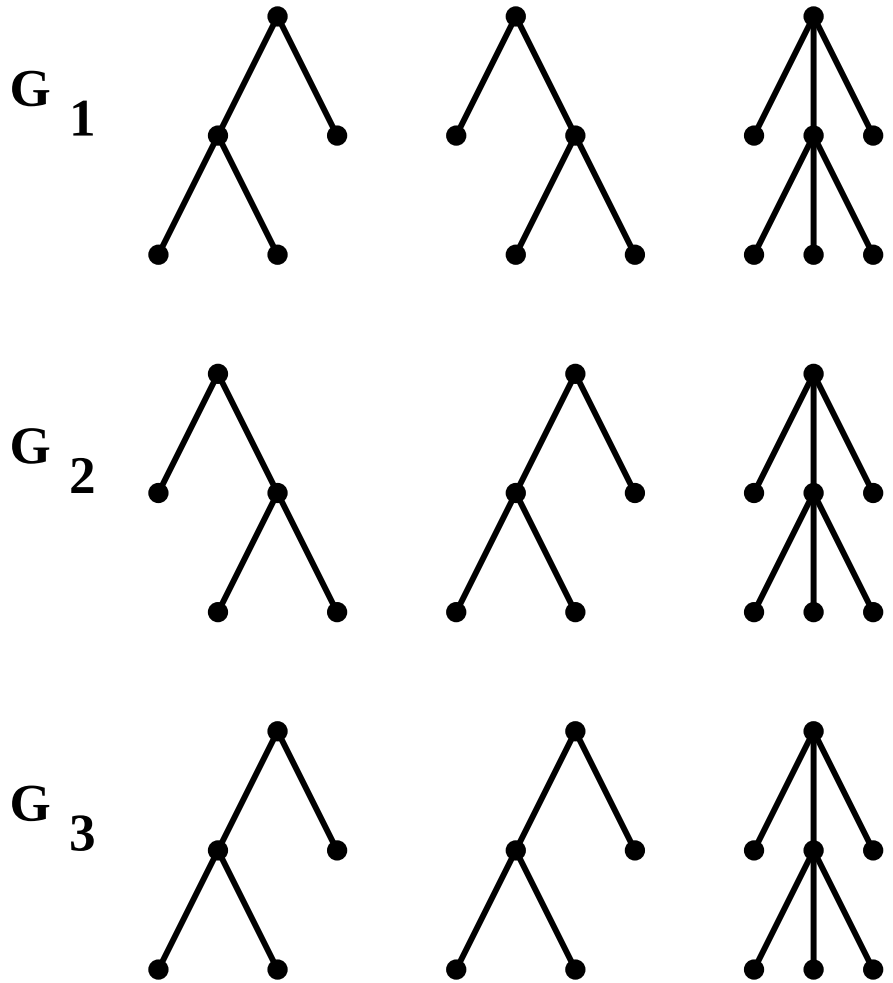
Ένα δένδρο με ρίζα λέγεται **διατεταγμένο** αν η αλλαγή της σχετικής θέσης των υποδένδρων της ρίζας του θεωρείται ότι δημιουργεί μη ισόμορφο δένδρο.

Παράδειγμα



Αν τα G_1, G_2 δεν θεωρηθούν διατεταγμένα τότε $G_1 \simeq G_2$, αλλά τα διατεταγμένα δένδρα G_1, G_2 δεν είναι ισόμορφα. (Σ' ένα διατεταγμένο δένδρο, τα υποδένδρα της ρίζας χαρακτηρίζονται σαν πρώτο, δεύτερο κ.λπ. από αριστερά πρός τα δεξιά).

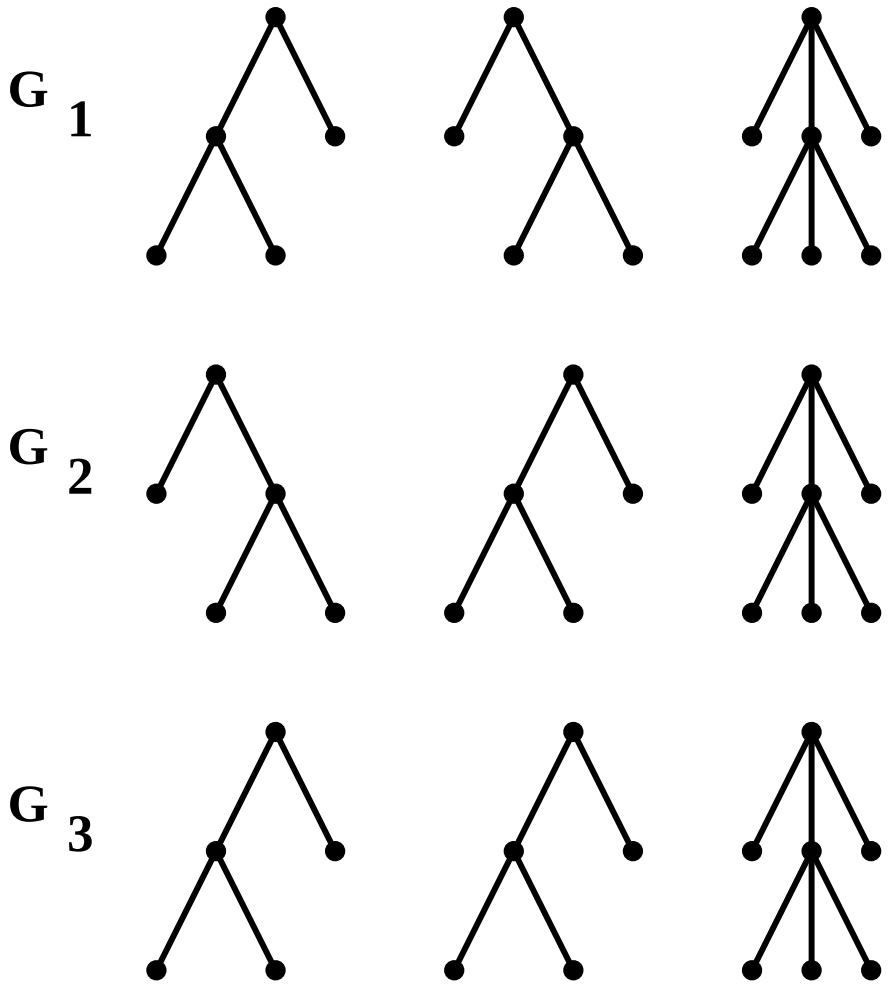
Διατεταγμένο δάσος διατεταγμένων δένδρων είναι ένα διατεταγμένο σύνολο από ξένα, διατεταγμένα δέντρα.



Τα τρία δάση G_1, G_2, G_3 είναι ισόμορφα. Δεν είναι όμως ισόμορφα, αν θεωρηθούν ως διατεταγμένα δάση.

Ένα διατεταγμένο δένδρο, στο οποίο κάθε κόμβος επιτρέπεται να έχει το πολύ k παιδιά, λέγεται k-**δένδρο**.

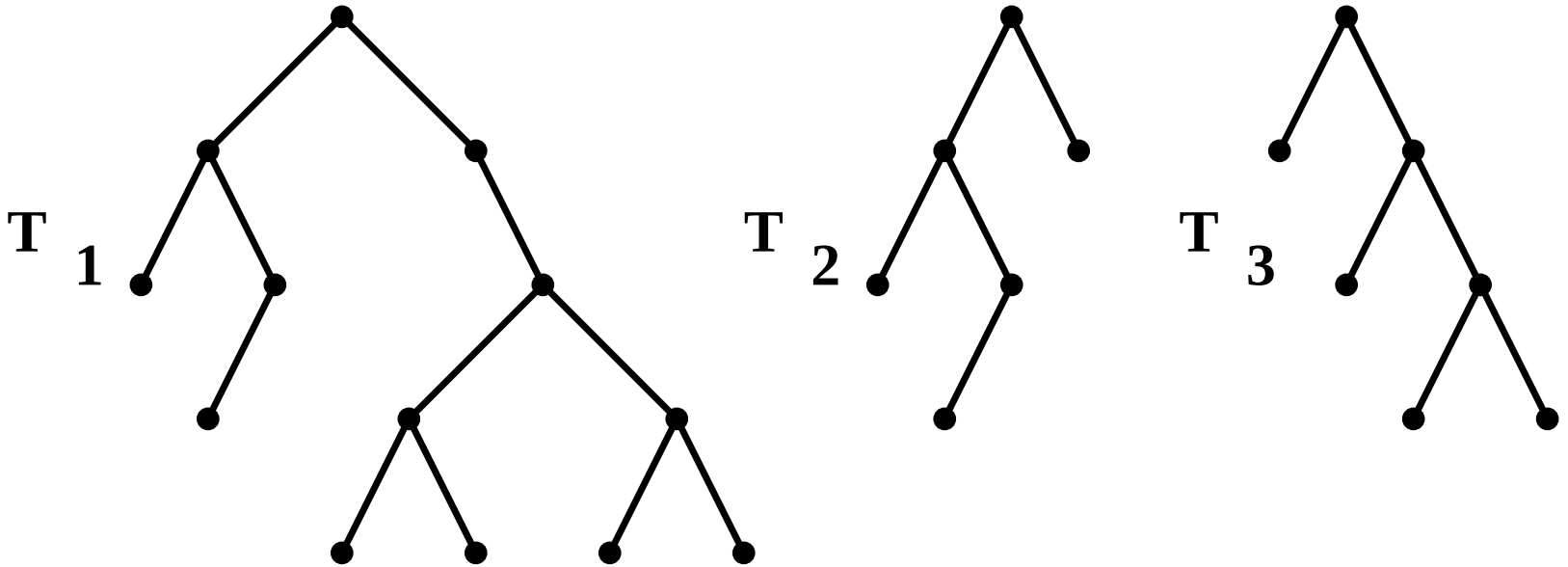
Παράδειγμα



Το πρώτο και το δεύτερο διατεταγμένο δένδρο του διατεταγμένου δάσους G_1 είναι 2-δένδρα, ενώ το τρίτο είναι 3-δένδρο.

Δυαδικά δένδρα

Ένα δένδρο με ρίζα λέγεται **δυαδικό δένδρο** αν κάθε κόμβος του που δεν είναι φύλλο έχει είτε ένα αριστερό, είτε ένα δεξιό παιδί, είτε δύο παιδιά (ένα αριστερό και ένα δεξιό).



Τρία δυαδικά δένδρα T_1, T_2 και T_3.

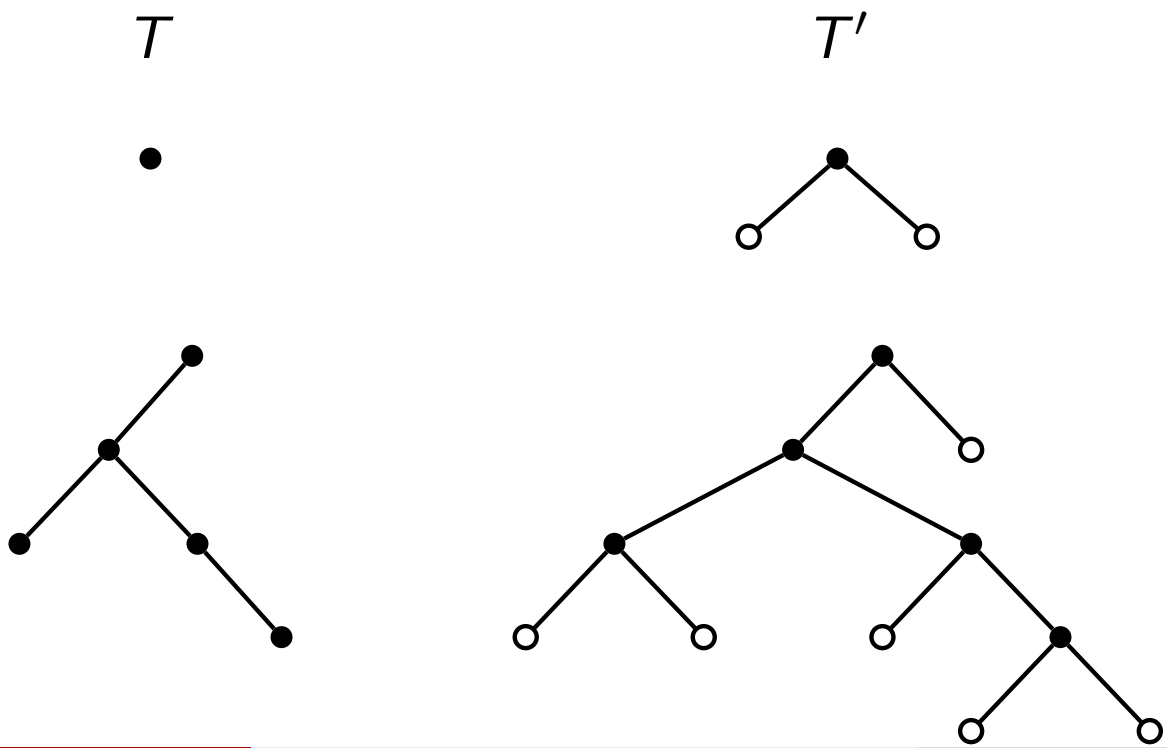
Παρατήρησεις

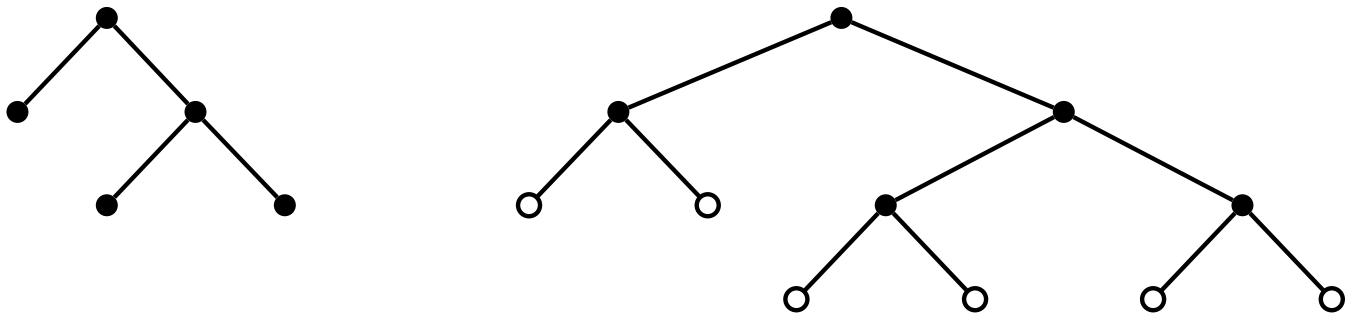
- Σε αντίθεση με τον γενικό ορισμό των γραφημάτων, στα δυαδικά δένδρα συμπεριλαμβάνεται και το κενό δυαδικό δένδρο, δηλαδή το δένδρο T με $X(T) = \emptyset$.
- Ισοδύναμος ορισμός του δυαδικού δένδρου μπορεί να δοθεί αναδρομικά:
  - Το κενό δένδρο είναι δυαδικό.
  - Κάθε μη κενό δυαδικό δένδρο T αποτελείται από ένα κόμβο r που έχει ως παιδιά δύο δυαδικά δένδρα T_1 (το αριστερό), T_2 (το δεξιό) τα οποία μπορεί να είναι κενά.

  Οι δύο ορισμοί του δυαδικού δένδρου είναι ισοδύναμοι.

Η ισοδυναμία είναι η εξής: Κάθε δυαδικό δένδρο T (πρώτος ορισμός) μπορεί να επεκταθεί κατά μοναδικό τρόπο σε ένα δυαδικό δένδρο T' (δεύτερος ορισμός), όπου κάθε κόμβος του έχει 0 ή 2 παιδιά, προσθέτοντας ένα αριστερό (αντ. δεξιό) (**κενό**) παιδί σε κάθε κόμβο με δεξιό (αντ. αριστερό) παιδί, και δύο (**κενά**) παιδιά σε κάθε φύλλο του T. Τα φύλλα του T' είναι όλα κενά δυαδικά δένδρα. Προφανώς, το δυαδικό δένδρο T μπορεί να προκύψει ξανά από το T' σβήνοντας όλα τα φύλλα του T'.

Παραδείγματα





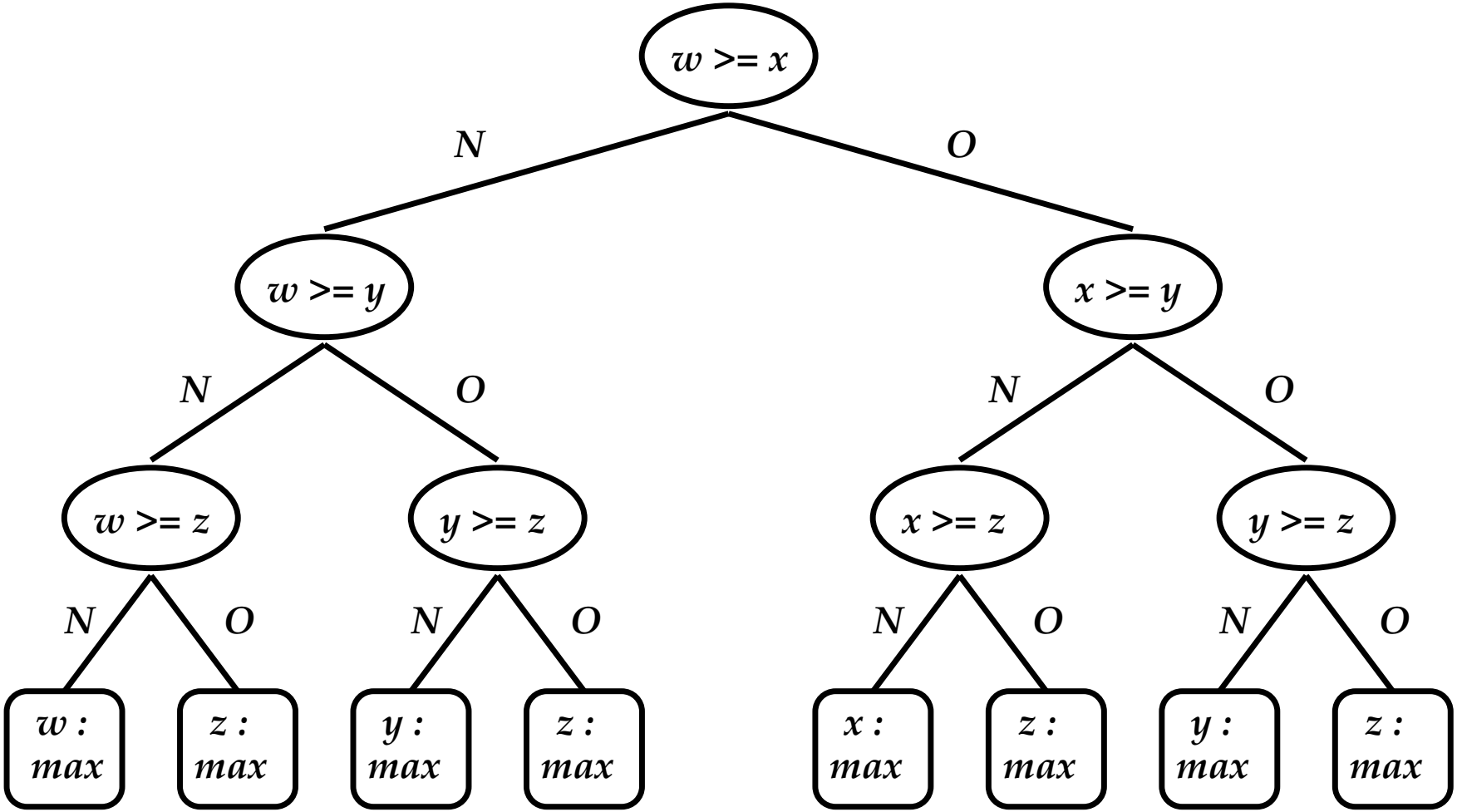
- Προφανώς, αντίστοιχα με το διατεταγμένο δάσος διατεταγμένων δένδρων ορίζεται και το **διατεταγμένο δάσος δυαδικών δένδρων**.

6η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

- **Δένδρα αποφάσεων**
- **Διασχίσεις δυαδικών δένδρων**
- **Διασχίσεις διατεταγμένων δένδρων**

Δένδρα αποφάσεων

Δένδρο απόφασης λέγεται ένα k-δένδρο του οποίου κάθε εσωτερική κορυφή παριστάνει μια ερώτηση για την οποία πρέπει να αποφασίσουμε.
Οι δυνατές απαντήσεις (αποφάσεις) παριστάνονται από τους δεσμούς που συνδέουν τον κόμβο με τους κόμβους του επόμενου επίπεδου.
Τα τελικά αποτελέσματα της διαδικασίας παριστάνονται από τα φύλλα του δένδρου.
Πολύ συχνά, οι δυνατές απαντήσεις κάθε φορά είναι : ‘‘Ναι’’ (N) ή ‘‘Όχι ’’ (O), οπότε το δένδρο απόφασης είναι ένα δυαδικό δένδρο, όπως το παρακάτω δένδρο που βρίσκει το μέγιστο μεταξύ 4 στοιχείων x, y, z, w χρησιμοποιώντας 3 συγκρίσεις.


w >= x
N
O
w >= y
x >= y
N
O
N
O
w >= z
y >= z
x >= z
y >= z
N
O
N
O
N
O
N
O
w : max
z : max
y : max
z : max
x : max
z : max
y : max
z : max

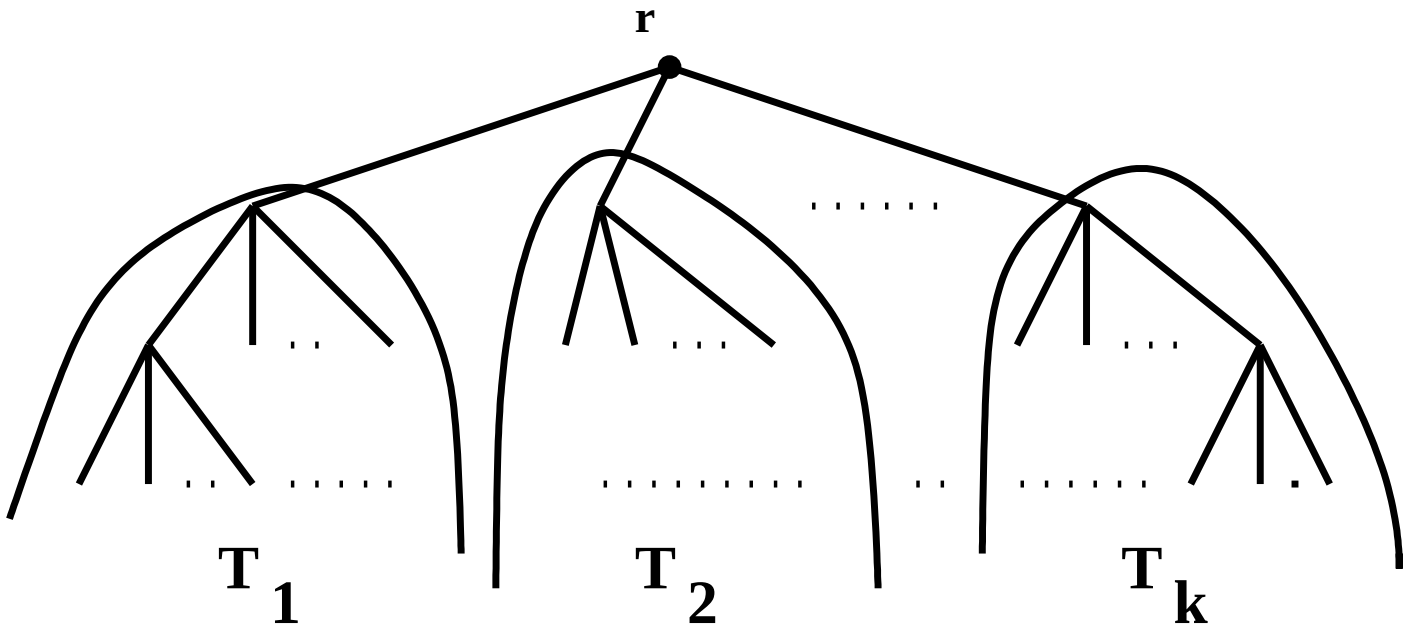
Ένα πολύ χρήσιμο θεώρημα που μας δίνει κάτω φράγματα για το πλήθος των ερωτήσεων που απαιτούνται σε κάποιο πρόβλημα απόφασης που μοντελοποιείται από ένα k-δένδρο είναι το επόμενο.

Πρόταση 19

Έστω T ένα k-δένδρο με ύψος h και l φύλλα. Τότε $h \geq \log_k l$.

Απόδειξη. Θέλουμε να δείξουμε ότι $h \geq \log_k l$, ή ισοδύναμα ότι $k^h \geq l$, δηλαδή θέλουμε να δείξουμε ότι ένα k-δένδρο ύψους h έχει το πολύ k^h φύλλα. Χρησιμοποιούμε επαγωγή ως προς h:
Για $h = 0$, το δένδρο είναι τετριμμένο και ο μοναδικός του κόμβος είναι και φύλλο, άρα έχει πράγματι k^0 φύλλα.

Έστω ότι ισχύει για κάθε $h \leq n$, και έστω T ενα k-δένδρο ύψους $n+1$. Στο γράφημα $T - r$ (όπου r η ρίζα του T) έχουμε k το πολύ υποδένδρα, τα $T_1, T_2, \ldots, T_k$.



Αλλά τα $T_1, T_2, \ldots, T_k$ έχουν ύψος το πολύ n. Άρα, από την υπόθεση της επαγωγής, έχουν συνολικά k^n το πολύ φύλλα το καθένα. Συνολικά λοιπόν, τα υποδένδρα $T_1, T_2, \ldots, T_k$ έχουν το πολύ $k \cdot k^n = k^{n+1}$ φύλλα, τα οποία είναι τα φύλλα και του αρχικού δένδρου T.

Παρατήρηση.

Από την προηγούμενη πρόταση προκύπτει ότι

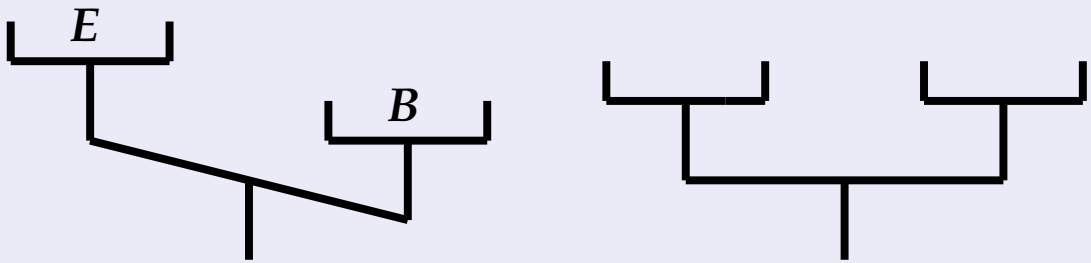
- ένα k-δένδρο με ύψος h περιέχει το πολύ k^h φύλλα.
- ένα k-δένδρο με l φύλλα έχει ύψος τουλάχιστον $\log_k l$.

Στο επόμενο παράδειγμα δίνεται μια εφαρμογή που χρησιμοποιεί ένα 3-δένδρο απόφασης:

Το πρόβλημα των κίβδηλων νομισμάτων

Ένα κανονικό νόμισμα έχει αριθμό 0. Υπάρχουν n άλλα νομίσματα ίδια ακριβώς στην εμφάνιση με το 0, που έχουν όμως αριθμούς $1, 2, \ldots, n$. Υποψιαζόμαστε ότι ένα νόμισμα μπορεί να είναι ''κίβδηλο'' (είτε λίγο ελαφρύτερο, είτε λίγο βαρύτερο).

- Να δειχθεί ότι χρειάζονται τουλάχιστον $\log_3(2n+1)$ ζυγίσματα σε μια ζυγαριά η οποία δείχνει το ελαφρύτερο και το βαρύτερο, ή δύο ίσου βάρους



για να αποφασίσουμε αν υπάρχει κίβδηλο νόμισμα, ποιο είναι αυτό και αν είναι βαρύ ή ελαφρύ.

- Να περιγραφεί μια διαδικασία που να χρησιμοποιεί ακριβώς αυτό τον αριθμό ζυγισμάτων, όταν $n = 4$.

- Υπάρχουν $2n+1$ πιθανές τελικές απαντήσεις (φύλλα) στο δένδρο απόφασης, για το παραπάνω πρόβλημα

K,	$1B$,	$1E$,	$2B$,	$2E$,	$\ldots$,	nB,	nE
καλά όλα	το 1 είναι βαρύ	το 1 είναι ελαφρύ	το 2 είναι βαρύ	το 2 είναι ελαφρύ		το n είναι βαρύ	το n είναι ελαφρύ

Το δένδρο απόφασης εδώ είναι ένα 3-δένδρο, αφού κάθε φορά που ζυγίζουμε υπάρχουν τρία πιθανά αποτελέσματα :

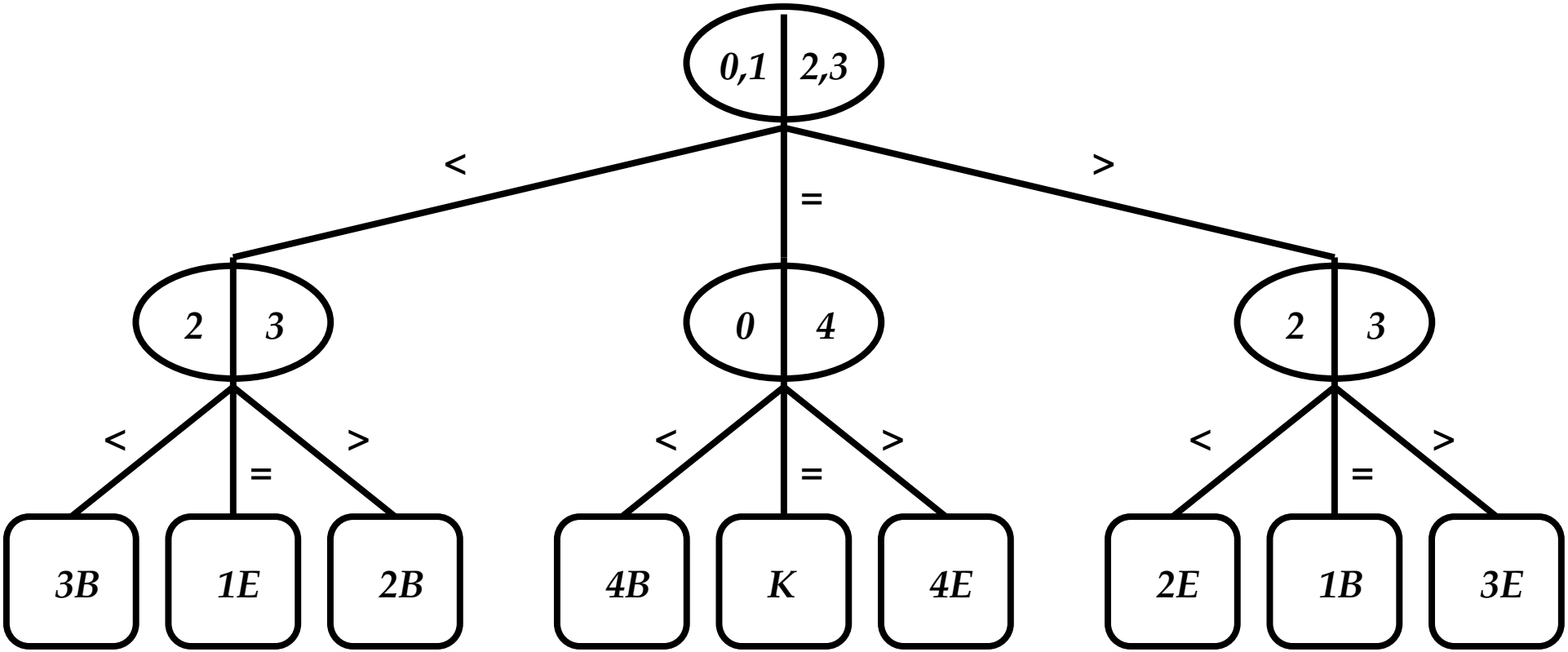
$<$: το αριστερό είναι ελαφρύτερο,

$=$: τα δύο μέρη έχουν το ίδιο βάρος,

$>$: το αριστερό είναι βαρύτερο.

Άρα το ύψος του δένδρου είναι τουλάχιστον $\log_3(2n+1)$,
οπότε το πλήθος των δεσμών του δένδρου (δηλαδή των ζυγισμάτων) από τη ρίζα μέχρι κάποιο (τουλάχιστον ένα) φύλλο (δηλαδή σε μία τουλάχιστον περίπτωση) είναι τουλάχιστον $\log_3(2n+1)$.
Για $n=4$, έχουμε $\log_3(2\cdot 4+1) = \log_3 9 = 2$, άρα το ύψος του δένδρου είναι τουλάχιστον 2, οπότε τα ζυγίσματα θα είναι τουλάχιστον 2.

- Η λύση με ακριβώς δύο ζυγίσματα φαίνεται στο παρακάτω 3-δένδρο αποφάσεων:

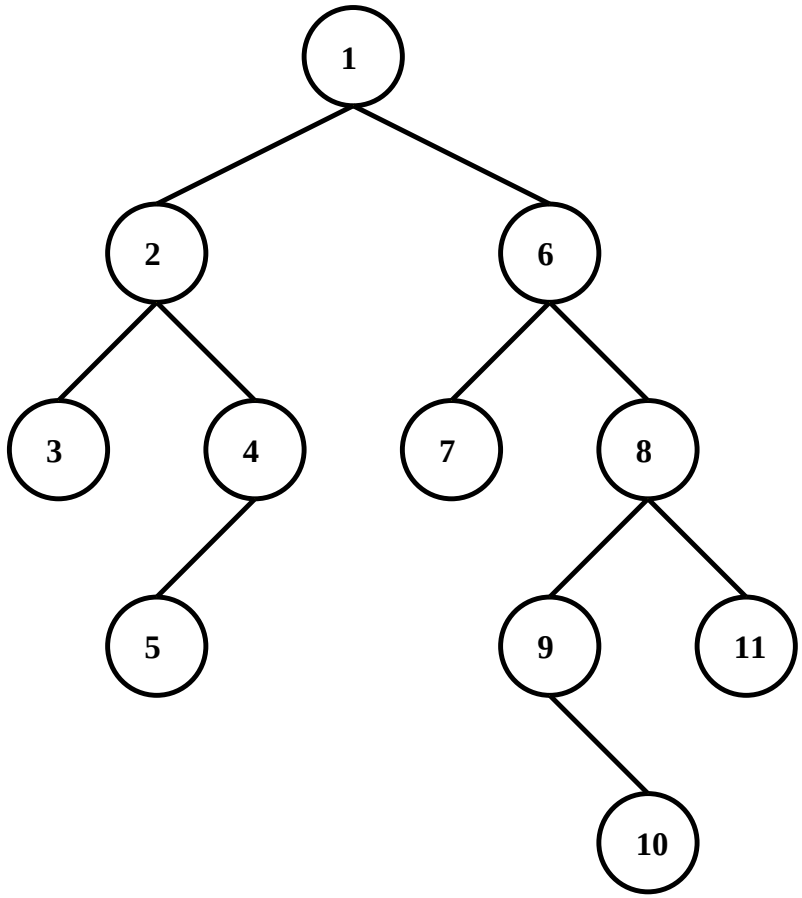


Διάσχιση δυαδικών δένδρων

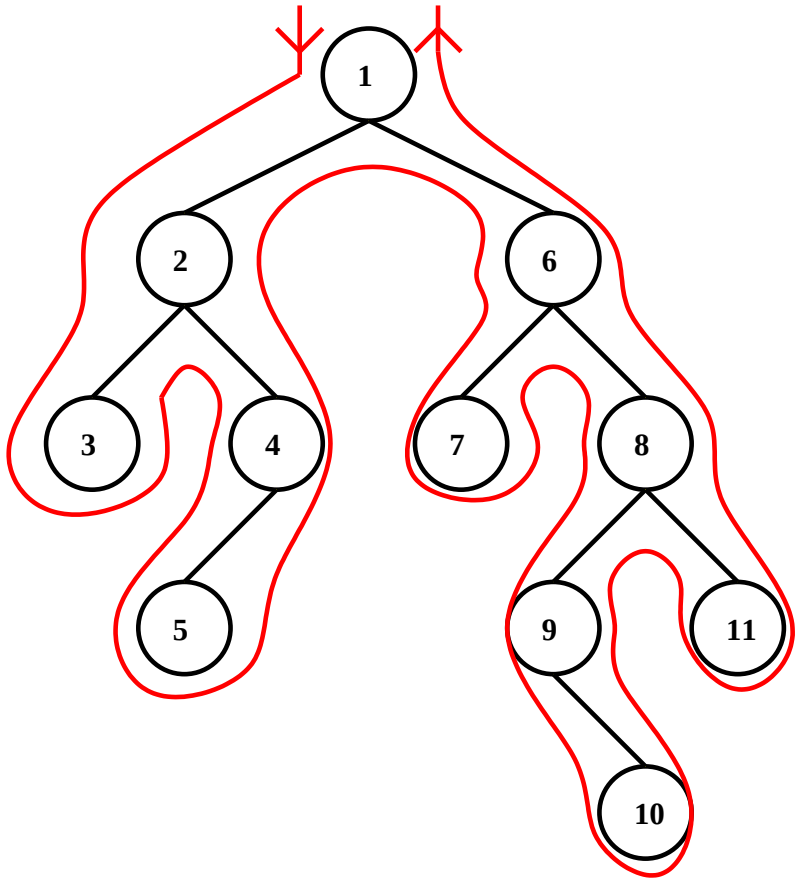
Υπάρχουν 4 βασικοί τρόποι να διασχίσουμε (αριθμήσουμε) τους κόμβους ενός δυαδικού δένδρου.
Καθένας απο τους τρόπους αυτούς καθορίζει μια ολική διάταξη των κόμβων του δένδρου.

1) **Προδιάταξη** : Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο πρίν διασχίσουμε σε προδιάταξη το αριστερό και το δεξιό υποδένδρο του. Δηλαδή, επισκεπτόμαστε πρώτα τον γονέα και μετά τα δένδρα – παιδιά του (πρώτα το αριστερό και μετά το δεξιό).

Παράδειγμα

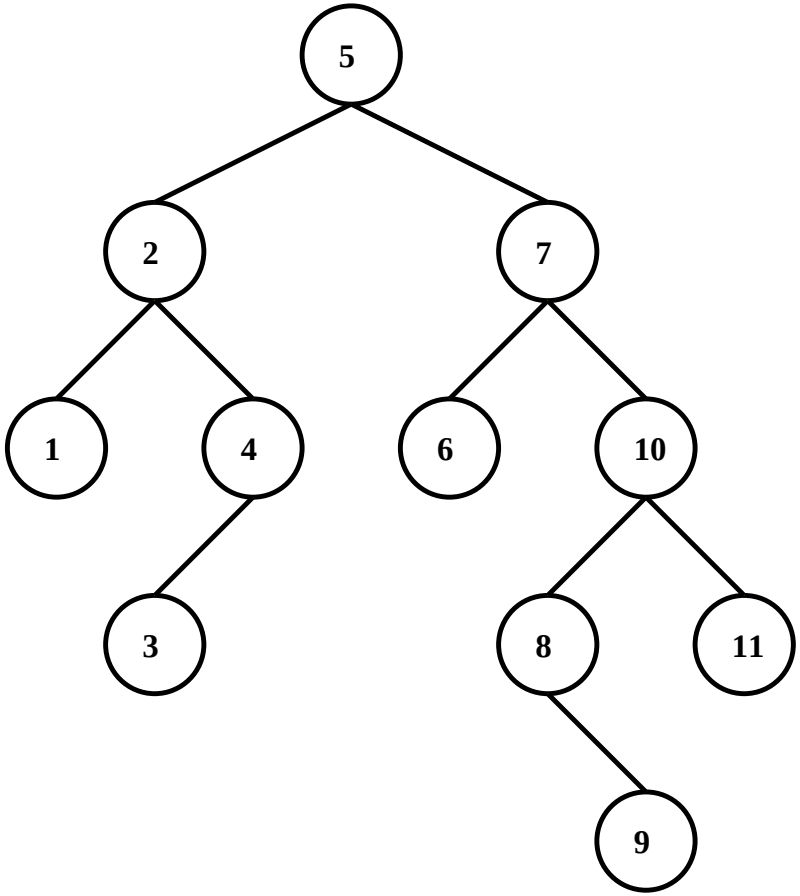


Πρακτικός τρόπος : Αριθμούμε κάθε κορυφή μόλις την πρωτοσυναντήσουμε καθώς κινούμαστε όπως δείχνει το σχήμα :

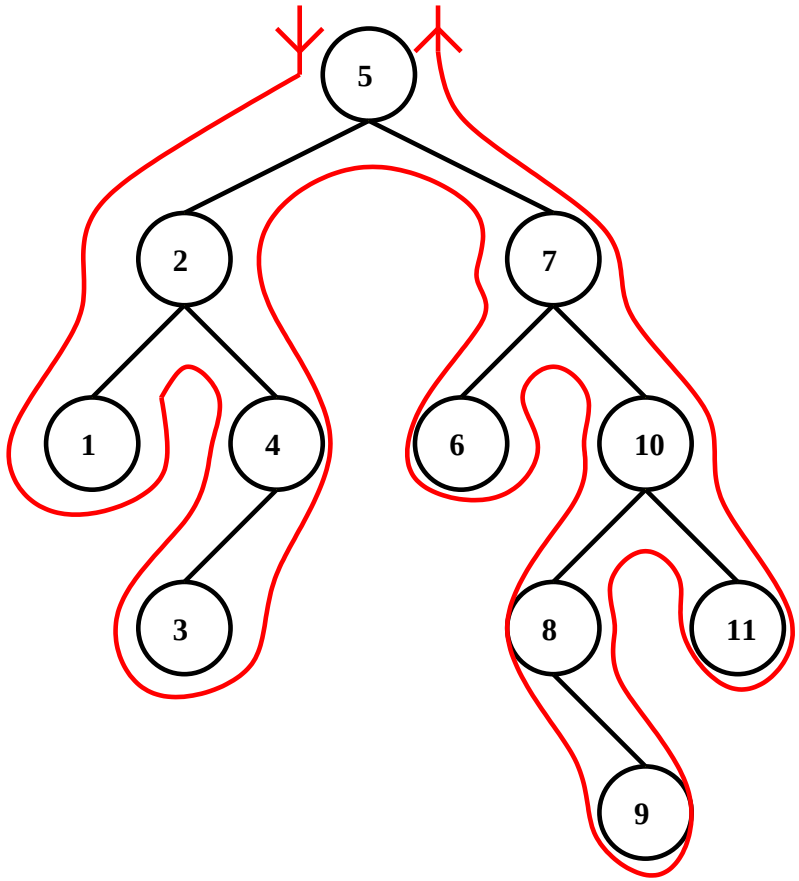


2) Ενδοδιάταξη : Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο αφού διασχίσουμε σε ενδοδιάταξη το αριστερό υπόδενδρο του και πρίν την διασχίσουμε σε ενδοδιάταξη το δεξιό υποδένδρο του. Δηλαδή, επισκεπτόμαστε πρώτα το αριστερό δένδρο – παιδί, μετά τον γονέα και μετά το δεξιό δένδρο – παιδί.

Παράδειγμα

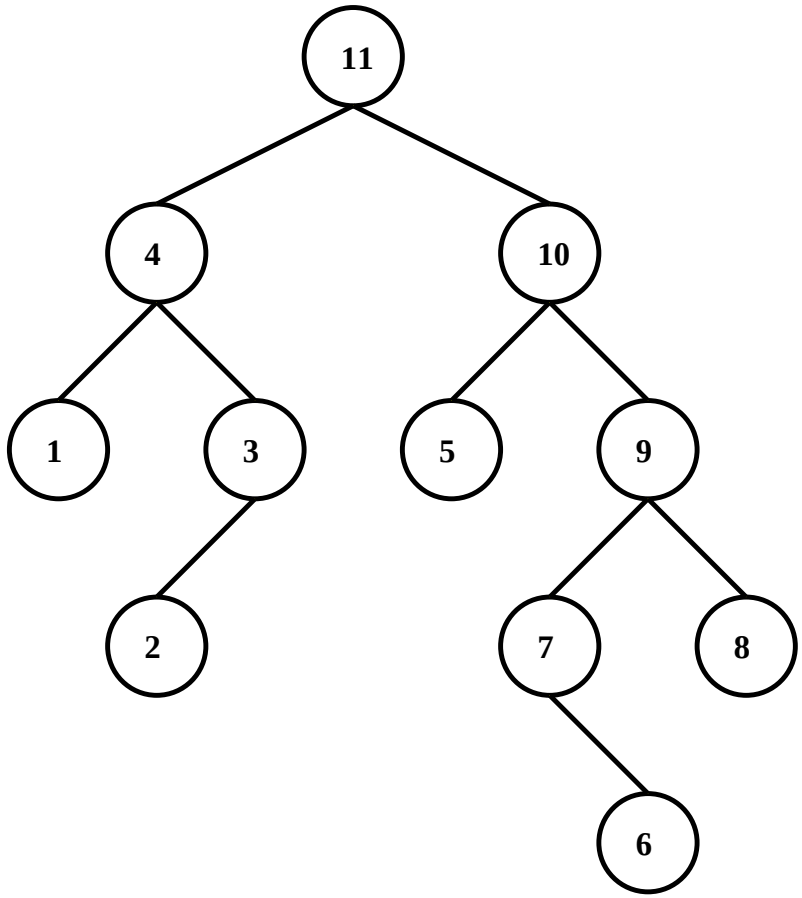


Πρακτικός τρόπος : Αριθμούμε κάθε κορυφή τη πρώτη φορά που τη συναντάμε αν δεν έχει αριστερό παιδί, ενώ την αριθμούμε τη δεύτερη φορά αν έχει αριστερό παιδί, καθώς κινούμαστε όπως δείχνει το σχήμα :

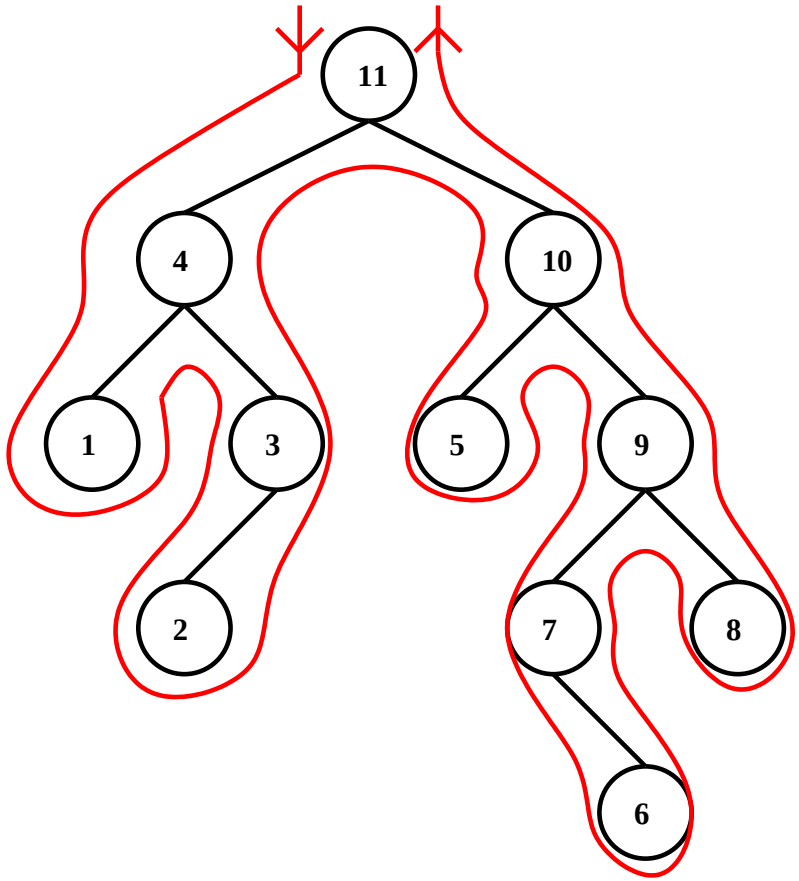


3) Μεταδιάταξη : Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο αφού έχουμε διασχίσει σε μεταδιάταξη και το αριστερό και το δεξιό υποδένδρο του. Δηλαδή, επισκεπτόμαστε πρώτα τα δένδρα – παιδιά (πρώτα το αριστερό και μετά το δεξιό) και μετά τον γονέα.

Παράδειγμα

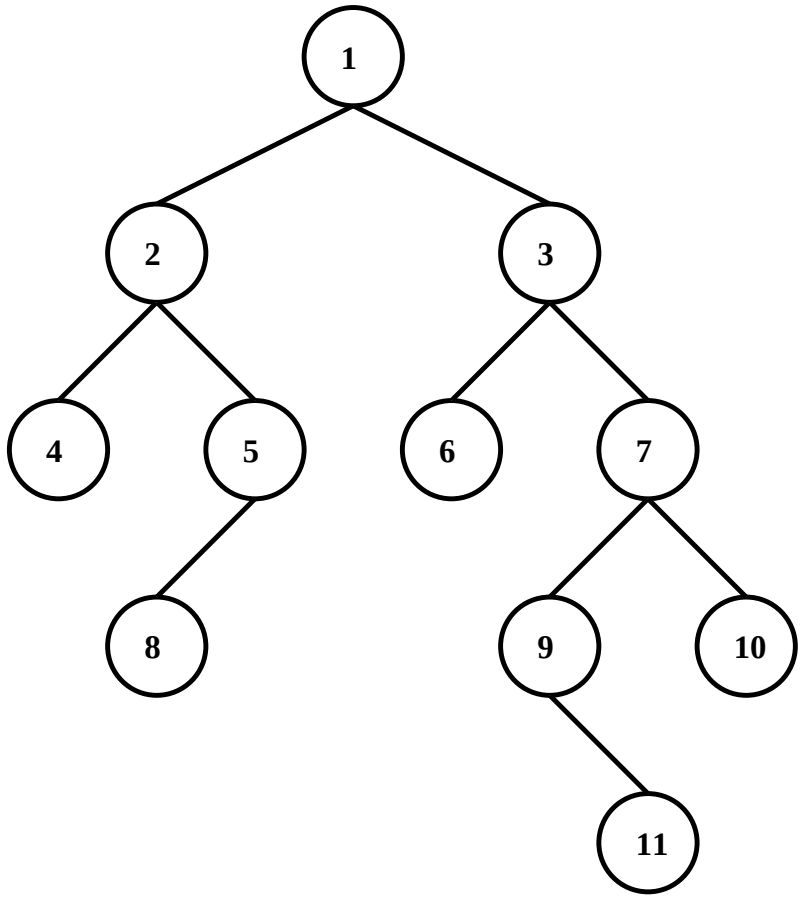


Πρακτικός τρόπος : Αριθμούμε κάθε κόμβο την τελευταία φορά που τον συναντάμε (δηλαδή καθως τον εγκαταλείπουμε για να πάμε προς τον γονέα του) καθώς κινούμαστε όπως δείχνει το σχήμα :



4) **Διάσχιση κατά σειρά επιπέδων** : Επισκεπτόμαστε (αριθμούμε) τους κόμβους κατά επίπεδο (από το μικρότερο προς το μεγαλύτερο), όπου σε κάθε επίπεδο επισκεπτόμαστε τους κόμβους από τα αριστερά προς τα δεξιά.

Παράδειγμα

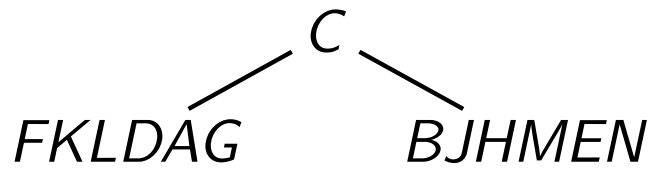


Παρατήρηση. Αν γνωρίζουμε τις διασχίσεις ενός δυαδικού δένδρου σε προδιάταξη και ενδοδιάταξη μπορούμε να ανακτήσουμε το δυαδικό δένδρο.

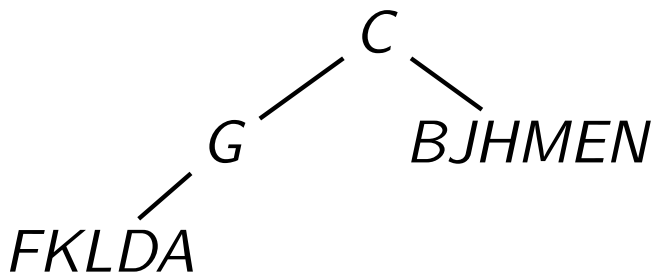
Παράδειγμα Να βρεθεί το δυαδικό δένδρο T με ετικέτες A, B, C, D, E, F, G, H, J, K, L, M, N του οποίου οι διασχίσεις σε προδιάταξη και ενδοδιάταξη είναι αντίστοιχα

	1	2	3	4	5	6	7	8	9	10	11	12	13
προδιάταξη:	C	G	D	K	F	L	A	J	B	H	E	M	N
ενδοδιάταξη:	F	K	L	D	A	G	C	B	J	H	M	E	N

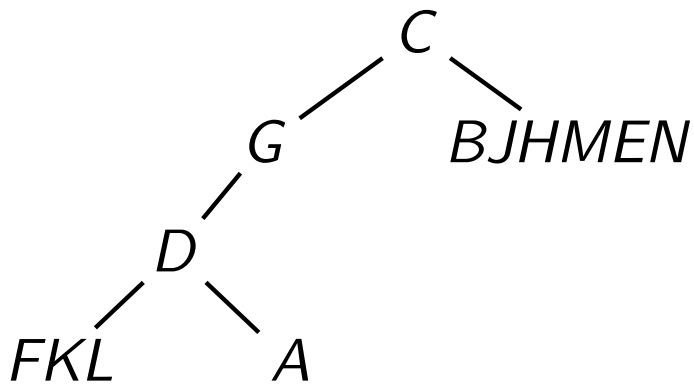
Επειδή το 1ο στοιχείο σε προδιάταξη είναι το C, έπεται ότι το C είναι η ρίζα του δένδρου. Στην ενδοδιάταξη το C βρίσκεται στην 7η θέση, άρα οι κορυφές με ετικέτες F, K, L, D, A, G περιέχονται στο αριστερό υποδένδρο του C, ενώ οι υπόλοιπες κορυφές (με ετικέτες B, J, H, M, E, N) περιέχονται στο δεξιό υποδένδρο του C.



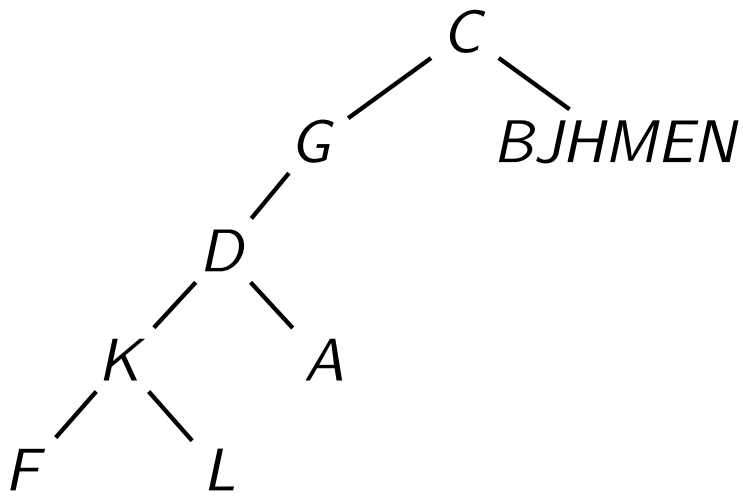
Επειδή το 2ο στοιχείο σε προδιάταξη είναι το G έπεται ότι το G είναι ρίζα του αριστερού υποδενδρου του C. Από την ενδοδιάταξη συμπεραίνουμε το αριστερό υποδένδρο του G αποτελείται από τις κορυφές με ετικέτες F, K, L, D, A, ενώ το δεξιό υποδένδρο του είναι κενό.



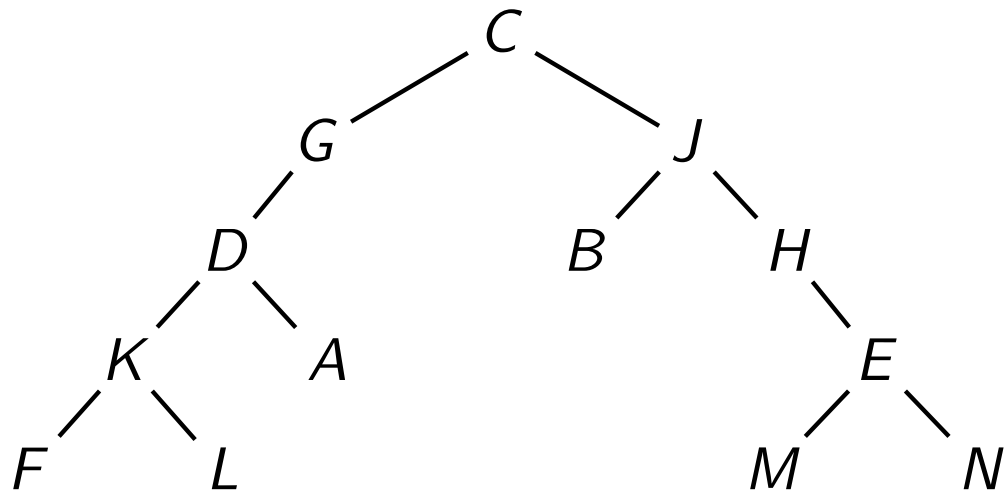
Επειδή το 3ο στοιχείο σε προδιάταξη είναι το D έπεται ότι το D είναι ρίζα του αριστερού υποδενδρου του G. Από την ενδοδιάταξη συμπεραίνουμε το αριστερό υποδένδρο του D αποτελείται από τις κορυφές με ετικέτες F, K, L, ενώ το δεξιό υποδένδρο του αποτελείται από την κορυφή A.



Επειδή το 4ο στοιχείο σε προδιάταξη είναι το K έπεται ότι το K είναι ρίζα του αριστερού υποδενδρου του D. Από την ενδοδιάταξη συμπεραίνουμε το αριστερό υποδένδρο του K αποτελείται από τις κορυφή F, ενώ το δεξιό υποδένδρο του αποτελείται από την κορυφή L.



Συνεχίζοντας με τον ίδιο τρόπο προκύπτει τελικά το δυαδικό δένδρο



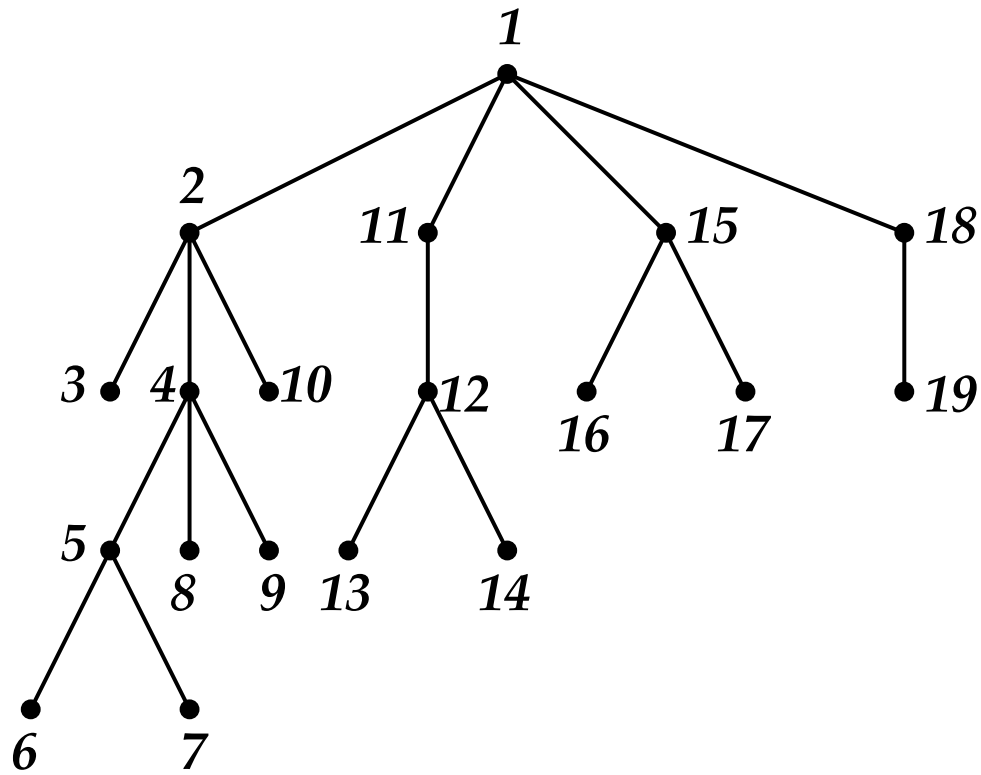
	1	2	3	4	5	6	7	8	9	10	11	12	13
προδιάταξη:	*C*	*G*	*D*	*K*	*F*	*L*	*A*	*J*	*B*	*H*	*E*	*M*	*N*
ενδοδιάταξη:	*F*	*K*	*L*	*D*	*A*	*G*	*C*	*B*	*J*	*H*	*M*	*E*	*N*

Διάσχιση διατεταγμένων δένδρων

1) Προδιάταξη

Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο πριν διασχίσουμε (σύμφωνα με τη διάταξή τους) τα δένδρα-παιδιά του σε προδιάταξη. Δηλαδή πρώτα τον γονέα και έπειτα τα δένδρα παιδιά του (από το πρώτο προς το τελευταίο).

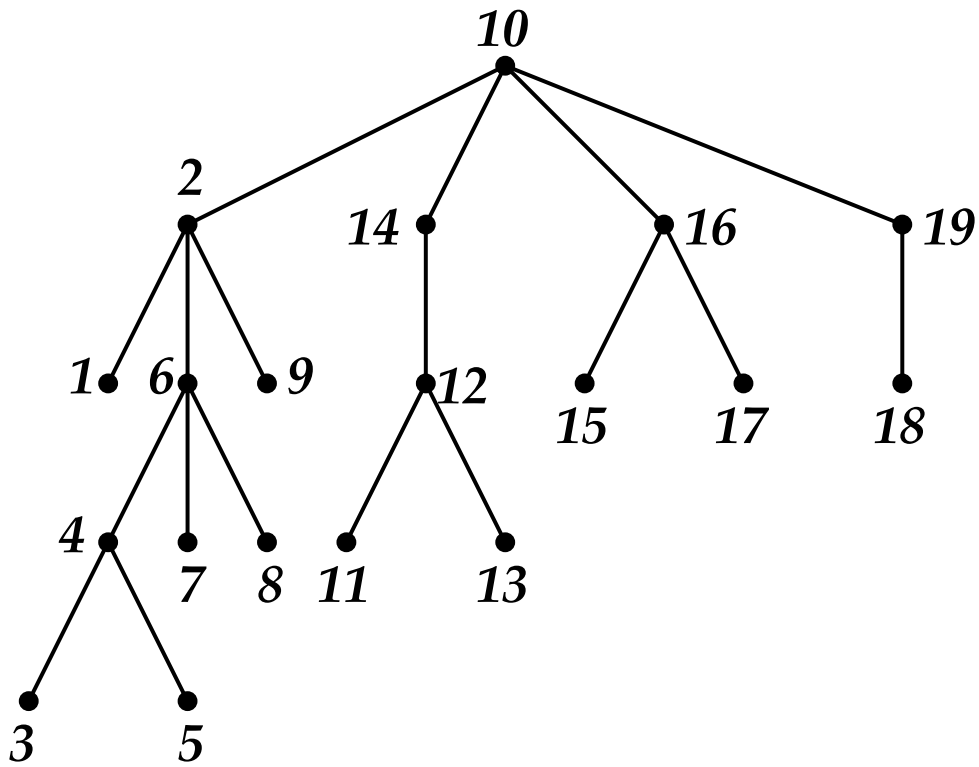
Παράδειγμα



2) Ενδοδιάταξη

Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο αφού διασχίσουμε σε ενδοδιάταξη το πρώτο δένδρο-παιδί και πριν διασχίσουμε (σύμφωνα με την διάταξή τους) τα υπόλοιπα δένδρα-παιδιά του σε ενδοδιάταξη. Δηλαδή πρώτα το πρώτο δένδρο-παιδί, μετά τον γονέα κι έπειτα τα υπόλοιπα δένδρα-παιδιά του (από το δεύτερο προς το τελευταίο).

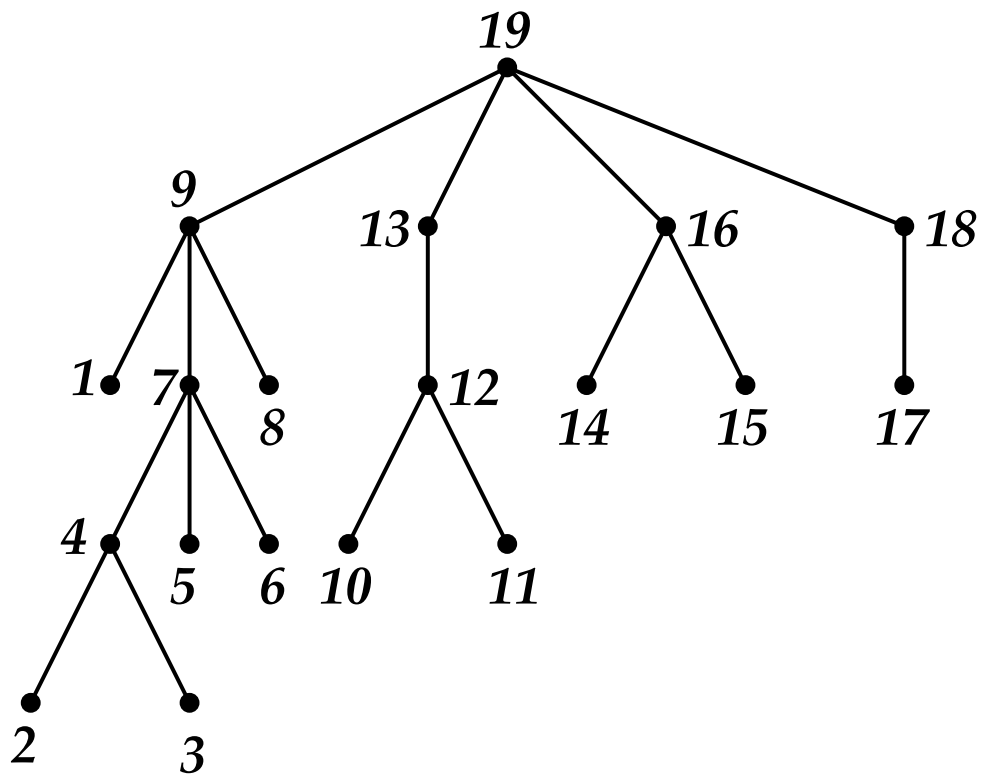
Παράδειγμα



3) Μεταδιάταξη

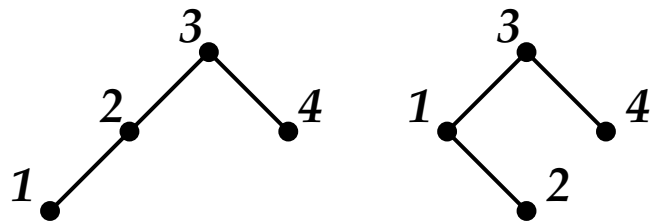
Επισκεπτόμαστε (αριθμούμε) κάθε κόμβο αφού διασχίσουμε (σύμφωνα με τη διάταξή τους) τα δένδρα-παιδιά του σε μεταδιάταξη. Δηλαδή πρώτα τα δένδρα-παιδιά (από το πρώτο προς το τελευταίο) και έπειτα τον γονέα.

Παράδειγμα

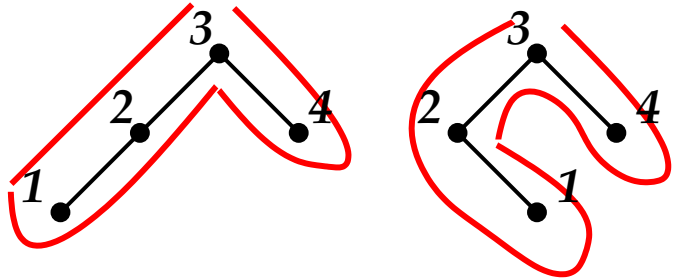


Παρατήρηση. Μπορούμε να αριθμήσουμε τα διατεταγμένα δένδρα σε προδιάταξη και μεταδιάταξη με πρακτικό τρόπο, αντίστοιχα με τα δυαδικά δένδρα. Ο πρακτικός τρόπος για την ενδοδιάταξη των διατεταγμένων δένδρων όμως είναι διαφορετικός: Αριθμούμε τον κάθε κόμβο τη δεύτερη φορά που τον συναντάμε, εκτός αν είναι φύλλο, οπότε τον αριθμούμε την πρώτη φορά. Η διαφορά αυτή οφείλεται στο ότι στα δυαδικά δένδρα, στην περίπτωση γονέα με μοναδικό παιδί η σειρά αρίθμησης τους δεν είναι μονοσήμαντα καθορισμένη. Αν το παιδί είναι αριστερό παιδί τότε προηγείται του γονέα ενώ αν είναι δεξιό τότε έπεται του γονέα.

Έτσι για παράδειγμα, ενώ η ενδοδιάταξη και στα δύο παρακάτω δυαδικά δένδρα δίνει



ο πρακτικός τρόπος θα έδινε αντίστοιχα

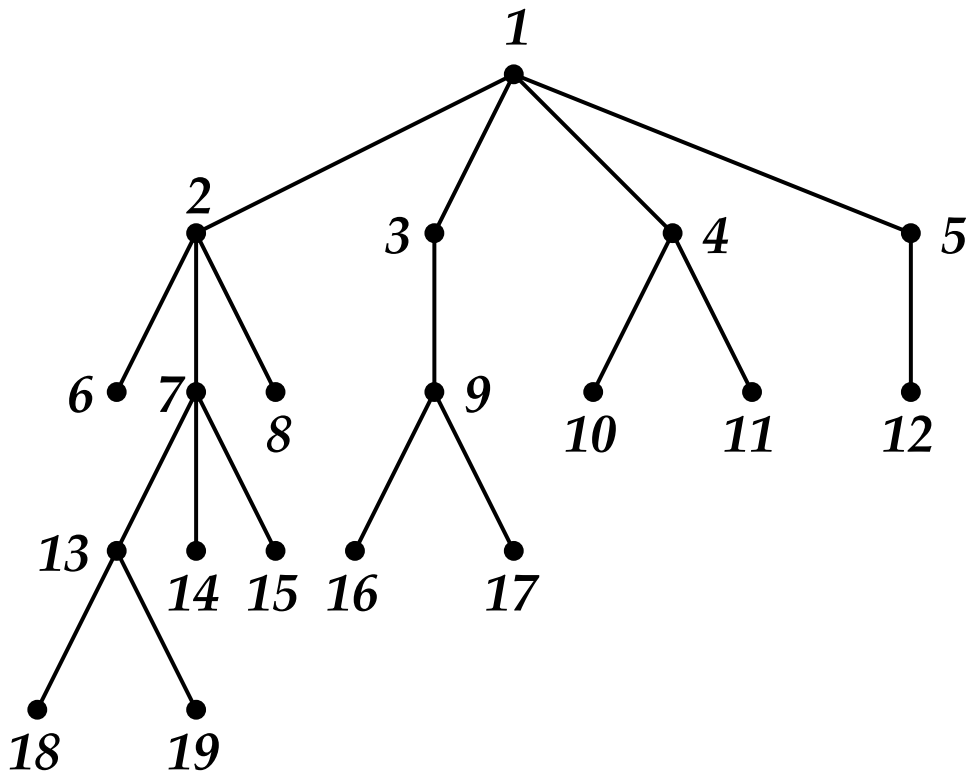


με το δεύτερο δένδρο να δίνει διαφορετική ενδοδιάταξη από ό,τι ο ορισμός.

4. Διάταξη κατά επίπεδα

Επισκεπτόμαστε (αριθμούμε) τους κόμβους κατά επίπεδο (από το μικρότερο επίπεδο στο μεγαλύτερο), όπου σε κάθε επίπεδο επισκεπτόμαστε τους κόμβους από τα αριστερά προς τα δεξιά.

Παράδειγμα



Παρατήρηση. Οι διασχίσεις των δένδρων (δυαδικών, ή διατεταγμένων) σύμφωνα με οποιαδήποτε από τις παραπάνω διατάξεις γενικεύονται προφανώς στα διατεταγμένα δάση, διασχίζοντας σύμφωνα με τη συγκριμένη κάθε φορά διάταξη το πρώτο δένδρο του διατεταγμένου δάσους, ακολούθως το δεύτερο δένδρο, κ.ο.κ.

7η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

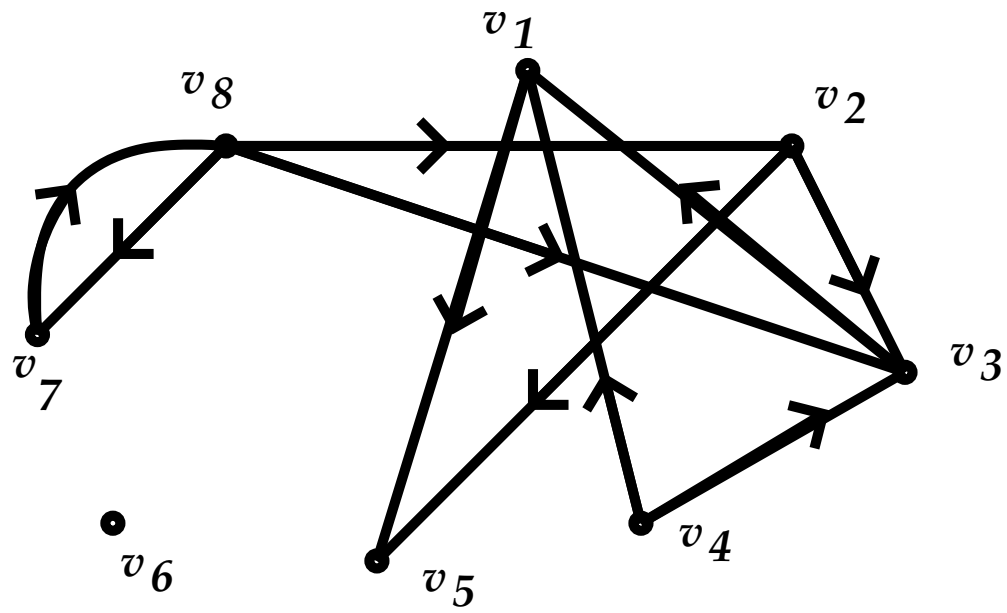
- **Γραφήματα τόξων**
- **Κυκλώματα**
- **Μήτρα γραφήματος τόξων**
- **Τοπολογική διάταξη**
- **Συνάρτηση Grundy - Sprague**

Γραφήματα τόξων

Κάθε δυάδα $G = (V(G), U(G))$, ή (V, U), όπου V είναι ένα μη κενό σύνολο και U είναι ένα σύνολο από διατεταγμένα ζεύγη $(v, u) \in V^2$ ονομάζεται **γράφημα τόξων**, ή **προσανατολισμένο γράφημα**, ή **γράφημα με κατεύθυνση**, ή **διγράφημα**.
Τα στοιχεία του V καλούνται **κορυφές**, ή **σημεία**, ή **κόμβοι** όπως και στα γραφήματα δεσμών, ενώ τα στοιχεία του U καλούνται **τόξα** και συμβολίζονται γραφικά με τόξα. Αν $(v, u) \in U$, τότε λέμε το v (αντ. u) είναι **αρχή** (αντ. **τέλος**) του (v, u).

Παράδειγμα

Η δυάδα $G = (V, U)$ όπου $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$ και $U = \{(v_1, v_5), (v_2, v_3), (v_2, v_5), (v_3, v_1), (v_4, v_1), (v_4, v_3), (v_7, v_8), (v_8, v_2), (v_8, v_3), (v_8, v_7)\}$ είναι ένα γράφημα τόξων. Η γραφική του απεικόνιση είναι η ακόλουθη:



Το τόξο (v, v), $v \in V$ ονομάζεται **βρόχος**.

Οι ορισμοί είναι αντίστοιχοι με αυτούς που δώσαμε στα γραφήματα δεσμών με τις εξής επισημάνσεις :
Τώρα ορίζεται ο **βαθμός εξόδου** $d_+(v)$ και ο **βαθμός εισόδου** $d_-(v)$ ενός κόμβου $v \in V(G)$, ως

$$d_+(v) = |\{u \in V(G) : (v, u) \in U(G)\}|,$$

(δηλαδή, το πλήθος των τόξων του G, των οποίων ο v είναι αρχή), και

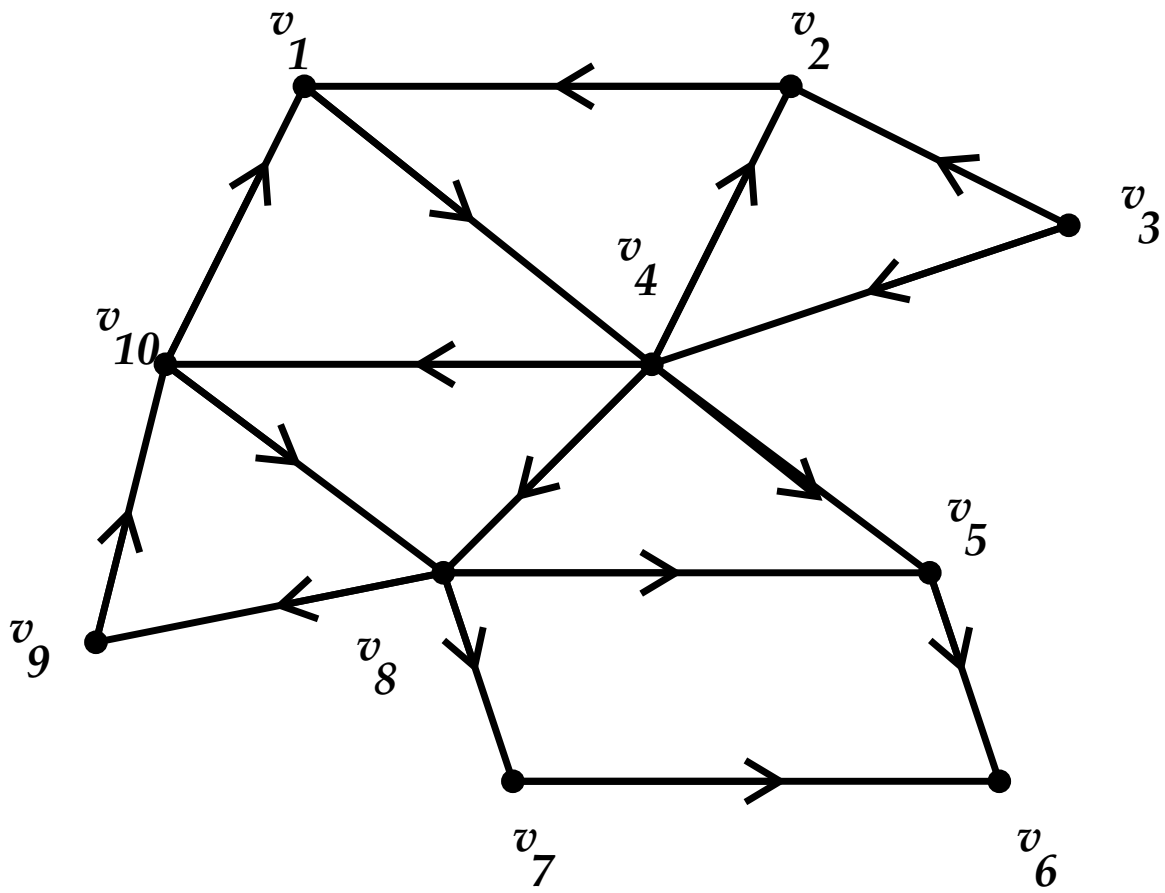
$$d_-(v) = |\{u \in V(G) : (u, v) \in U(G)\}|$$

(δηλαδή, το πλήθος των τόξων του G, των οποίων ο v είναι τέλος.)
Προφανώς τώρα ο **βαθμός** $d(v)$ ενός κόμβου v ορίζεται από την σχέση

$$d(v) = d_+(v) + d_-(v).$$

Παράδειγμα

Στο επόμενο γράφημα

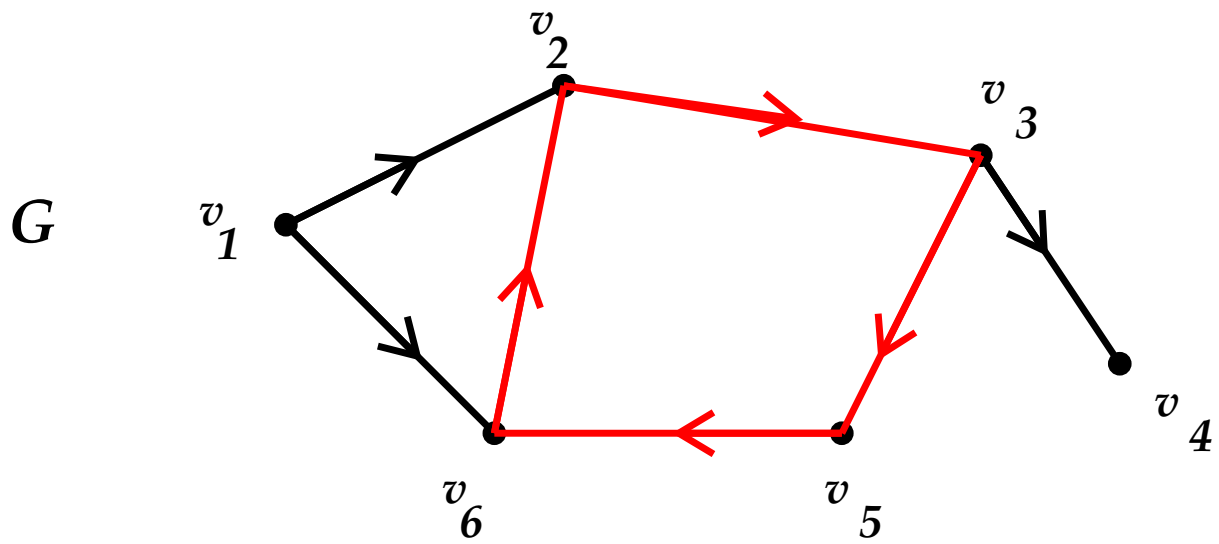


είναι $d_+(v_8) = 3$, $d_-(v_8) = 2$, $d(v_8) = 5$, $d_-(v_2) = 2$, $d_-(v_3) = 0$, κ.λπ.

Η διαδρομή σε ένα διγράφημα πρέπει εν γένει να ‘‘ακολουθεί’’ τη διεύθυνση κάθε τόξου από την αρχή προς το τέλος του. Αν αυτό δεν συμβαίνει, έχουμε μια **ημιδιαδρομή**.

Συνήθως η κλειστή διαδρομή που σχηματίζεται από τόξα λέγεται **κύκλωμα**.

Παράδειγμα

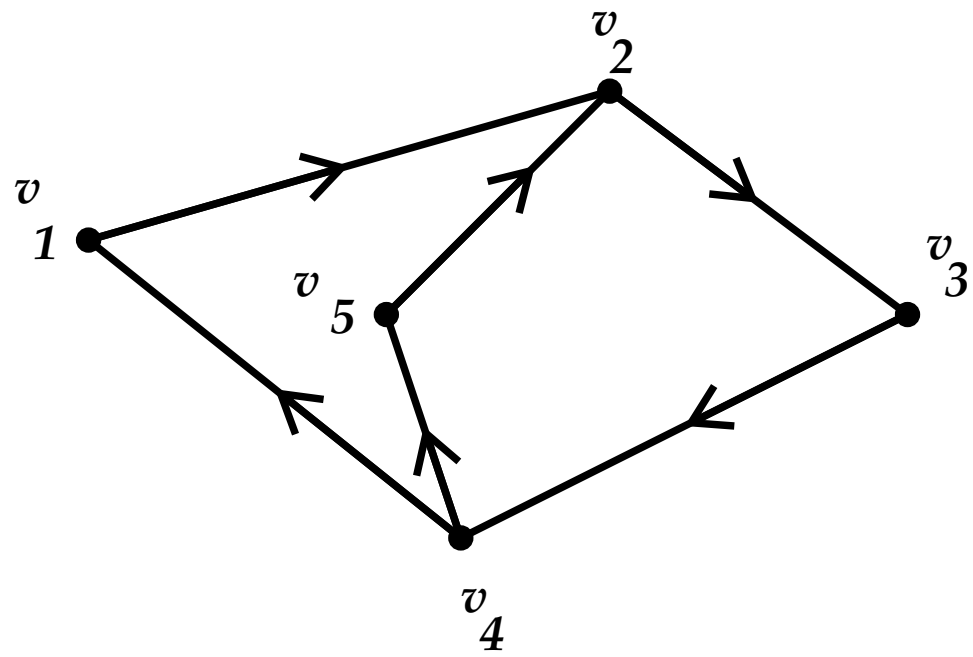


Στο γράφημα G η διαδρομή $(v_2, v_3, v_5, v_6, v_2)$ είναι κύκλωμα, ενώ η ημιδιαδρομή (v_1, v_2, v_6, v_1) δεν είναι.

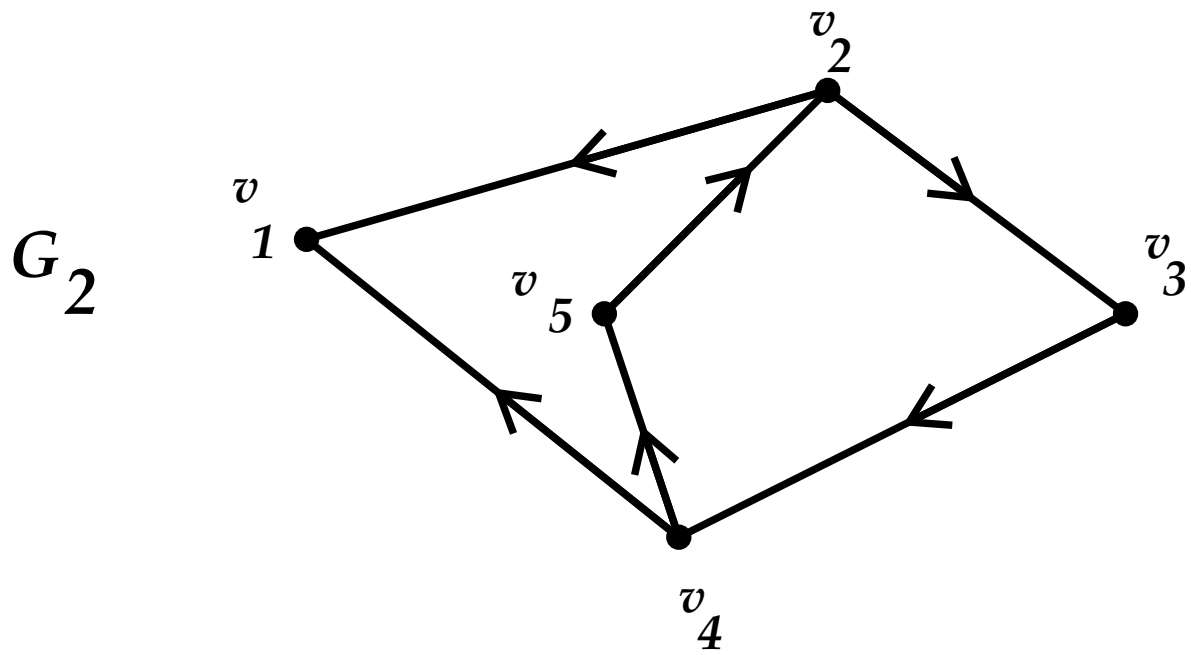
Υπάρχουν διάφορα είδη συνεκτικότητας στα διγραφήματα :
Ένα γράφημα τόξων λέγεται **ισχυρά συνεκτικό** αν για οποιοδήποτε ζεύγος κόμβων του υπάρχει μονοπάτι και από τον πρώτο προς τον δεύτερο, και από τον δεύτερο προς τον πρώτο.
Ένα γράφημα τόξων λέγεται **μονομερώς συνεκτικό** αν δεν είναι ισχυρά συνεκτικό αλλά για οποιοδήποτε ζεύγος κόμβων του υπάρχει μονοπάτι είτε από τον πρώτο προς τον δεύτερο, είτε από τον δεύτερο προς τον πρώτο.
Ένα γράφημα τόξων λέγεται **ασθενώς συνεκτικό** αν δεν είναι μονομερώς συνεκτικό, αλλά για οποιοδήποτε ζεύγος κόμβων του υπάρχει ημιδιαδρομή μεταξύ τους.

Παραδείγματα

G_1

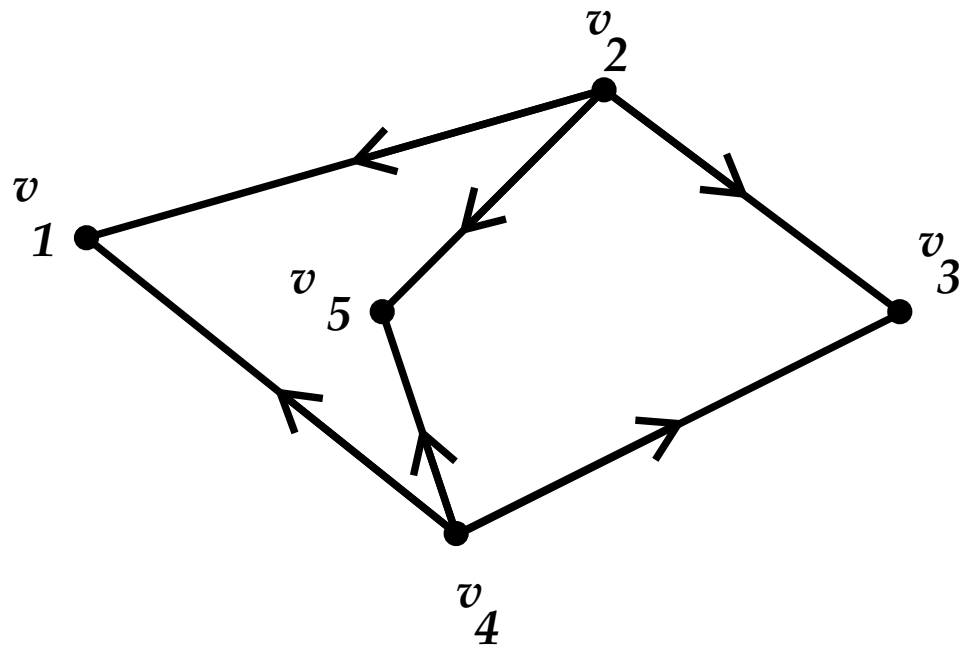


Το γράφημα G_1 είναι ισχυρά συνεκτικό.



Το γράφημα G_2 είναι μονομερώς συνεκτικό (αφού για παράδειγμα, ενώ υπάρχει $v_3 - v_1$ μονοπάτι, δεν υπάρχει $v_1 - v_3$ μονοπάτι).

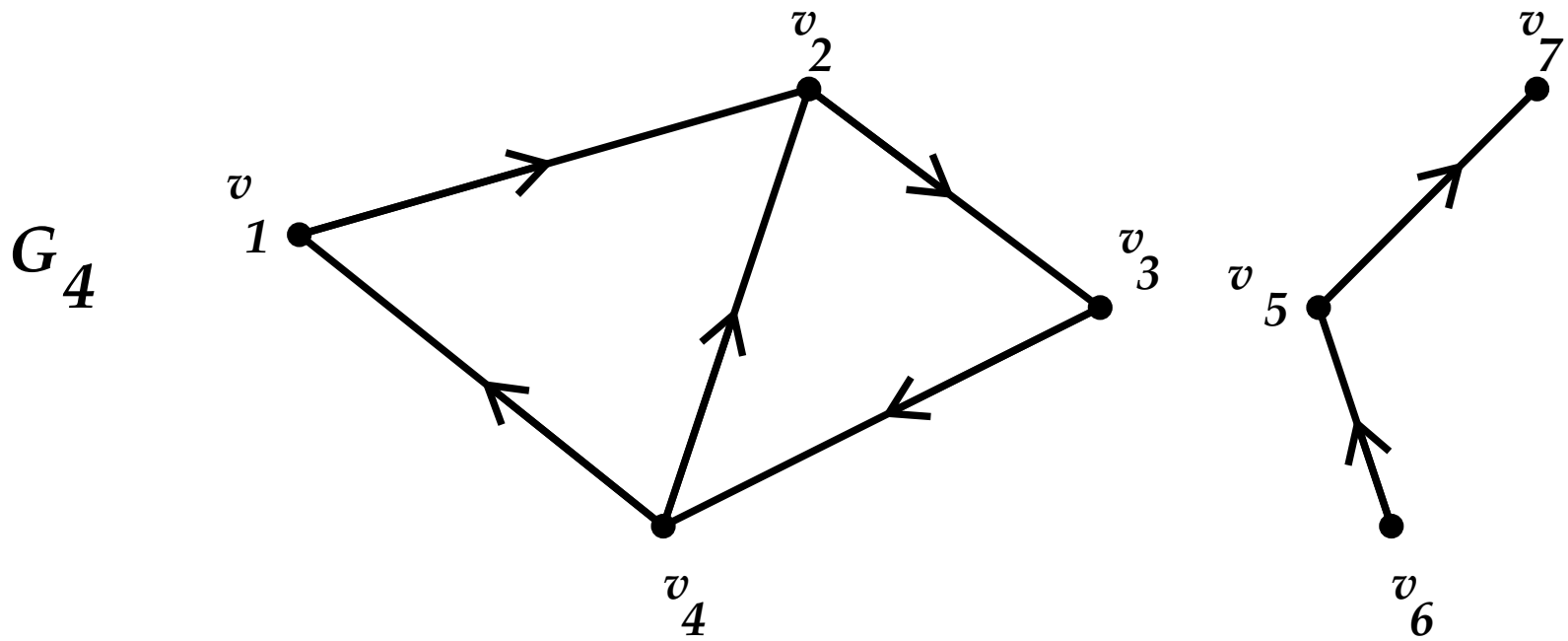
G_3



Το γράφημα G_3 είναι ασθενώς συνεκτικό (αφού για παράδειγμα δεν υπάρχει ούτε $v_2 - v_4$, ούτε $v_4 - v_2$ μονοπάτι, ενώ υπάρχει η ημιδιαδρομή (v_2, v_3, v_4)).

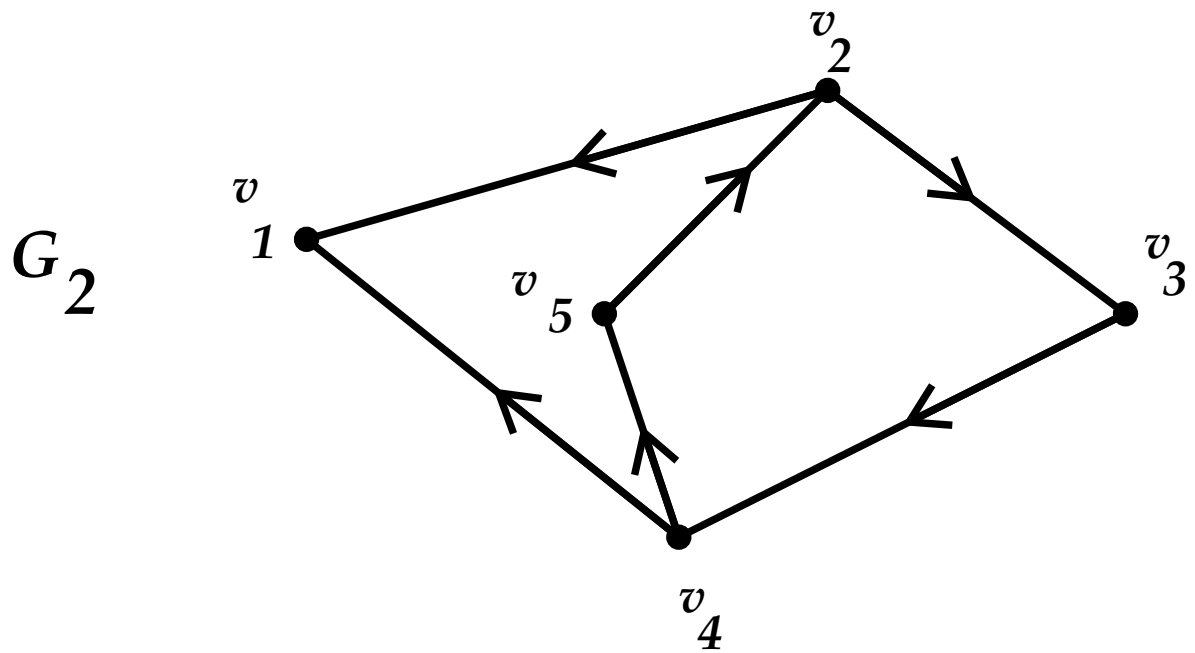
Ένα γράφημα τόξων ονομάζεται **μη συνεκτικό** αν δεν είναι ούτε ισχυρά, ούτε μονομερώς, ούτε ασθενώς συνεκτικό.

Παράδειγμα



Το γράφημα G_4 είναι μη συνεκτικό.

Ισχυρά συνεκτική συνιστώσα ενός γραφήματος τόξων G ονομάζεται οποιοδήποτε μεγιστικό ισχυρά συνεκτικό υπογράφημα του G.
Για παράδειγμα, το γράφημα G_2

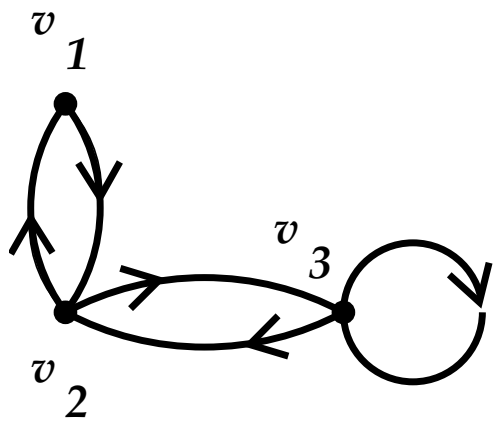


(το οποίο όπως είδαμε δεν είναι ισχυρά συνεκτικό) περιέχει μια ισχυρά συνεκτική συνιστώσα: το κύκλωμα $(v_2, v_3, v_4, v_5, v_2)$.

Στα γραφήματα τόξων ορίζουμε και τα παρακάτω είδη γραφημάτων:
Συμμετρικό ονομάζεται ένα γράφημα τόξων $G = (V, U)$ για το οποίο ισχύει

$$(u, v) \in U \Leftrightarrow (v, u) \in U.$$

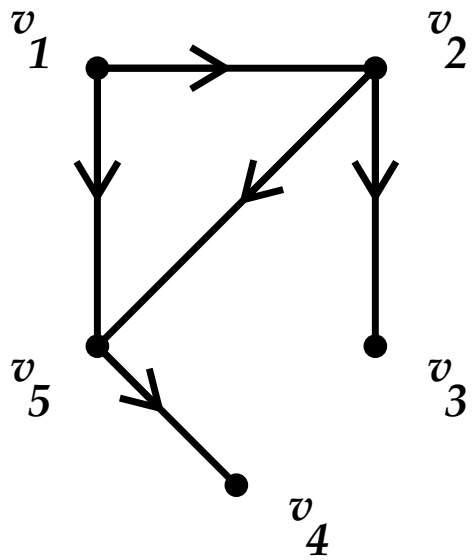
Παράδειγμα



Αντισυμμετρικό ονομάζεται ένα γράφημα τόξων $G = (V, U)$ για το οποίο ισχύει

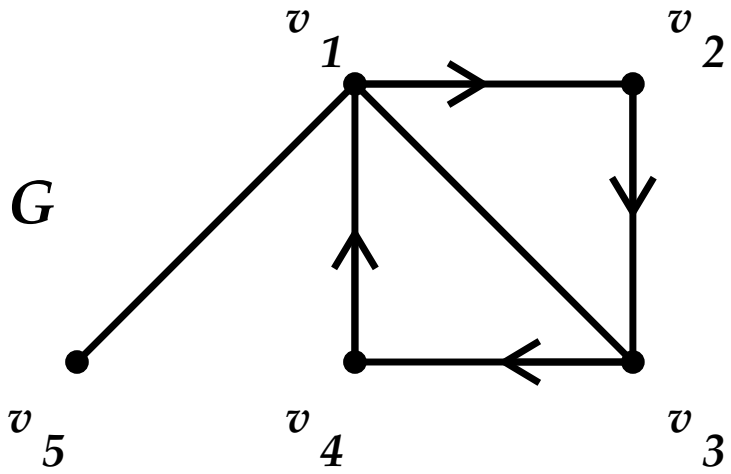
$$(u, v) \in U \Rightarrow (v, u) \notin U$$

Παράδειγμα

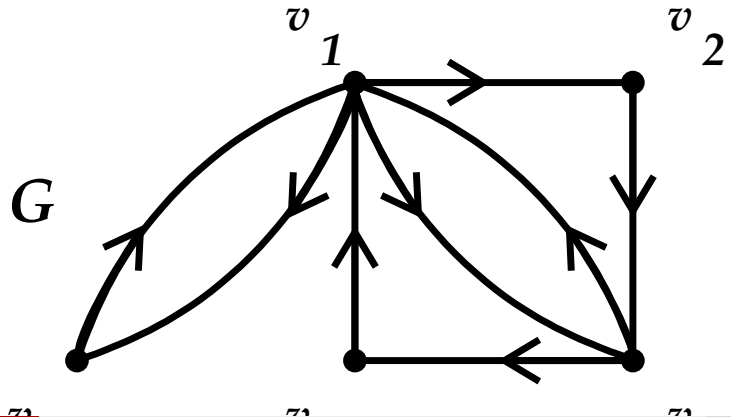


Παρατήρηση. Μερικές φορές εμφανίζονται γραφήματα που περιέχουν συγχρόνως και δεσμούς και τόξα.

Παράδειγμα



Τα γραφήματα αυτά, τα θεωρούμε ουσιαστικά ως γραφήματα τόξων, αντικαθιστώντας κάθε δέσμο $\{v, u\}$ με δύο τόξα (v, u) και (u, v). Έτσι, το προηγούμενο παράδειγμα γράφεται:



Παρατήρηση. Φυσικά, με την ίδια λογική μπορούμε γενικά οποιοδήποτε γράφημα δεσμών να το θεωρήσουμε αντίστοιχα ως γράφημα τόξων, το οποίο θα είναι προφανώς συμμετρικό. Το μειονέκτημα μιας τέτοιας προσέγγισης είναι ότι η αντίστοιχη θεωρία και οι εφαρμογές γίνονται γενικά πολύ πιο πολύπλοκες.

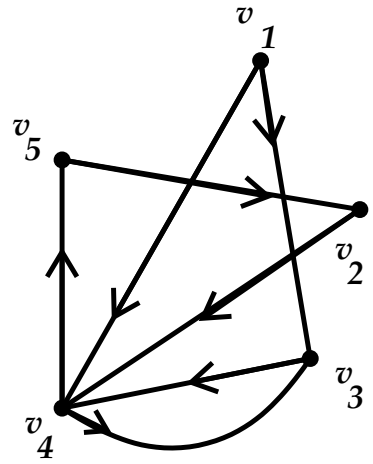
Μήτρα γραφήματος τόξων

Έστω $G = (V, U)$ ένα γράφημα τόξων. Ορίζουμε την $|V| \times |V|$ μήτρα M_G ή M του G ως εξής :

$$M = [m_{ij}] \text{ με } m_{ij} = \begin{cases} 1, & \text{αν } (x_i, x_j) \in U \\ 0, & \text{αν } (x_i, x_j) \notin U. \end{cases}$$

Η μήτρα αυτή ονομάζεται **μήτρα (γειτονικότητας) του γραφήματος τόξων**.

Παράδειγμα Στο γράφημα G



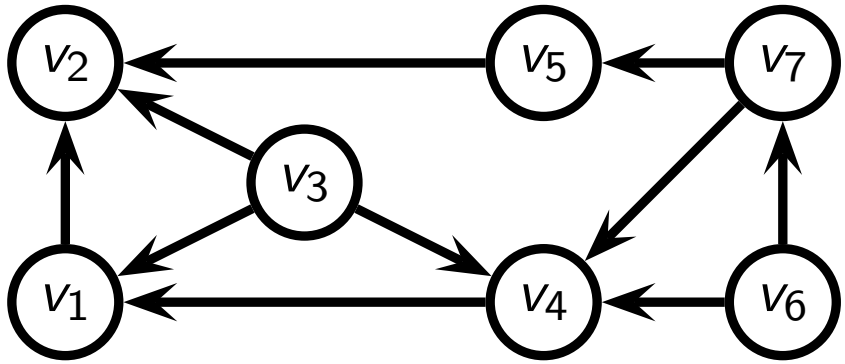
αντιστοιχεί η μήτρα

$$M = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Τοπολογική διάταξη

Έστω $G = (V, U)$ ένα γράφημα τόξων. Μια ολική διάταξη $<$ στο σύνολο των κορυφών του G ονομάζεται **τοπολογική** (topological sorting) αν και μόνο αν για κάθε τόξο $(v, u) \in U$ ισχύει ότι $v < u$ (v προηγείται του u) στην διάταξη.

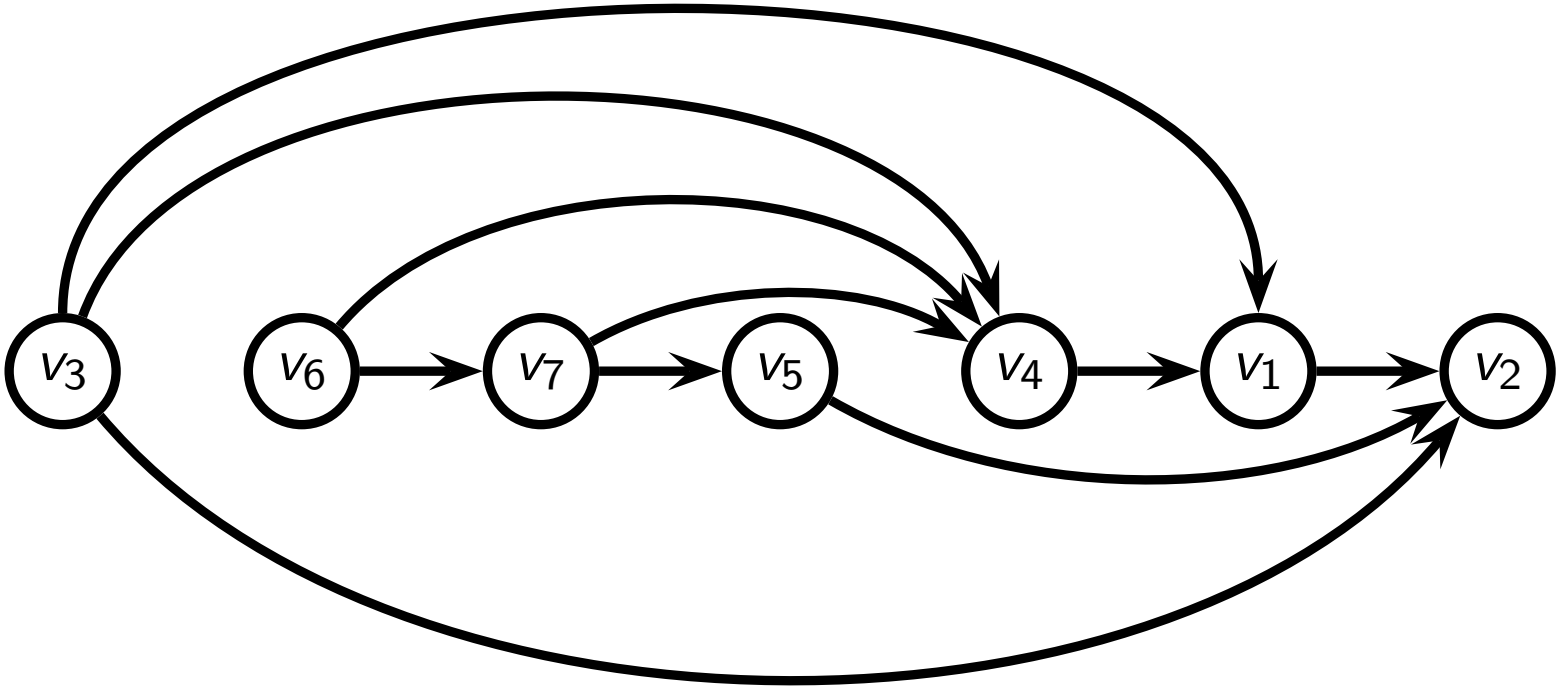
Παράδειγμα Στο επόμενο γράφημα τόξων



μια τοπολογική διάταξη κορυφών του είναι η διάταξη:

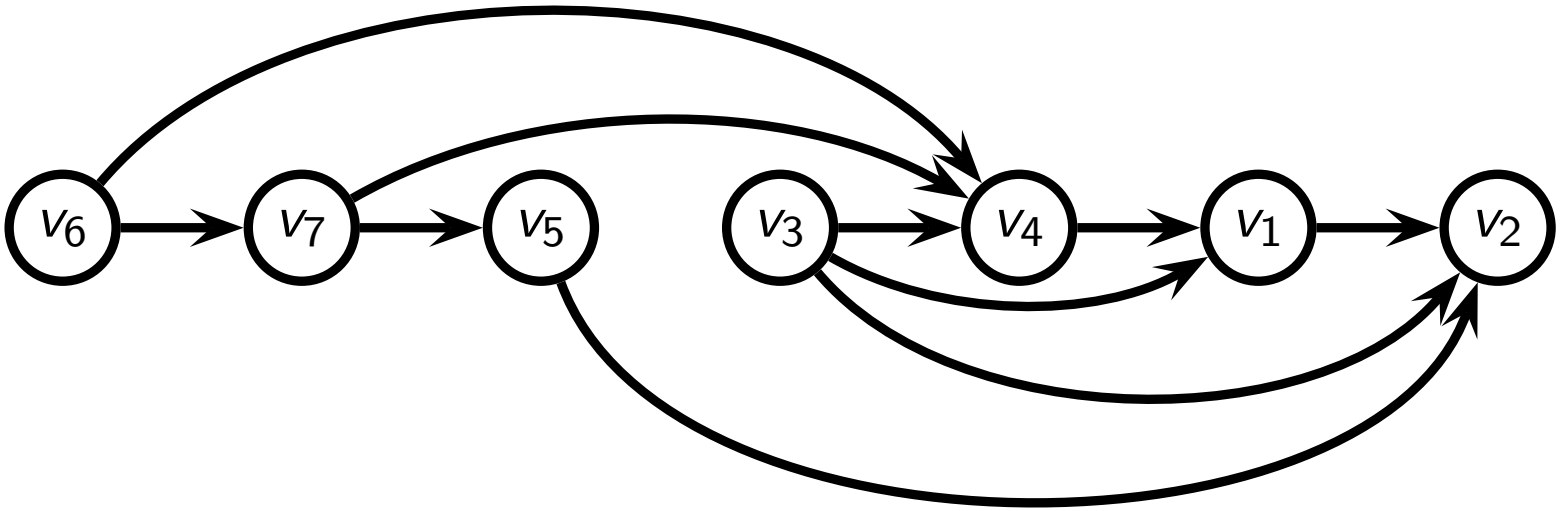
$$(v_3,\ v_6, v_7,\ v_5,\ v_4,\ v_1,\ v_2)$$

Μπορούμε εύκολα να επιβεβαιώσουμε ότι η σειρά αυτή ικανοποιεί τον ορισμό της τοπολογικής διάταξης σχεδιάζοντας ξανά το γράφημα, τοποθετώντας τις κορυφές του πάνω σε μια ευθεία με την σειρά που εμφανίζονται στην διάταξη. Τότε όλα τα τόξα έχουν προσανατολισμό από τα αριστερά προς τα δεξιά (από μικρότερη προς μεγαλύτερη κορυφή).

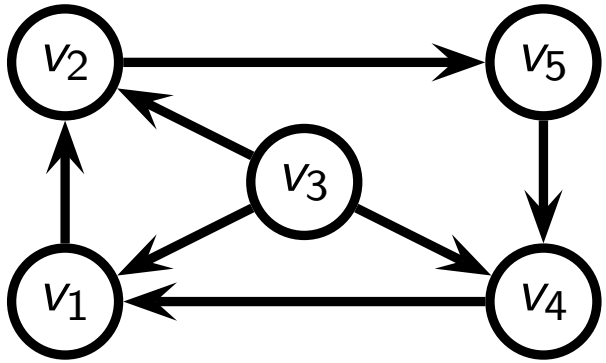


Μια άλλη τοπολογική διάταξη είναι η διάταξη:

$$(v_6,\ v_7, v_5,\ v_3,\ v_4,\ v_1,\ v_2)$$



Το επόμενο γράφημα τόξων δεν έχει τοπολογική διάταξη

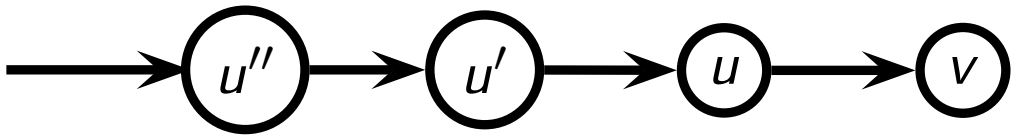


διότι η ύπαρξη του κυκλώματος $(v_1, v_2, v_5, v_4, v_1)$ παραβιάζει την μεταβατική ιδιότητα που έχουν όλες οι διατάξεις.

Πρόταση 20

Ένα γράφημα τόξων G έχει τοπολογική διάταξη αν και μόνο αν δεν περιέχει κυκλώματα.

Παρατήρηση. Σε κάθε γράφημα τόξων χωρίς κυκλώματα υπάρχει τουλάχιστον μια κορυφή με βαθμό εισόδου 0.
Πράγματι, έστω ότι δεν υπάρχει κορυφή με βαθμό εισόδου 0, τότε για κάθε κορυφή v υπάρχει προηγούμενη κορυφή u τέτοια ώστε $(u, v) \in U$.
Οπότε, αν ξεκινήσουμε από μια οποιαδήποτε κορυφή v μπορούμε να δημιουργήσουμε μια λίστα που κατασκευάζεται προθέτοντας την προηγούμενη κορυφή u' κάθε κορυφής u που είναι στην λίστα.



Επειδή το πλήθος των κορυφών είναι φραγμένο, η λίστα αυτή θα περιέχει επαναλήψεις, επομένως το γράφημα θα έχει κύκλωμα, άτοπο.

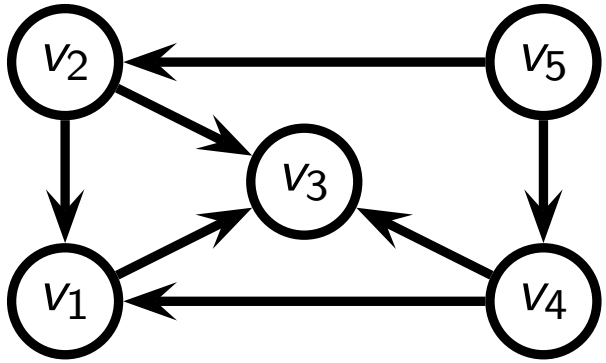
Η εύρεση μιας τοπολογικής διάταξης των κορυφών ενός γραφήματος τόξων G, αν υπάρχει, μπορεί να γίνει μέσω του επόμενου αλγόριθμου.

Αλγόριθμος τοπολογικής διάταξης Είσοδος: Ένα γράφημα τόξων G

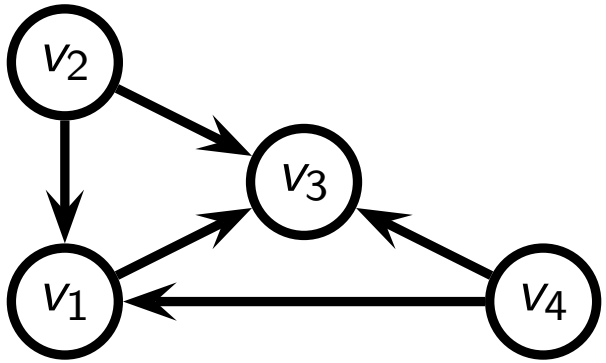
Έξοδος: Μια λίστα L με την τοπολογική διάταξη των κορυφών, αν υπάρχει.

- Όσο υπάρχουν κορυφές με βαθμό εισόδου 0 στο G
  - Επιλέγουμε μια κορυφή v του G με βαθμό εισόδου 0 και την τοποθετούμε στο τέλος της λίστας L.
  - Αφαιρούμε από το G την κορυφή v (καθώς και όλα τα τόξα που ξεκινούν από αυτήν).
- Αν το G δεν περιέχει άλλες κορυφές, τότε η σειρά των κορυφών στην λίστα L είναι μια τοπολογική διάταξη.
- Αλλιώς, το γράφημα περιέχει κύκλωμα και άρα δεν έχει τοπολογική διάταξη.

Παράδειγμα Να βρεθεί, αν υπάρχει, μια τοπολογική διάταξη των κορυφών του γραφήματος:

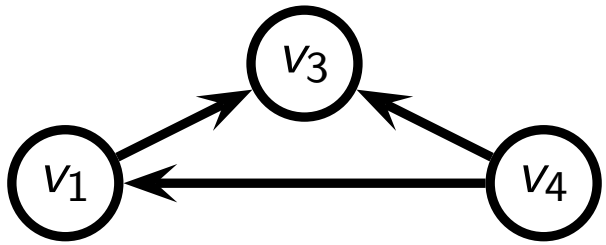


- Η μοναδική κορυφή με βαθμό εισόδου 0 είναι η v_5. Τοποθετούμε την v_5 στο τέλος της λίστας L και αφαιρούμε τα τόξα που αρχίζουν από την v_5



$L = (v_5)$

- Μετά την αφαίρεση της v_5 υπάρχουν δύο κορυφές με βαθμό εισόδου 0, οι v_2 και v_4. Επιλέγουμε την v_2, την τοποθετούμε στο τέλος της λίστας L, και αφαιρούμε όλα τα τόξα της



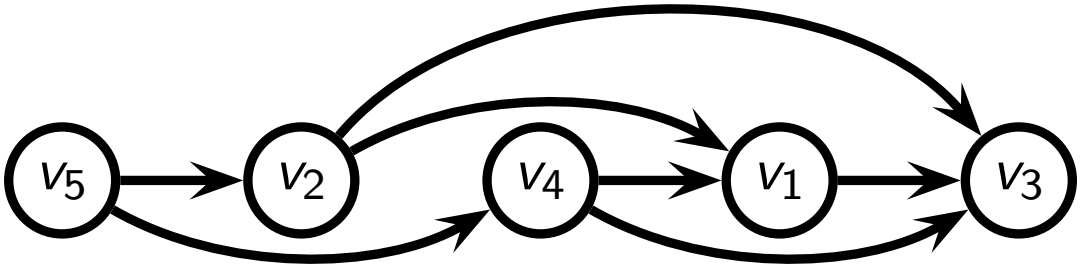
$$L = (v_5, v_2)$$

- Συνεχίζοντας με τον ίδιο τρόπο καταλήγουμε στο κενό γράφημα και στην λίστα

$$L = (v_5, v_2, v_4, v_1, v_3)$$

η οποία είναι τοπολογική διάταξη των κορυφών του γραφήματος.

Πράγματι,



Παρατήρηση. Για την εύρεση της τοπολογικής διάταξης ενός γραφήματος υπάρχει και άλλος αλγόριθμος που χρησιμοποιεί την αναζήτηση σε βάθος των κορυφών του.

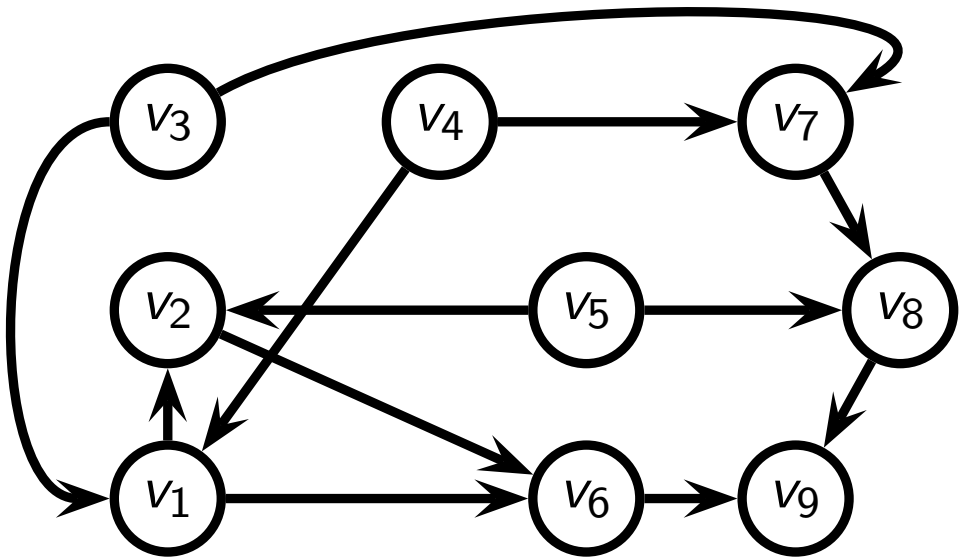
Άσκηση 1

Για την ολοκλήρωση ενός έργου πρέπει να εκτελεστούν 9 δραστηριότητες $T_1, T_2, \ldots, T_9$. Κάποιες από αυτές χρειάζονται τα αποτελέσματα μερικών άλλων, των οποίων η εκτέλεση πρέπει να προηγηθεί. Οι απαιτήσεις κάθε μιας δίδονται στον επόμενο πίνακα:

	απαιτήσεις		απαιτήσεις		απαιτήσεις
T_1	$T_3,\ T_4$	T_4		T_7	$T_3,\ T_4$
T_2	$T_1,\ T_5$	T_5		T_8	$T_5,\ T_7$
T_3		T_6	$T_1,\ T_2$	T_9	$T_6,\ T_8$

Να βρεθεί με ποια σειρά πρέπει να εκτελεστούν οι $T_1, T_2, \ldots, T_9$ ώστε να ολοκληρωθεί το έργο.

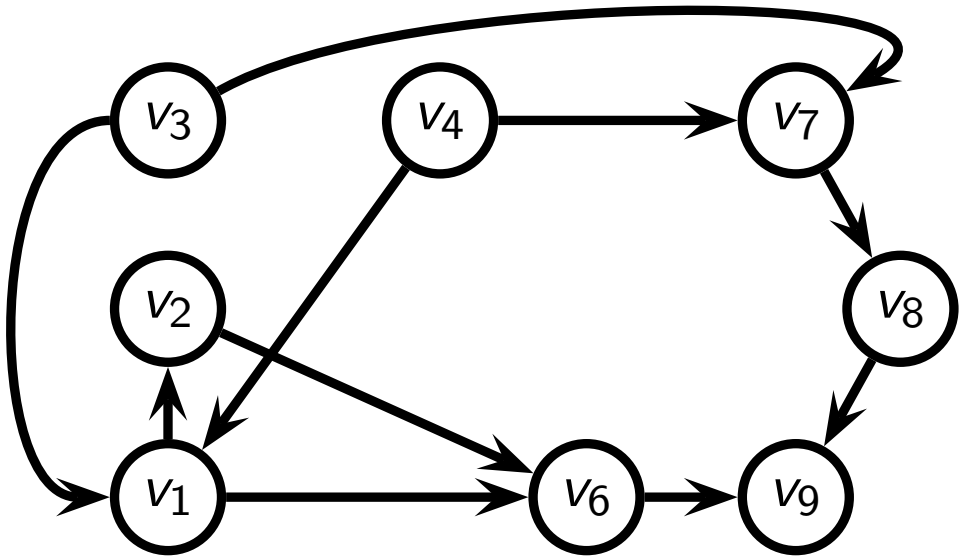
Στο πρόβλημα της εκτέλεσης αντιστοιχεί ένα κατευθυνόμενο γράφημα στο οποίο η δραστηριότητα T_i αναπαρίσταται από την κορυφή v_i, και αν η δραστηριότητα T_i απαιτεί την ολοκλήρωση της δραστηριότητας T_j τότε στην κορυφή v_i καταλήγει ένα τόξο με αρχή την κορυφή v_j.



Το πρόβλημα είναι ισοδύναμο με την εύρεση της τοπολογικής διάταξης των κορυφών του γραφήματος. Προκειμένου να υπάρχει λύση πρέπει να μην υπάρχει κύκλωμα.

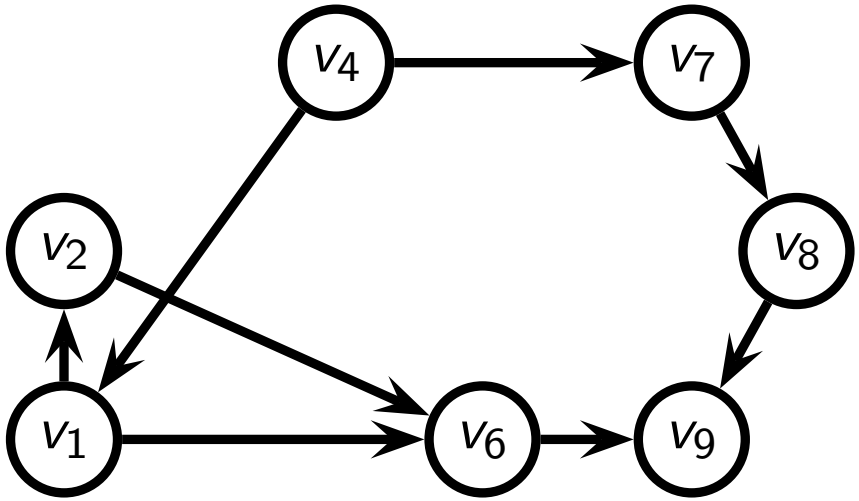
Θα εφαρμόσουμε τον αλγόριθμο της τοπολογικής διάταξης. Θα φτιάξουμε μια λίστα L που θα περιέχει τις κορυφές του γραφήματος με την σειρά της τοπολογικής διάταξης. Κάθε φορά επιλέγουμε μια κορυφή με βαθμό εισόδου 0, την προσθέτουμε στο τέλος της λίστας L και αφαιρούμε όλα τα τόξα που αρχίζουν από αυτή, μέχρις ότου να μην υπάρχουν κορυφές με βαθμό εισόδου 0 στο γράφημα.

1. Επιλέγουμε την v_5 οπότε έχουμε:



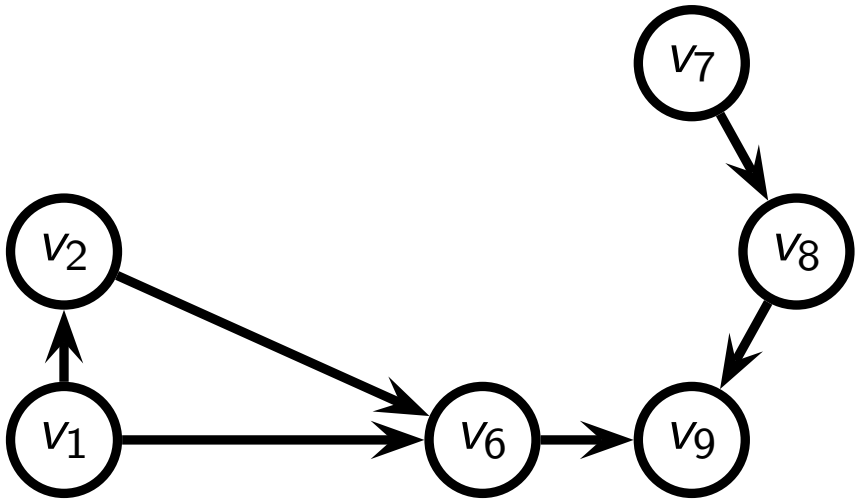
$$L = (v_5)$$

2. Επιλέγουμε την v_3 οπότε έχουμε:

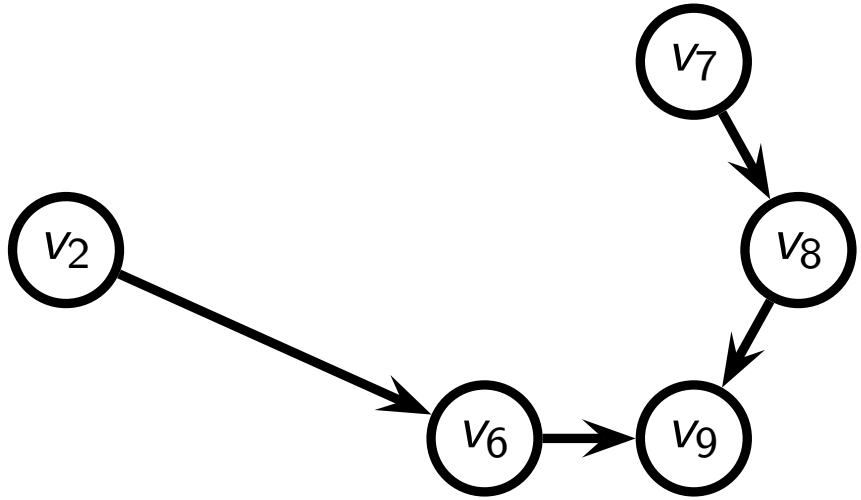


$$L = (v_5, v_3)$$

3. Επιλέγουμε την v_4 οπότε έχουμε:

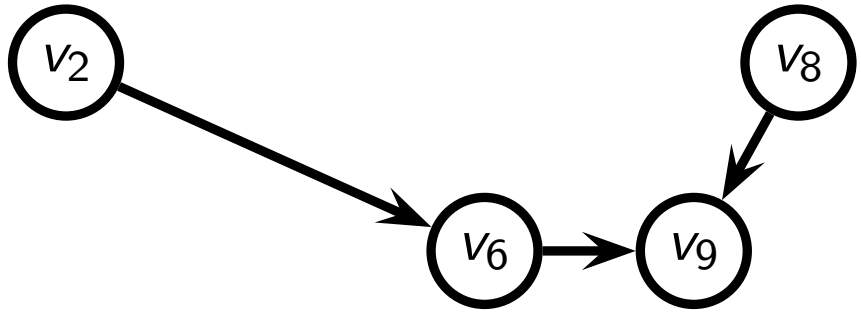


4. Επιλέγουμε την v_1 οπότε έχουμε:



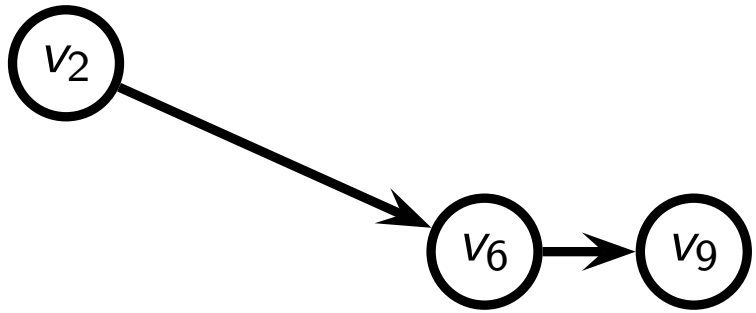
$$L = (v_5, v_3, v_4, v_1)$$

5. Επιλέγουμε την v_7 οπότε έχουμε:



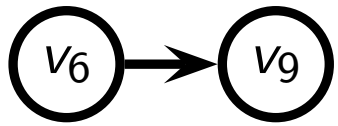
$$L = (v_5, v_3, v_4, v_1, v_7)$$

6. Επιλέγουμε την v_8 οπότε έχουμε:



$$L = (v_5, v_3, v_4, v_1, v_7, v_8)$$

7. Επιλέγουμε την v_2 οπότε έχουμε:



$$L = (v_5, v_3, v_4, v_1, v_7, v_8, v_2)$$

8. Επιλέγουμε την v_6 οπότε έχουμε:

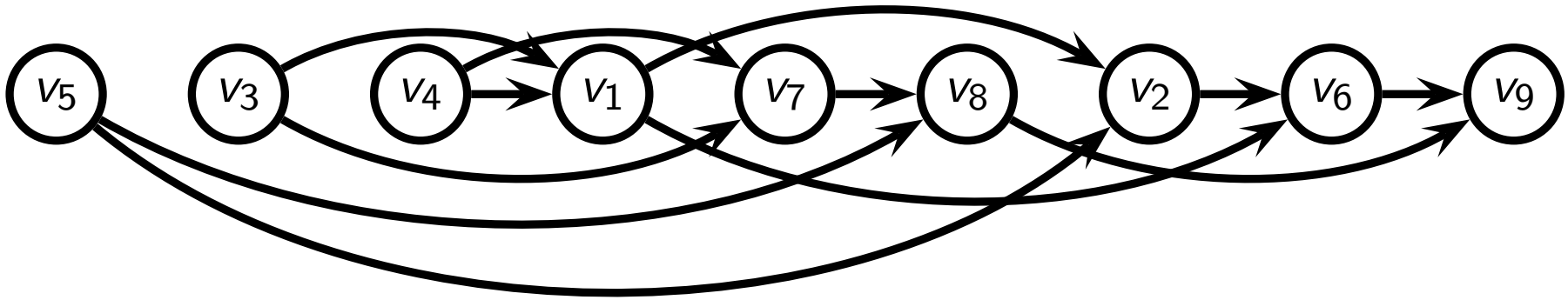
$$L = (v_5, v_3, v_4, v_1, v_7, v_8, v_2, v_6)$$

9. Επιλέγουμε την v_9 οπότε έχουμε:

$$L = (v_5, v_3, v_4, v_1, v_7, v_8, v_2, v_6, v_9)$$

Μια τοπολογική διάταξη των κορυφών του γραφήματος είναι η σειρά

$$(v_5, v_3, v_4, v_1, v_7, v_8, v_2, v_6, v_9)$$



Επομένως, μια πιθανή σειρά εκτέλεσης των 9 δραστηριοτήτων έτσι ώστε να ικανοποιούνται οι απαιτήσεις της καθεμιάς είναι η εξής:

$$(T_5, T_3, T_4, T_1, T_7, T_8, T_2, T_6, T_9)$$

Μπορούμε να υπολογίζουμε την τοπολογική διάταξη του G χρησιμοποιώντας την βιβλιοθήκη networkx:

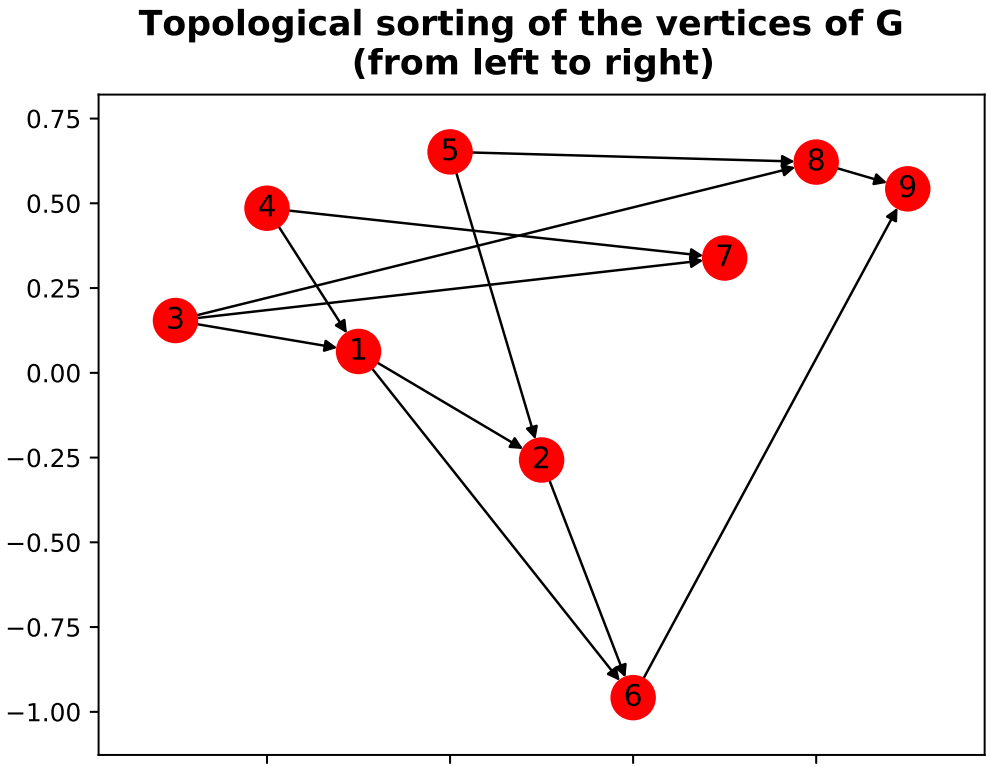
```
import networkx as nx
import matplotlib.pyplot as plt
import random

G = nx.DiGraph()
V = [1,2,3,4,5,6,7,8,9]
U = [(1,2),(1,6),(2,6),(3,1),(3,7),(4,1),(4,7),(5,2),(5,8),(6,9)
    ,(3,8),(8,9)]
G.add_nodes_from(V)
G.add_edges_from(U)

# Topological sorting of G using method topological_sort(G)
if(nx.is_directed_acyclic_graph(G)):
    R = list(nx.topological_sort(G))
    print("A topological sorting of G:",R,"(using the method
    topological_sort())")
else:
    print("G contains a cycle")
```

```
# A simple implementation of topological sorting algorithm
L = [] #Topological sorting of the nodes of G
H = G.copy() #Work on an copy of G
while(len(H)!=0):
    notfound = True
    for v in H.nodes():
        if (H.in_degree(v) == 0):
            L.append(v)
            H.remove_node(v)
            notfound = False # vertex v has indegree 0
            break
    if notfound: break # no vertex has indegree 0
# If L contains all the nodes of G a solution is found
if len(L) == len(G):
    print("A topological sorting of G:",L)
else:
    print("G contains a cycle")

# Position all vertices of G in a line according to the topological
     sorting L
x = y = 0
mypos = {} #empty dictionary
for i in L:
    x = x + 1
    y = random.uniform(-1,1)
    mypos[i] = [x,y] #coordinates of node i
```

```
# Create a figure in order to add title text
fig = plt.figure()
fig.suptitle('Topological sorting of the vertices of G \n (from
    left to right)', fontsize=14, fontweight='bold')
nx.draw_networkx(G,pos=mypos)
plt.show()
```

Output:

```
A topological sorting of G: [5, 4, 3, 8, 7, 1, 2, 6, 9] (using the
    method topological_sort())
A topological sorting of G: [3, 4, 1, 5, 2, 6, 7, 8, 9]
```



Συνάρτηση Grundy - Sprague

Έστω $A \subseteq \mathbb{N}$. Ορίζουμε το mex **(minimum excluded value)** του A ως εξής:

$$\operatorname{mex} A = \min \mathbb{N} \setminus A,$$

δηλαδή $\operatorname{mex} A$ **είναι ο ελάχιστος φυσικός αριθμός που δεν ανήκει στο** A.

Παράδειγμα

- $\operatorname{mex}\{1, 2, 4\} = 0$
- $\operatorname{mex}\{0, 1, 2, 6\} = 3$
- $\operatorname{mex}\{0, 1, 2, 3\} = 4$
- $\operatorname{mex} \emptyset = 0$.

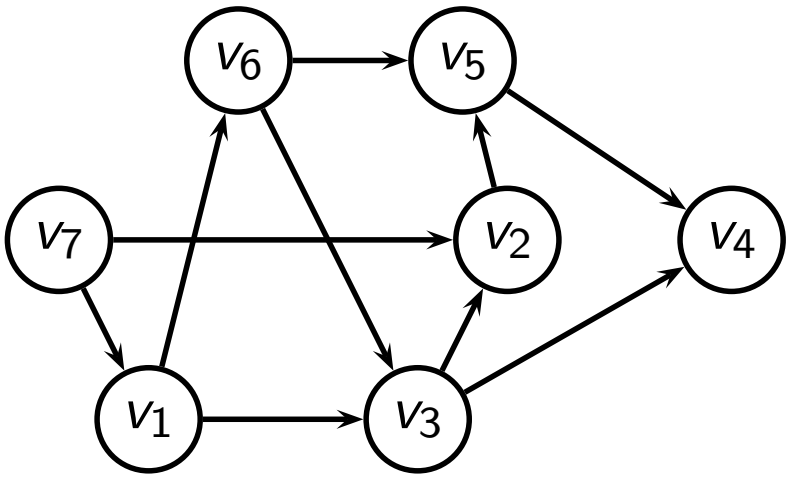
Για κάθε κορυφή v ενός γραφήματος τόξων $G = (V, U)$ με $\Gamma(v)$ συμβολίζουμε το σύνολο των κορυφών u που είναι άκρα τόξων με αρχή την κορυφή v (**γείτονες της** v), δηλαδή

$$\Gamma(v) = \{u \in X : (v, u) \in U\}$$

Παράδειγμα Για το γράφημα τόξων $G = (V, U)$ όπου $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ και

$U = \{(v_1, v_3), (v_1, v_6), (v_2, v_5), (v_3, v_2), (v_3, v_4), (v_5, v_4), (v_6, v_3), (v_6, v_5), (v_7, v$

Έχουμε ότι



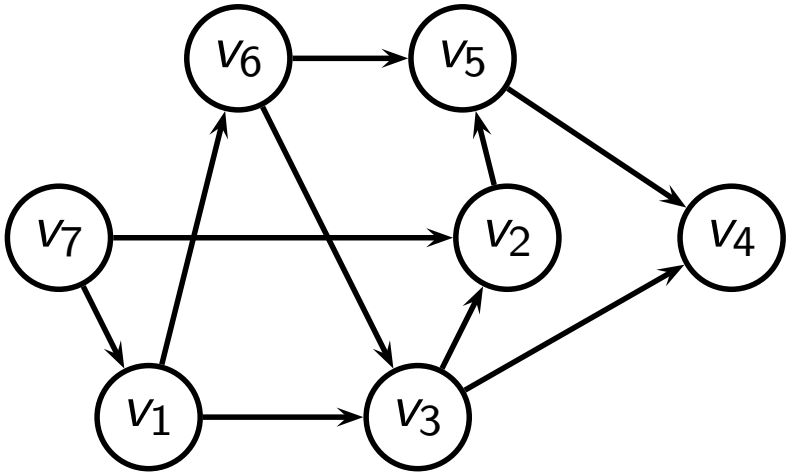
$$\begin{aligned}
\Gamma(v_1) &= \{v_3, v_6\} \\
\Gamma(v_2) &= \{v_5\} \\
\Gamma(v_3) &= \{v_2, v_4\} \\
\Gamma(v_4) &= \emptyset \\
\Gamma(v_5) &= \{v_4\} \\
\Gamma(v_6) &= \{v_3, v_5\} \\
\Gamma(v_7) &= \{v_1, v_2\}.
\end{aligned}$$

Σε κάθε γράφημα τόξων $G = (V, U)$ **χωρίς κυκλώματα** αντιστοιχεί μια (μοναδική) συνάρτηση $g : V \to \mathbb{N}$ η οποία ονομάζεται **συνάρτηση Grundy - Sprague** του G.
Συγκεκριμένα, για κάθε κορυφή v του γραφήματος ορίζουμε

$$g(v) = \text{mex}\{g(u) : \text{ για κάθε } u \in \Gamma(v)\}.$$

Η συνάρτηση g τοποθετεί στις κορυφές του γραφήματος G φυσικούς αριθμούς, έτσι ώστε κάθε κορυφή v να έχει τιμή $g(v)$ τον ελάχιστο φυσικό αριθμό που δεν έχει τοποθετηθεί στους γειτονές της. Η τιμή $g(v)$ ονομάζεται g-**value** της κορυφής v.

Παράδειγμα Να βρεθούν οι τιμές της συνάρτησης g του γραφήματος τόξων G:



$$\begin{aligned}
\Gamma(v_1) &= \{v_3, v_6\} \\
\Gamma(v_2) &= \{v_5\} \\
\Gamma(v_3) &= \{v_2, v_4\} \\
\Gamma(v_4) &= \emptyset \\
\Gamma(v_5) &= \{v_4\} \\
\Gamma(v_6) &= \{v_3, v_5\} \\
\Gamma(v_7) &= \{v_1, v_2\}.
\end{aligned}$$

Το G δεν έχει κυκλώματα, άρα ορίζεται η συνάρτηση Grundy - Sprague σ' αυτό. Στόχος μας είναι να συμπληρώσουμε τις κενές θέσεις στον επόμενο πίνακα

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$							

$$g(v) = \text{mex}\{g(u) : \text{ για κάθε } u \in \Gamma(v)\},$$

Έχουμε ότι

$$\Gamma(v_1) = \{v_3, v_6\}$$
$$\Gamma(v_2) = \{v_5\}$$
$$\Gamma(v_3) = \{v_2, v_4\}$$
$$\Gamma(v_4) = \emptyset$$
$$\Gamma(v_5) = \{v_4\}$$
$$\Gamma(v_6) = \{v_3, v_5\}$$
$$\Gamma(v_7) = \{v_1, v_2\}.$$

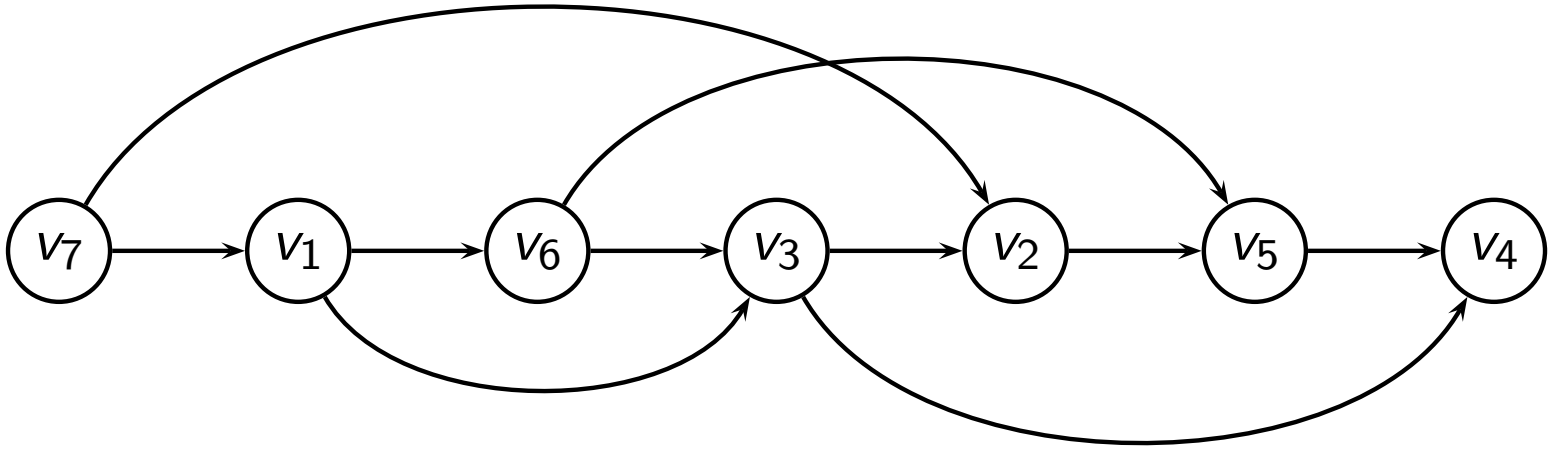
οπότε

$$g(v_1) = \mathrm{mex}\{\mathrm{g(v_3)}, \mathrm{g(v_6)}\}$$
$$g(v_2) = \mathrm{mex}\{\mathrm{g(v_5)}\}$$
$$g(v_3) = \mathrm{mex}\{\mathrm{g(v_2)}, \mathrm{g(v_4)}\}$$
$$g(v_4) = \mathrm{mex}\emptyset$$
$$g(v_5) = \mathrm{mex}\{\mathrm{g(v_4)}\}$$
$$g(v_6) = \mathrm{mex}\{\mathrm{g(v_3)}, \mathrm{g(v_5)}\}$$
$$g(v_7) = \mathrm{mex}\{\mathrm{g(v_1)}, \mathrm{g(v_2)}\}.$$

Παρατηρούμε ότι:
Για να υπολογισθεί το $g(v_1)$ πρέπει να υπολογισθούν πρώτα τα $g(v_3)$, $g(v_6)$.
Για να υπολογισθεί το $g(v_2)$ πρέπει να υπολογισθεί πρώτα το $g(v_5)$,
Για να υπολογισθεί το $g(v_3)$ πρέπει να υπολογισθούν πρώτα τα $g(v_2)$, $g(v_4)$.
Για να υπολογισθεί το $g(v_5)$ πρέπει να υπολογισθεί πρώτα το $g(v_4)$,
Για να υπολογισθεί το $g(v_6)$ πρέπει να υπολογισθούν πρώτα τα $g(v_3)$,

Επομένως, για να υπολογίσουμε τις τιμές της συνάρτησης g πρέπει να βρούμε (αν υπάρχει) μια σειρά στις κορυφές του, σύμφωνα με την οποία να υπολογίζουμε την τιμή $g(v)$ μια κορυφής αφού πρώτα έχουμε ήδη υπολογίσει τις τιμές $g(u)$ όλων των γειτόνων u της v.
Συνεπώς, ο υπολογισμός της συνάρτησης g είναι πρόβλημα προτεραιοτήτων, επομένως μπορεί να λυθεί υπολογίζοντας μια τοπολογική διάταξη στο σύνολο των κορυφών του γραφήματος G. Η διαφοροποίηση στο πρόβλημα αυτό είναι ότι το τόξο (v, u) λειτουργεί ανάποδα, αφού για τον υπολογισμό της $g(v)$ χρειαζόμαστε την τιμή $g(u)$.
Άρα, για το πρόβλημα αυτό, θα βρούμε μια τοπολογική διάταξη στο γράφημα G και έπειτα θα υπολογίσουμε την συνάρτηση με την ανάστροφη σειρά.

Μια τέτοια διάταξη απεικονίζεται στο επόμενο σχήμα



(από την κορυφή v_i ξεκινάει βέλος προς την κορυφή v_j αν ο υπολογισμός της $g(v_i)$ απαιτεί τον υπολογισμό της $g(v_j)$).
Θα υπολογίσουμε τις τιμές της συνάρτησης g με την ανάστροφη σειρά

$$v_4, v_5, v_2, v_3, v_6, v_1, v_7$$

οπότε σε κάθε βήμα οι τιμές της g που απαιτούνται έχουν υπολογισθεί σε κάποιο προηγούμενο βήμα.

Πράγματι, έχουμε τα εξής:

1. $g(v_4) = \text{mex}\emptyset = 0.$

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$				0			

2. $g(v_5) = \text{mex}\{g(v_4)\} = \text{mex}\{0\} = 1.$

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$				0	1		

3. $g(v_2) = \text{mex}\{\text{g}(\text{v}_5)\} = \text{mex}\{1\} = 0.$

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0		0	1		

4. $g(v_3) = \text{mex}\{\text{g}(\text{v}_2), \text{g}(\text{v}_4)\} = \text{mex}\{0, 0\} = \text{mex}\{0\} = 1.$

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0	1	0	1		

5. $g(v_6) = \text{mex}\{g(v_3), g(v_5)\} = \text{mex}\{1, 1\} = \text{mex}\{1\} = 0.$

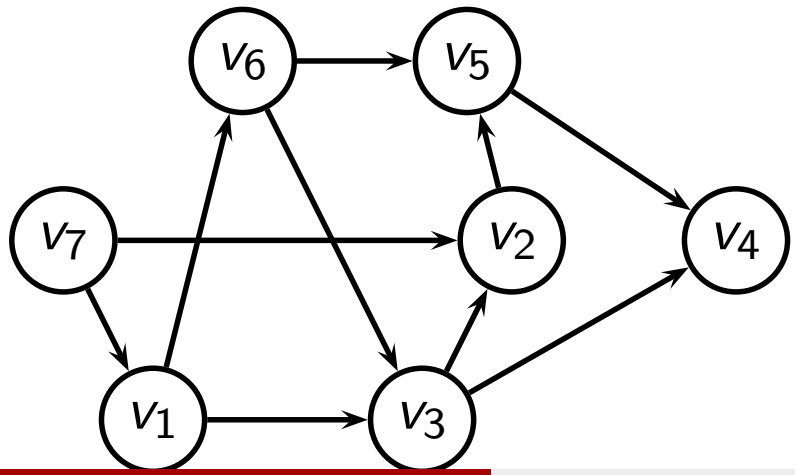
v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0	1	0	1	0	

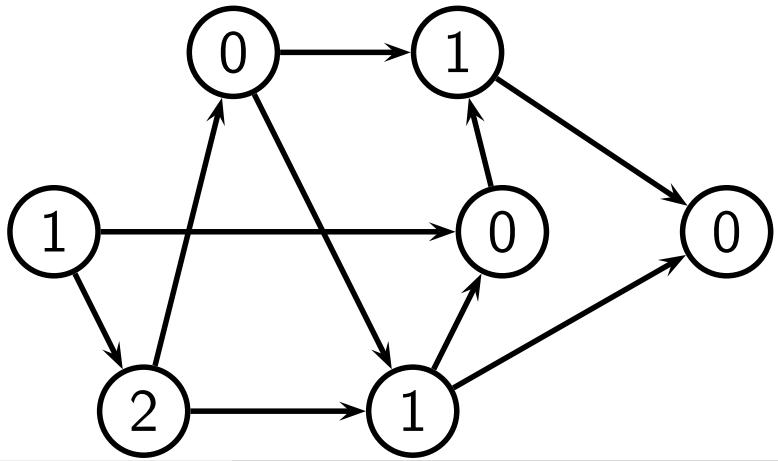
6. $g(v_1) = \text{mex}\{g(v_3), g(v_6)\} = \text{mex}\{1, 0\} = 2.$

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$	2	0	1	0	1	0	

7. Τέλος, $g(v_7) = \text{mex}\{g(v_1), g(v_2)\} = \text{mex}\{2, 0\} = 1.$

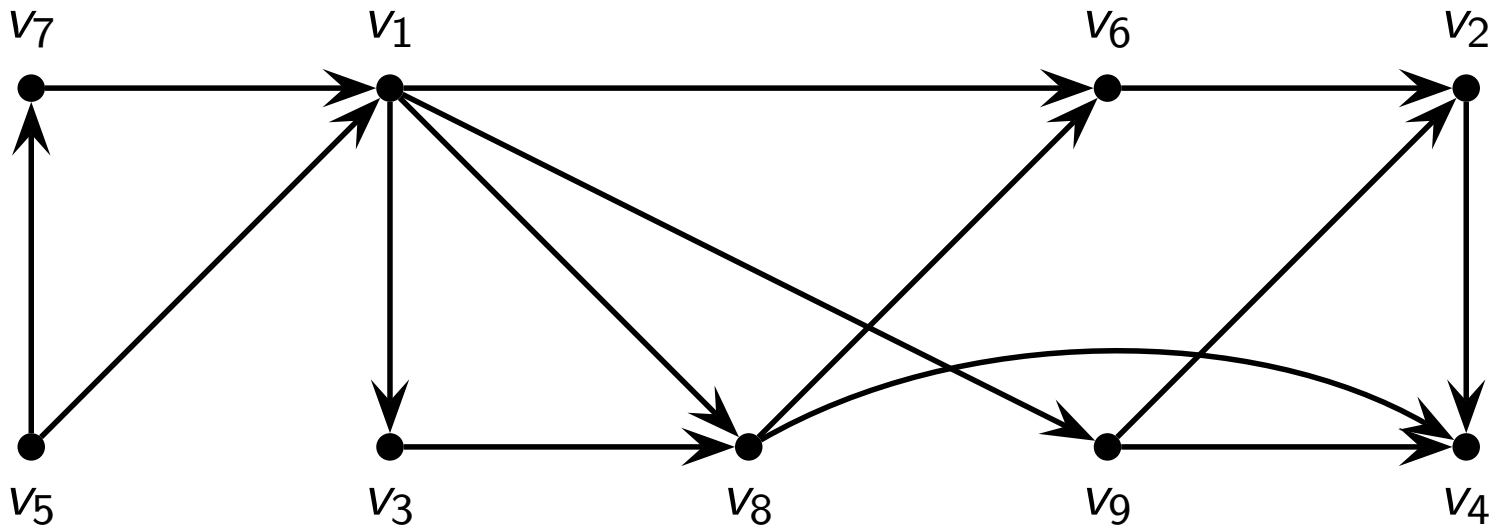
v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$	2	0	1	0	1	0	1.





Άσκηση 2

Να υπολογισθούν οι τιμές της συνάρτησης Grundy - Sprague για τις κορυφές του επόμενου γραφήματος τόξων.



Παρατήρηση. Η συνάρτηση Grundy - Sprague ικανοποιεί τις παρακάτω ιδιότητες

- Αν για την κορυφή v ισχύει ότι $g(v) \neq 0$, τότε υπάρχει $u \in \Gamma(v)$ με $g(u) = 0$.
- Αν για την κορυφή v ισχύει ότι $g(v) = 0$, τότε για κάθε $u \in \Gamma(v)$ ισχύει ότι $g(u) \neq 0$.

Μια εφαρμογή της συνάρτησης Grundy - Sprague είναι ότι

- αποδεικνύει ότι υπάρχει στρατηγική νίκης σε συνδυαστικά παιχνίδια δύο παιχτών όπου έχουμε αποκλείσει την ισοπαλία (π.χ. σκάκι), δηλαδή αποδεικνύει ότι κάποιος από τους δύο παίχτες μπορεί πάντα να κερδίζει (αρκεί να γνωρίζει την στρατηγική).
- δίνει την στρατηγική νίκης στο παιχνίδι αυτό, αρκεί να είναι υπολογιστικά εφικτός ο υπολογισμός της συνάρτησης g. Η στρατηγική νίκης είναι (πάντα) να φτάνουμε σε καταστάσεις με τιμή g ίση με 0.

Μια παραλλαγή του Nim

Δύο παίχτες A και B παίζουν το εξής παιχνίδι:
Υπάρχουν δύο σωροί με σπίρτα. Ο πρώτος σωρός αποτελείται από x σπίρτα και ο δεύτερος σωρός από y σπίρτα.
Οι παίχτες παίζουν εναλλάξ κάνοντας **μια κίνηση ο καθένας**: Κάθε παίχτης αφαιρεί ένα ή περισσότερα σπίρτα

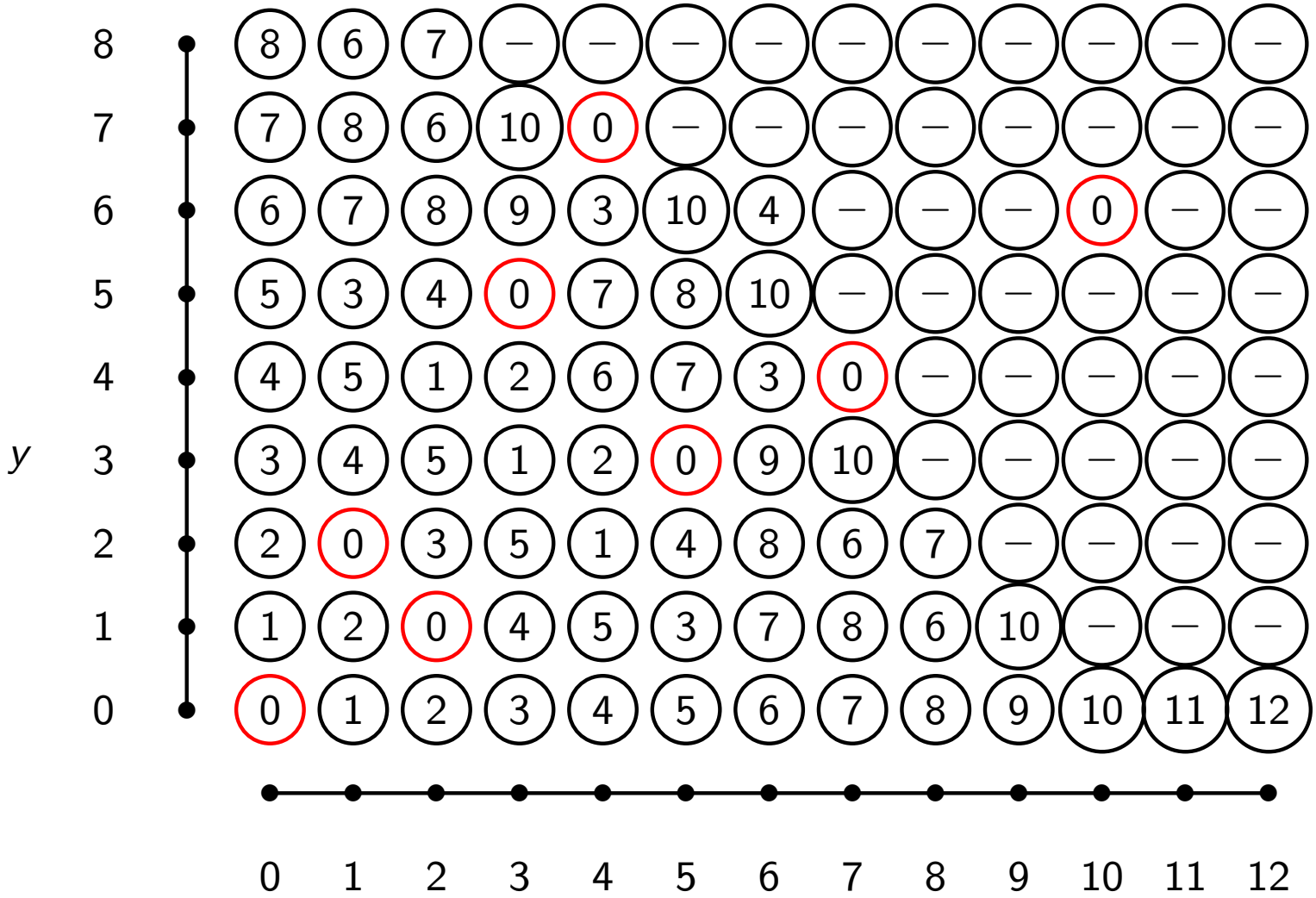
- ή από τον πρώτο σωρό,
- ή από τον δεύτερο σωρό,
- ή τον ίδιο αριθμό και από τους δύο σωρούς.

Χάνει ο παίχτης που δεν μπορεί να κάνει κίνηση.

- Για παράδειγμα, αν ο πρώτος σωρός περιέχει 12 σπίρτα και ο δεύτερος σωρός 8 σπίρτα ποιος από τους δύο παίχτες θα κερδίσει;

Μπορούμε να μοντελοποίησουμε το παιχνίδι ως ένα γράφημα τόξων. Οι κορυφές είναι οι καταστάσεις του παιχνιδιού. Κάθε κορυφή αντιστοιχεί σε ένα διατεταγμένο ζεύγος (x, y) όπου x ο αριθμός των σπίρτων στον πρώτο σωρό και y ο αριθμός των σπίρτων στον δεύτερο σωρό.
Ισχύει ότι $0 \leq x \leq 12$ και $0 \leq y \leq 8$. Συνολικά υπάρχουν $13 \times 9 = 117$ καταστάσεις.
Αρχικά το παιχνίδι βρίσκεται στην κορυφή (κατάσταση) $(12, 8)$. Με κάθε κίνηση των παιχτών το παιχνίδι αλλάζει καταστάσεις. Οι επιτρεπτές κινήσεις αντιστοιχούν στην μετακίνηση οριζόντια (προς τα αριστερά) ή κατακόρυφα (προς τα κάτω) ή διαγώνια (προς το κέντρο) οσαδήποτε βήματα. Κερδίζει ο παίχτης που θα φτάσει πρώτος στην κορυφή $(0, 0)$.

Στο επόμενο σχήμα απεικονίζονται μόνο οι κορυφές του γραφήματος. Τα τόξα ορίζονται από τις επιτρεπτές κινήσεις. Επίσης, έχουν σημειωθεί οι τιμές της συνάρτησης Grundy - Sprague. Οι μηδενικές τιμές (τιμές με κόκκινο) ορίζουν την νικηφόρα στρατηγική του παιχνιδιού.



Στο παράδειγμα, μπορεί να κερδίζει πάντα ο πρώτος παίχτης, διότι αφαιρώντας 2 σπίρτα και από τους δύο σωρούς καταλήγει στην κατάσταση $(10, 6)$ η οποία έχει τιμή g ίση με 0.

Παρατήρηση. Σε κάθε συνδυαστικό παιχνίδι δύο παιχτών (χωρίς ισοπαλία) μπορούμε να κατασκευάσουμε το γράφημα των καταστάσεων του παιχνιδιού. Το γράφημα αυτό δεν θα περιέχει κυκλώματα (αφού δεν επιτρέπεται ισοπαλία).

Πόρισμα 2

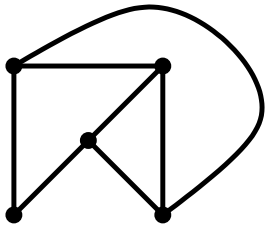
Κάθε συνδυαστικό παιχνίδι δύο παιχτών (χωρίς ισοπαλία) έχει συνάρτηση g. Επομένως, μπορεί να κερδίζει πάντα ή ο πρώτος, ή ο δεύτερος παίχτης. (Αρκεί να γνωρίζει τις καταστάσεις που μηδενίζουν την συνάρτηση g.)

8η ΔΙΑΛΕΞΗ
ΓΡΑΦΗΜΑΤΑ

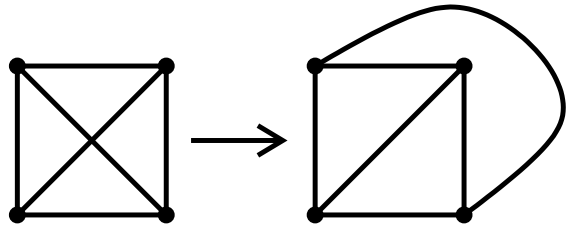
- **Επίπεδα γραφήματα**
- **Τύπος του Euler**
- **Ανισότητες δεσμών – εδρών - κορυφών**
- **Θεώρημα του Kuratowski**

Επίπεδα γραφήματα

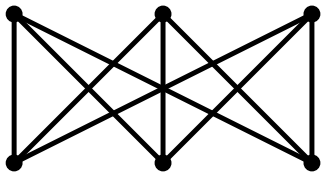
Επίπεδο (planar) λέγεται ένα γράφημα που μπορεί να απεικονισθεί στο επίπεδο, έτσι ώστε :
α) Οι κόμβοι του να είναι διακεκριμένα σημεία.
β) Οι δεσμοί του να είναι απλές, επίπεδες καμπύλες.
γ) Κάθε ζεύγος δεσμών (αν συναντιούνται), συναντιούνται μόνο στους κόμβους.
Παράδειγμα :



Επίπεδο γράφημα



Επίπεδο γράφημα

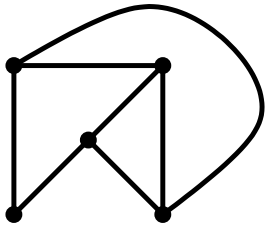


Μη επίπεδο γράφημα

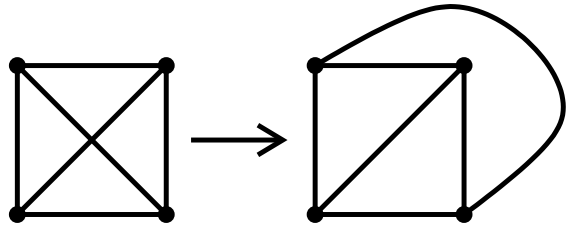
Επίπεδο τοπολογικο γράφημα

Επίπεδο τοπολογικό γράφημα (plane graph) λέγεται ένα επίπεδο γράφημα που έχει ήδη απεικονισθεί στο επίπεδο, έτσι ώστε να ικανοποιεί τις συνθήκες α), β) και γ).

Παράδειγμα : Το πρώτο και το τρίτο από τα επόμενα γραφήματα είναι επίπεδο τοπολογικό γράφημα.



Επίπεδο γράφημα



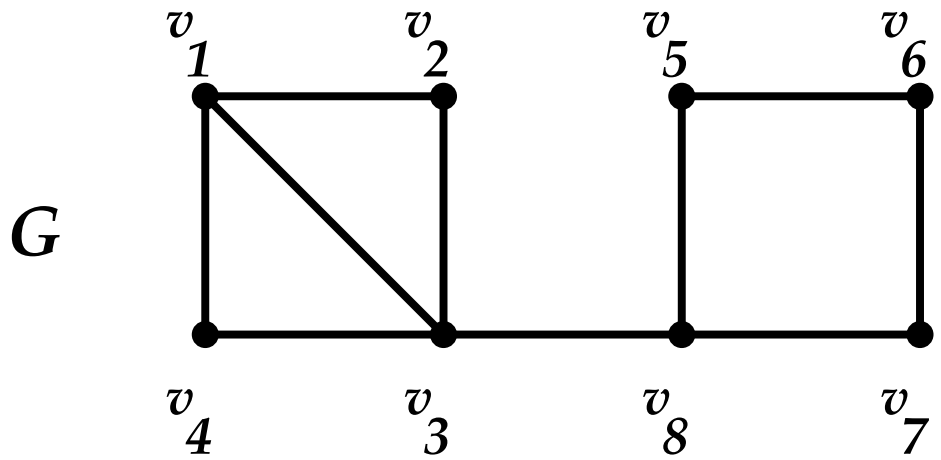
Επίπεδο γράφημα

Παρατήρηση : Στο ‘‘**πρόβλημα σύνδεσης**’’ το ερώτημα που τίθεται (κατά πόσον κάποιοι κόμβοι μπορούν να συνδεθούν με κάποιους άλλους χωρίς να υπάρχουν ‘‘διασταυρώσεις’’) είναι ουσιαστικά το ερώτημα : κατά πόσον το προκύπτον γράφημα είναι επίπεδο. (Εφαρμογές : Πληροφορική, ηλεκτρολογία (συνδεσμολογίες), συγκοινωνίες κ.λπ.). Ένα θεώρημα σχετικό με το θέμα αυτό (ικανή και αναγκαία συνθήκη για να είναι ένα γράφημα μη επίπεδο) είναι το διάσημο θεώρημα του Kuratowski, που θα δούμε αργότερα.

Έδρες επίπεδου (τοπολογικού) γραφήματος

Ορίζουμε σαν **έδρες** (faces) ενός επίπεδου τοπολογικού γραφήματος G τις κλειστές περιοχές του επίπεδου που ορίζονται από τους δεσμούς του γραφήματος. Η ανοιχτή περιοχή ονομάζεται **εξωτερική έδρα** (outer face) του G.

Παράδειγμα :



Το γράφημα G έχει τρεις έδρες : Τα ''τρίγωνα'' που ορίζονται από τους κύκλους (v_1, v_2, v_3, v_1) και (v_1, v_3, v_4, v_1) και το ''τετράγωνο'' που ορίζεται από τον κύκλο $(v_5, v_6, v_7, v_8, v_5)$.

Τύπος του Euler

Για τα επίπεδα γραφήματα ισχύει η επόμενη αξιοσημείωτη πρόταση:

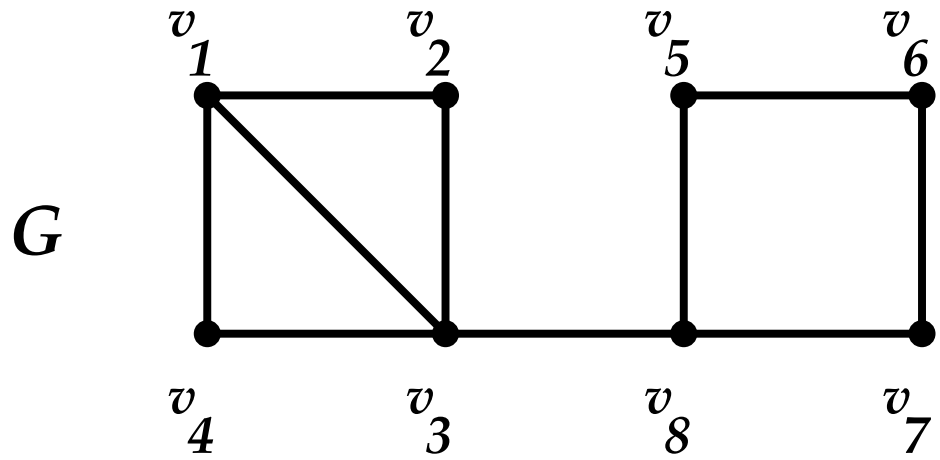
Πρόταση 21 (Τύπος του Euler)

Έστω $G = (V, E)$ ένα συνεκτικό επίπεδο τοπολογικό γράφημα με $|F| = |F(G)|$ έδρες (συμπεριλαμβανόμενης και της εξωτερικής). Τότε

$$|V| - |E| + |F| = 2.$$

Η απόδειξη γίνεται με επαγωγή (ως προς τον αριθμό των εδρών).

Παράδειγμα : Στο γράφημα G



έχουμε :
$|V| = 8$, $|E| = 10$, $|F| = 4$ και πράγματι $8 - 10 + 4 = 2$.

Παρατηρήσεις :

1. Από τον τύπο του Euler προκύπτει ότι αν ένα γράφημα είναι επίπεδο κάθε αναπαράσταση του ως επίπεδο τοπολογικό γράφημα θα έχει πάντα τον ίδιο αριθμό εδρών.
2. Ο τύπος του Euler ισχύει και για μη απλά γραφήματα, δηλαδή για γραφήματα που περιέχουν βρόχους ή και πολλαπλούς δεσμούς ανάμεσα στις κορυφές τους.
3. Ο τύπος του Euler ισχύει και για γραφήματα τα οποία έχουν απεικονισθεί πάνω σε μια σφαίρα, έτσι ώστε να ικανοποιούν τις συνθήκες α), β) και γ). Σε αυτή την περίπτωση ιδιαίτερο ενδιαφέρον έχουν τα πολύεδρα με κορυφές πάνω στην επιφάνεια της σφαίρας.

Ανισότητες εδρών – δεσμών

Λήμμα 1

1. *Σε κάθε επίπεδο συνεκτικό γράφημα* $G = (V, E)$, *με* $|E| \geq 2$, *ισχύει ότι*
$$3|F| \leq 2|E|$$
2. *Σε κάθε επίπεδο διμερές συνεκτικό γράφημα* $G = (V, E)$ *με* $|E| \geq 2$ *ισχύει ότι*
$$2|F| \leq |E|$$

Απόδειξη. Έστω $s(G)$ ο αριθμός των ζευγών (e, f) του G για τα οποία ο δεσμός e συνορεύει με την έδρα f, δηλαδή

$$s(G) = \sum_{e,f \text{ συνορεύουν}} 1$$

Μπορούμε να μετρήσουμε τα ζεύγη (e, f) με δύο τρόπους: Αθροίζοντας για κάθε έδρα f το πλήθος των δεσμών e που συνορεύουν με αυτήν, ή αθροίζοντας για κάθε δεσμό e το πλήθος των εδρών f οι οποίες συνορεύουν με αυτόν.

1. Με τον πρώτο τρόπο, κάθε έδρα συνορεύει με τουλάχιστον 3 δεσμούς, επομένως

$$s(G) = \sum_{f} \sum_{\substack{e \\ e \text{ συνορεύει με } f}} 1 \geq \sum_{f} 3 = 3|F|$$

Με τον δεύτερο τρόπο, κάθε δεσμός συνορεύει το πολύ με δύο έδρες, οπότε

$$s(G) = \sum_{e} \sum_{\substack{f \\ f \text{ συνορεύει με } e}} 1 \leq \sum_{e} 2 = 2|E|$$

Άρα

$$3|F| \leq s(G) \leq 2|E|.$$

2. Επειδή το γράφημα είναι διμερές, δεν περιέχει κύκλους περιττού μήκους, άρα, με τον πρώτο τρόπο, κάθε έδρα συνορεύει με τουλάχιστον 4 δεσμούς, επομένως

$$s(G) \geq 4|F|$$

Με τον πρώτο τρόπο, κάθε δεσμός συνορεύει το πολύ με δύο έδρες, οπότε

$$s(G) \leq 2|E|$$

Άρα

$$4|F| \leq 2|E| \Leftrightarrow 2|F| \leq |E|.$$

Ανισότητα κορυφών – δεσμών

Πόρισμα 3 (**Ανισότητα κορυφών-δεσμών**)

1. *Σε κάθε επίπεδο συνεκτικό γράφημα* $G = (V, E)$ *ισχύει ότι*

$$|E| \leq 3|V| - 6$$

2. *Σε κάθε επίπεδο διμερές συνεκτικό γράφημα* $G = (V, E)$ *ισχύει ότι*

$$|E| \leq 2|V| - 4$$

Απόδειξη.

1. Από το προηγούμενο λήμμα για κάθε επίπεδο γράφημα $G = (V, E)$ ισχύει ότι

$$3|F| \leq 2|E|$$

Όμως, από τον τύπο του Euler έχουμε ότι $|F| = |E| - |V| + 2$ οπότε

$$3(|E| - |V| + 2) \leq 2|E| \Leftrightarrow |E| \leq 3|V| - 6$$

2. Άσκηση.

Άσκηση 1

Τα γραφήματα K_5 και $K_{3,3}$ δεν είναι επίπεδα.

Λύση Για το γράφημα K_5 έχουμε ότι $|V(K_5)| = 5$ και $|E(K_5)| = \binom{5}{2} = 10$.
Από το προηγούμενο πόρισμα, έχουμε ότι αν το K_5 είναι επίπεδο πρέπει

$$|E(K_5)| \leq 3|V(K_5)| - 6 \Leftrightarrow 10 \leq 3 \cdot 5 - 6 \Leftrightarrow 10 \leq 9$$

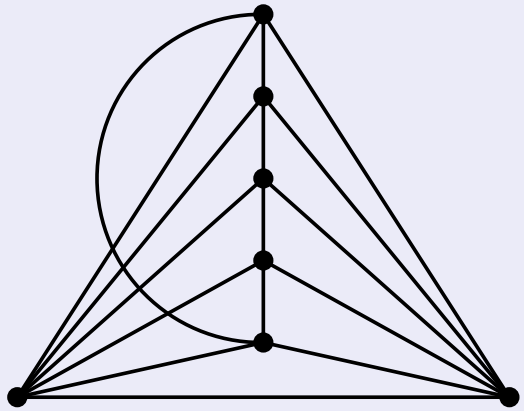
άτοπο, άρα το K_5 δεν είναι επίπεδο.
Για το γράφημα $K_{3,3}$ έχουμε ότι $|V(K_{3,3})| = 6$ και $|E(K_{3,3})| = 3 \cdot 3 = 9$.
Επειδή το $K_{3,3}$ είναι διμερές, από το προηγούμενο πόρισμα, έχουμε ότι αν το $K_{3,3}$ είναι επίπεδο πρέπει

$$|E(K_{3,3})| \leq 2|V(K_{3,3})| - 4 \Leftrightarrow 9 \leq 2 \cdot 6 - 4 \Leftrightarrow 9 \leq 8$$

άτοπο, άρα το $K_{3,3}$ δεν είναι επίπεδο.

Άσκηση 2

Να εξετασθεί αν το παρακάτω γράφημα είναι επίπεδο ή όχι.

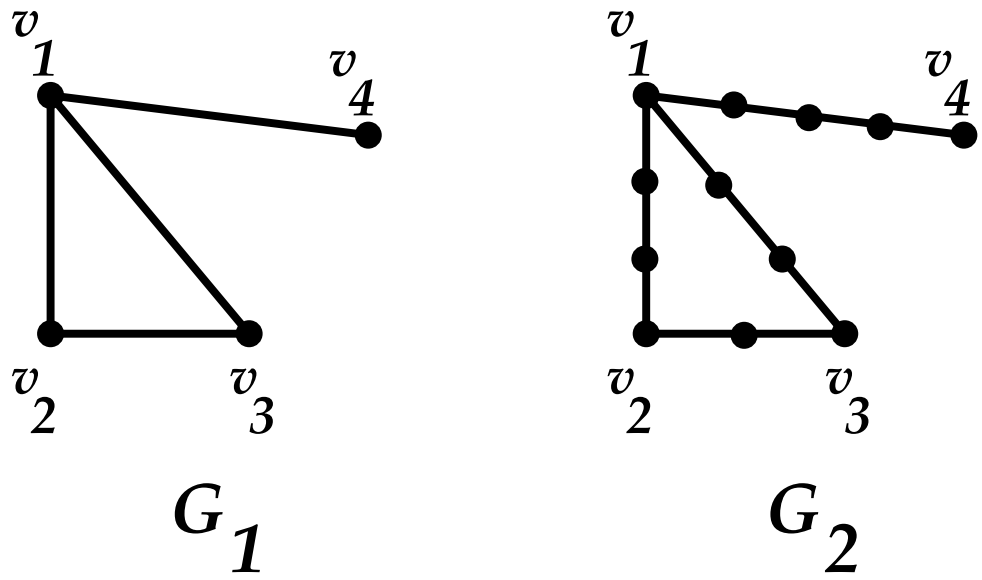


$\#$ *κορυφών* $= 7$,
$\#$ *δεσμών* $= 16$

Παρατήρηση: Προφανώς, αν ένα γράφημα G έχει ως υπογράφημα ένα μη επίπεδο γράφημα H, τότε το G είναι μη επίπεδο.

Εκλέπτυνση ή υποδιαίρεση γραφήματος

Εάν ένα γράφημα G_2 προκύπτει από ένα γράφημα G_1 με αντικατάσταση ενός ή περισσότερων δεσμών του G_1 από ένα μονοπάτι μεταξύ των αντίστοιχων κόμβων (δηλαδή, με την παρεμβολή νέων κόμβων), τότε το G_2 λέγεται **εκλέπτυνση** ή **υποδιαίρεση** του G_1.
Παράδειγμα :



Το G_2 είναι μια εκλέπτυνση του G_1.

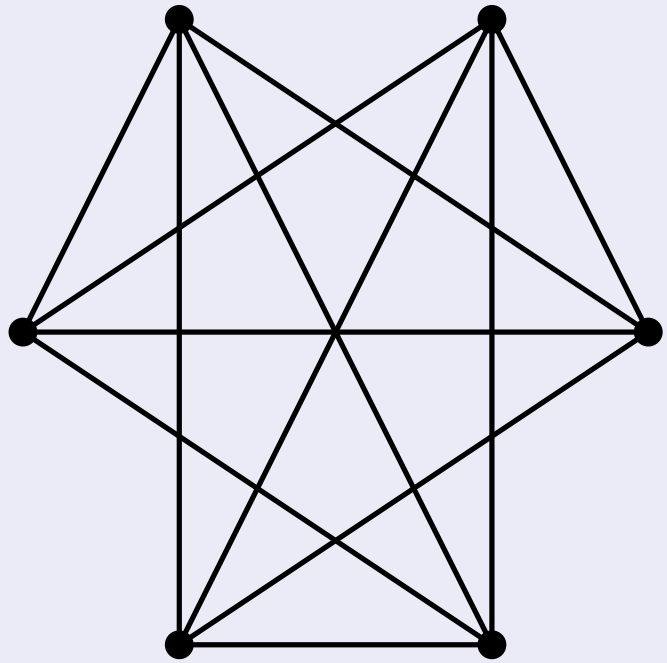
Το θεώρημα του Kuratwoski

Πρόταση 22 (Θεώρημα Kuratowski)

Ένα γράφημα είναι επίπεδο αν και μόνο αν δεν έχει ένα υπογράφημα που είναι εκλέπτυνση του K_5 ή του $K_{3,3}$.

Άσκηση 3

1. *Να εξετασθεί αν κάποιο από τα παρακάτω γραφήματα είναι επίπεδο.*
   (α) K_6, *(β)* $K_{5,2}$, *(γ)* $K_{4,3}$.
2. *Να εξετασθεί αν το παρακάτω γράφημα είναι επίπεδο.*



Παρατήρηση: Το Θεώρημα του Kuratoswki συνδέει μια τοπολογική ιδιότητα (την αναπαράσταση ενός γραφήματος στο επίπεδο χωρίς τεμνόμενους δεσμούς) με μια συνδυαστική ιδιότητα (την μη εμφάνιση ως υπογραφημάτων των εκλεπτύνσεων του K_5 και του $K_{3,3}$). Επειδή, ο έλεγχος αυτός μπορεί να γίνει σε πολυωνυμικό χρόνο έπεται ότι μπορούμε να ελέγξουμε αν ένα γράφημα είναι ή όχι επίπεδο σε πολυωνυμικό χρόνο.

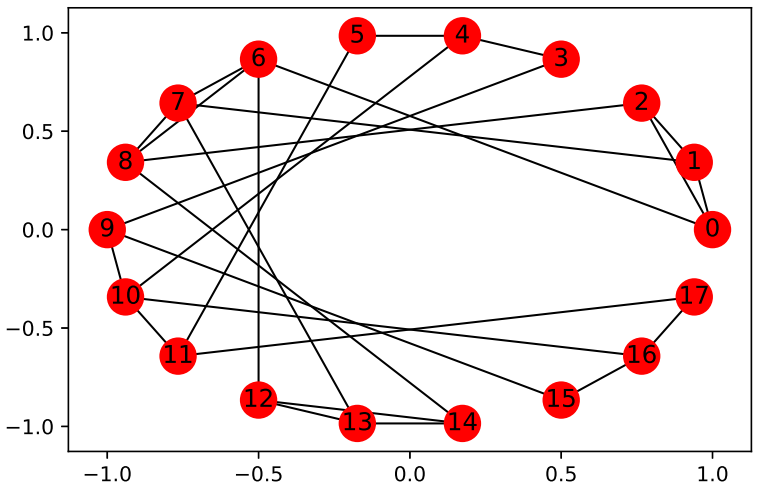
Χρησιμοποιώντας την μέθοδο `check_planarity(G)` της βιβλιοθήκης `networkx` μπορούμε αφενός να ελέγξουμε αν ένα γράφημα είναι επίπεδο και αφετέρου, εφόσον είναι επίπεδο, να το σχεδιάσουμε ως επίπεδο τοπολογικό γράφημα:

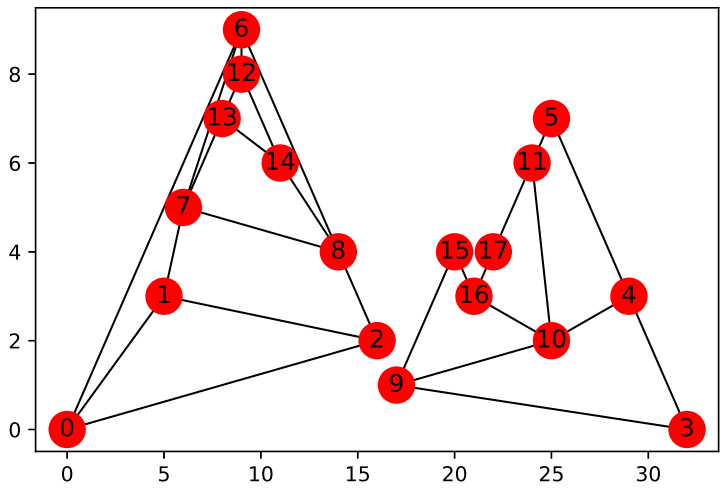
```
import networkx as nx
import matplotlib.pyplot as plt

K5 = nx.complete_graph(5)
#check_planarity(G) returns the tuple: bool is_planar,
    PlanarEmbedding
#is_planar is True iff G is planar
#PlanarEmbedding is a combinatorial description of a planar
    embedding
#to be used with nx.combinatorial_embedding_to_pos(embedding)
#to get the x,y coordinates of each vertex
if nx.check_planarity(K5)[0]:
    print("The graph is planar")
else:
    print("The graph is not planar")
```

```
#Create a planar graph G using some operations
G1 = nx.cycle_graph(3)
G2 = nx.path_graph(3)
G = nx.disjoint_union(G1,G2)
G = nx.cartesian_product(G2, G)
G = nx.disjoint_union(G, nx.Graph())
#A non-planar drawing
pos = nx.layout.shell_layout(G)
nx.draw_networkx(G,pos)
plt.show()
is_planar, embedding = nx.check_planarity(G)
#A planar drawing of G (if it exists)
if is_planar:
    pos = nx.combinatorial_embedding_to_pos(embedding)
    nx.draw_networkx(G,pos)
    plt.show()
```

Output:

```
The graph is not planar
```





Αριθμός διασταυρώσεων

Σχετικός με το πρόβλημα σύνδεσης είναι και ο παρακάτω ορισμός :
Αριθμός διασταυρώσεων (crossing number) $cr(G)$ ενός γραφήματος G είναι ο ελάχιστος αριθμός διασταυρώσεων των δεσμών του G ανά δύο, όταν το G ζωγραφιστεί στο επίπεδο.
Προφανώς $cr(G) = 0$ αν και μόνο αν το G είναι επίπεδο γράφημα. Έχει αποδειχθεί ότι ο αριθμός διασταυρώσεων ενός πλήρους γραφήματος K_n ικανοποιεί την ανισότητα :

$$cr(K_n) \leq \frac{1}{4}\lfloor\frac{n}{2}\rfloor\lfloor\frac{n-1}{2}\rfloor\lfloor\frac{n-2}{2}\rfloor\lfloor\frac{n-3}{2}\rfloor$$

(και μάλιστα, για $n \leq 12$, ισχύει το ‘‘=’’).
Επίσης, έχει αποδειχθεί ότι

$$cr(K_{m,n}) \leq \lfloor\frac{m}{2}\rfloor\lfloor\frac{m-1}{2}\rfloor\lfloor\frac{n}{2}\rfloor\lfloor\frac{n-1}{2}\rfloor$$

(και μάλιστα, όταν $m \leq n$ και είτε $m \leq 6$ είτε $m = 7$ και $n \leq 10$, ισχύει το ‘‘=’’).

Πρόταση 23

Έστω $G = (V, E)$ ένα συνεκτικό γράφημα δεσμών τότε

$$cr(G) \geq |E(G)| - 3|V(G)| + 6$$

Απόδειξη. Διακρίνουμε δύο περιπτώσεις:
Αν το G είναι επίπεδο, τότε $cr(G) = 0$ και $|E(G)| - 3|V(G)| + 6 \leq 0$, άρα η ανισότητα ισχύει.
Αν το G δεν είναι επίπεδο, τότε κάθε αναπαράσταση του στο επίπεδο θα περιέχει τουλάχιστον ένα σημείο διασταύρωσης. Έστω μια αναπαράσταση του G με $cr(G) = c$ σημεία διασταύρωσης.
Αν στο γράφημα G προσθέσουμε c επιπλέον κορυφές στα σημεία διασταύρωσης των δεσμών του, τότε θα προκύψει ένα γράφημα G' το οποίο είναι επίπεδο και έχει $|V(G)| + c$ κορυφές και $|E(G)| + 2c$ δεσμούς.
Επομένως, από το Πόρισμα 3 για το γράφημα G' θα ισχύει ότι

$$|E(G')| \leq 3|V(G')| - 6 \Leftrightarrow$$

$$|E(G)| + 2c \leq 3(|V(G)| + c) - 6 \Leftrightarrow$$

$$|E(G)| - 3|V(G)| + 6 \leq c$$