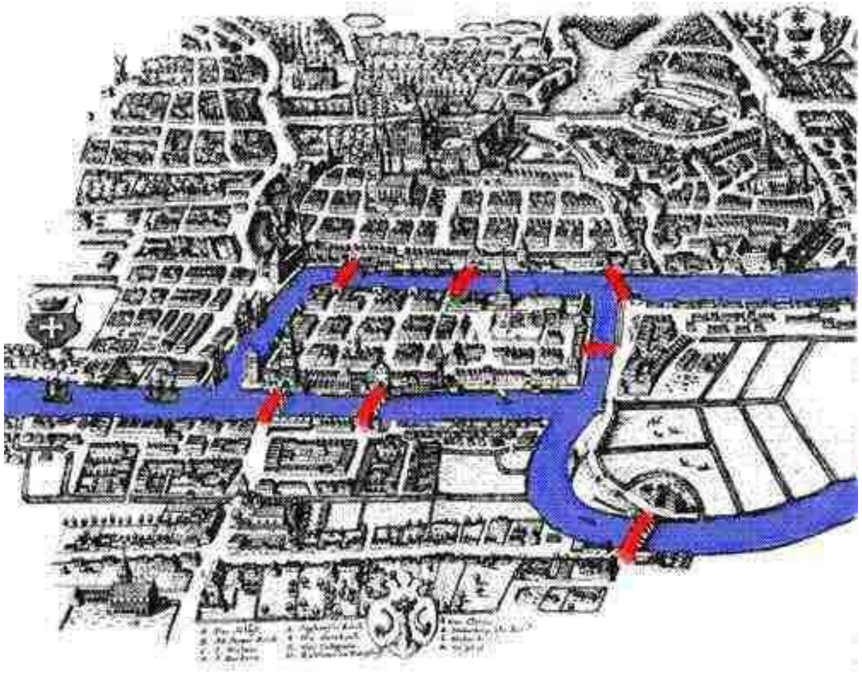
# Grafuri euleriene

# Istoric. Aplicații

din cursul 1

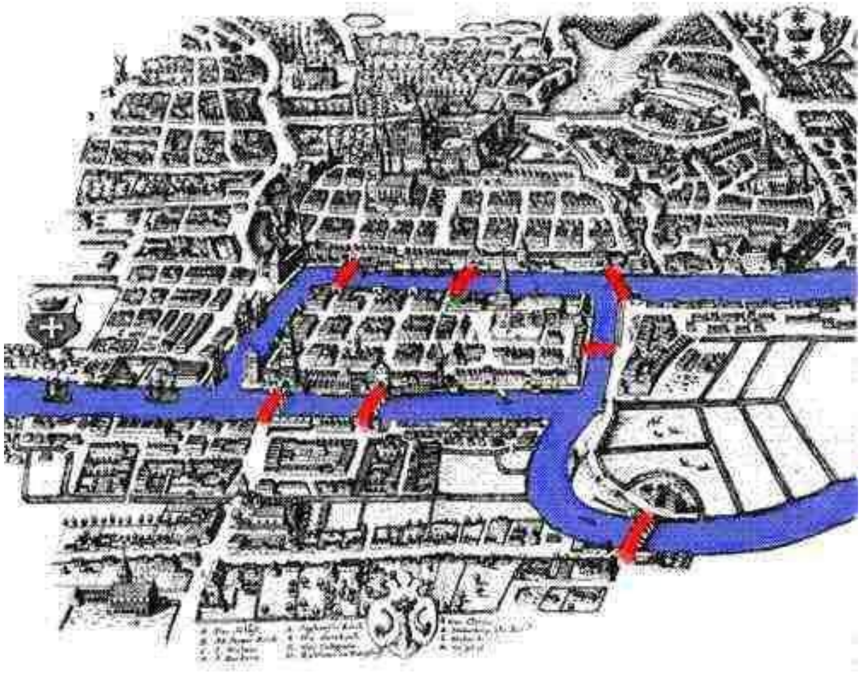
# Problema celor 7 poduri din Königsberg



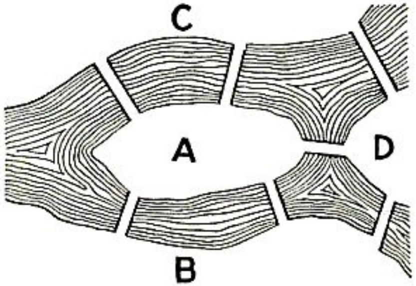
https://www.maa.org/book/export/html/116597

**Este posibil ca un om să facă o plimbare în care să treacă pe toate cele 7 poduri, o singură dată?**

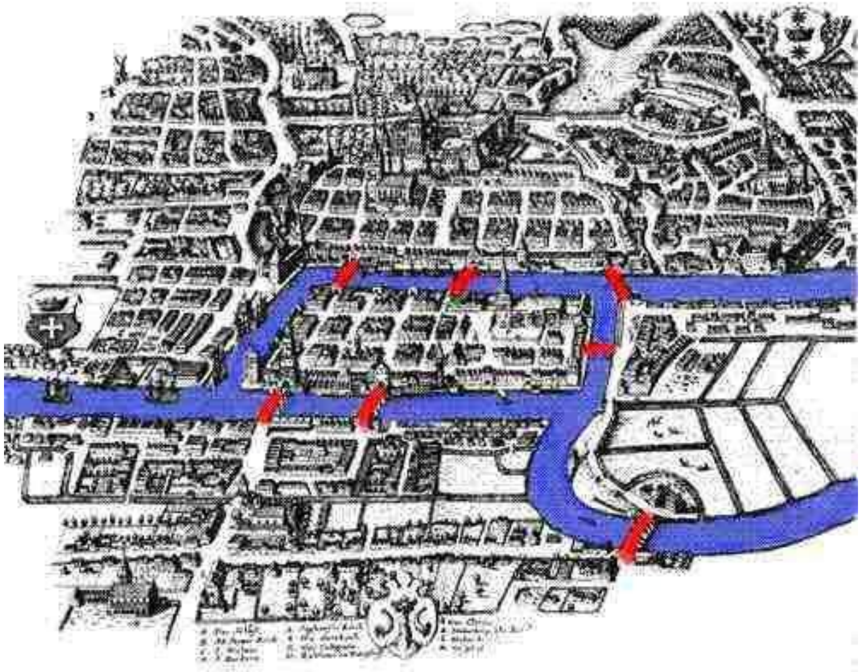
# Problema celor 7 poduri din Königsberg



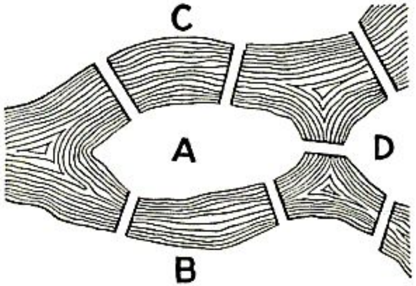
**Modelare:**

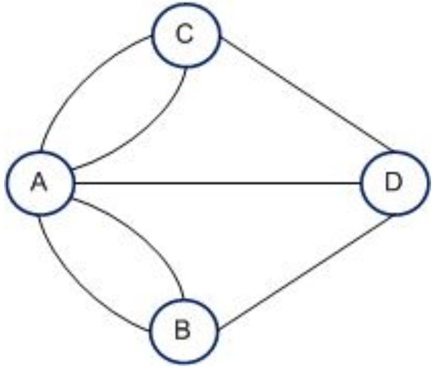


# Problema celor 7 poduri din Königsberg

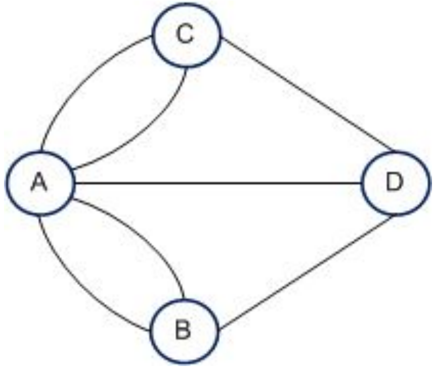


**Modelare:**

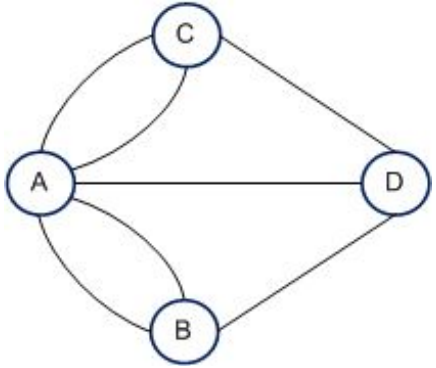


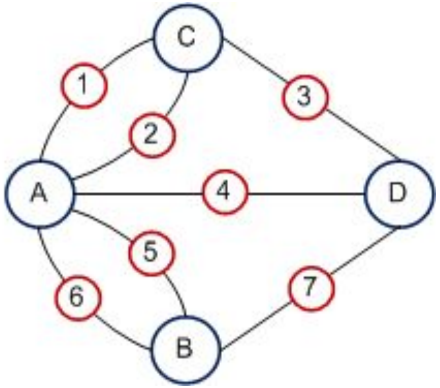



# Problema celor 7 poduri din Königsberg



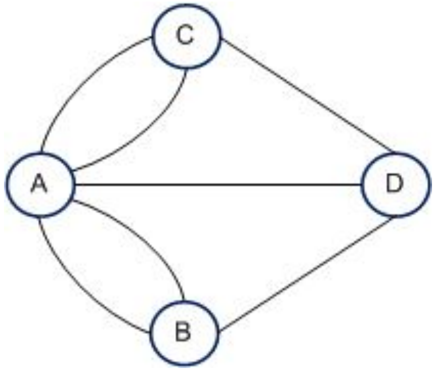
# Problema celor 7 poduri din Königsberg

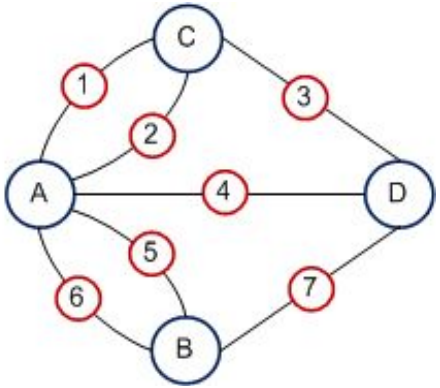



graf simplu

# Problema celor 7 poduri din Königsberg




- **1736 - Leonhard Euler**

  ***Solutio problematis ad geometriam situs pertinentis***

**Ciclu eulerian - traseu închis care trece o singură dată prin toate muchiile**

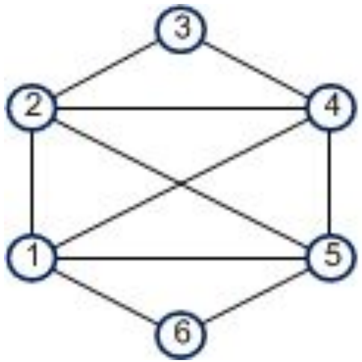
**Graf eulerian**

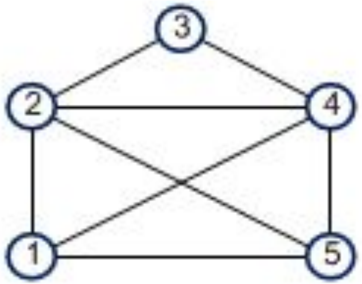
# Problema celor 7 poduri din Königsberg

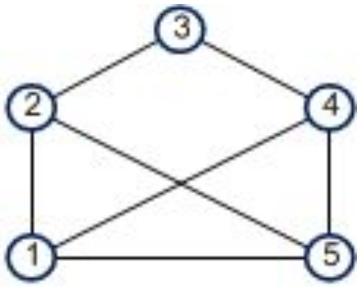
**Interpretare**

**Se poate desena diagrama printr-o curbă continuă închisă, fără a ridica pixul de pe hârtie şi fără a desena o linie de două ori (în plus: să terminăm desenul în punctul în care l-am început)?**

- tăierea unui material



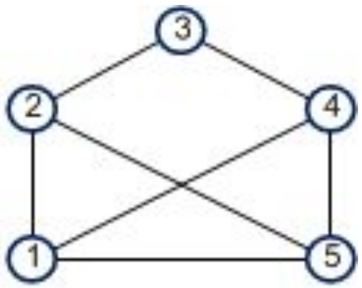



# Problema celor 7 poduri din Königsberg

**Interpretare**

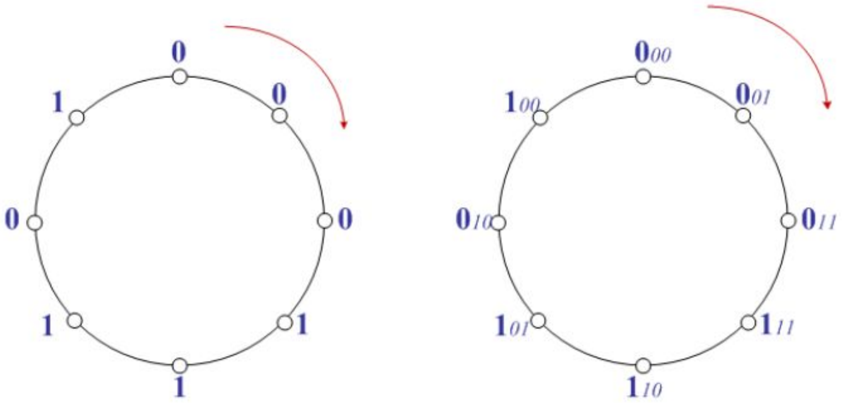
**De câte ori (minim) trebuie să ridicăm pixul de pe hârtie pentru a desena diagrama?**



# Grafuri euleriene

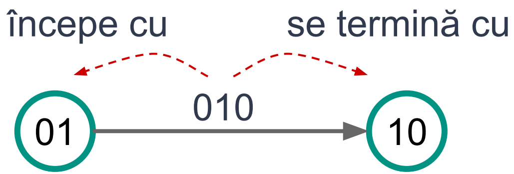
**Problema lui POSTHUMUS**

- $f(n)$ = numărul minim de cifre de 0 și 1 care se pot dispune circular a.î. între cele $f(n)$ secvențe de lungime n de cifre succesive apar toți cei $2^n$ vectori de lungime n peste {0, 1} (citite în același sens).
- Evident, $f(n) \geq 2^n$. **Are loc chiar egalitate?**

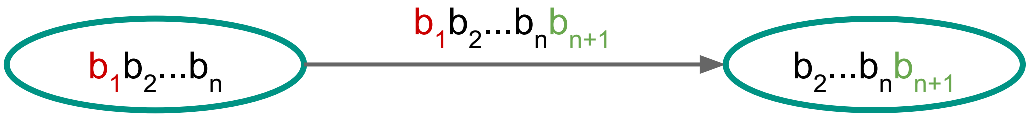


# Grafuri de Bruijn

- Etichete pe arce:

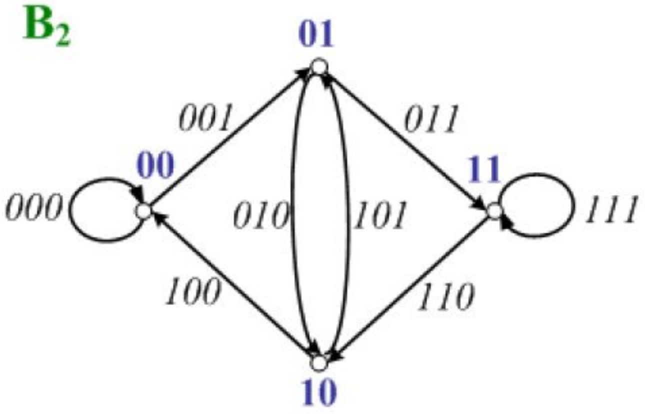


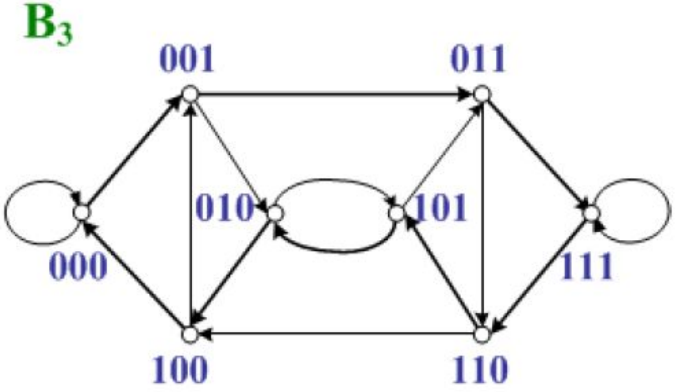

- În general:

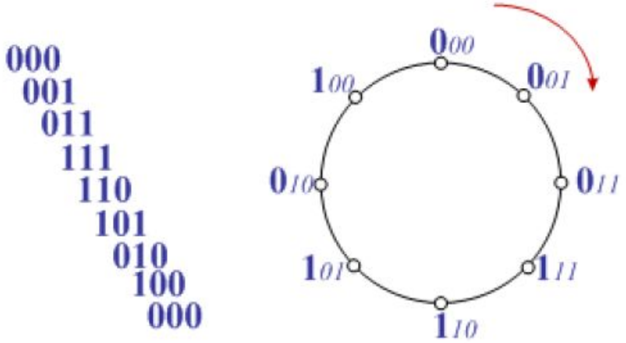



$b_i \in \{0, 1\}$

# Grafuri de Bruijn




**Soluția la problema lui POSTHUMUS pentru n = 3 ⇔** etichetele arcelor unui circuit eulerian în graful $B_2$

# Grafuri euleriene

Fie G graf neorientat.

- **Ciclu eulerian** al lui G = ciclu C în G cu

  $$E(C) = E(G)$$

- G **eulerian** = conține un **ciclu** eulerian

- **Lanț eulerian** al lui G = lanț simplu P în G cu

  $$E(P) = E(G)$$

# Grafuri euleriene

**Observație:**

Fie $P = [v_1, \ldots, v_k]$

- Dacă $v_1 \neq v_k$, atunci vârfurile interne din P au gradul în P par, iar extremitățile au gradul în P impar

- Dacă $v_1 = v_k$, atunci toate vârfurile din P au gradul în P par

# Grafuri euleriene

**Lemă:**

Fie $G = (V, E)$ un graf neorientat, conex, cu **toate vârfurile de grad par** și $E \neq \varnothing$.

Atunci, pentru orice $x \in V$, există un ciclu $C$ în $G$ cu $\mathbf{x \in V(C)}$

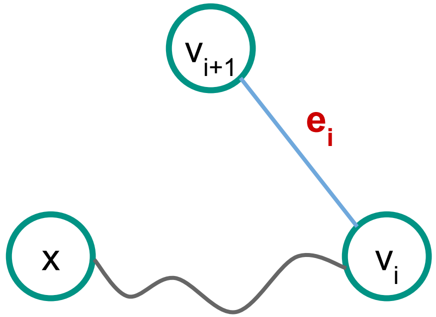
(ciclu care conține x, nu neapărat eulerian, nici neapărat elementar).

# Grafuri euleriene

**Demonstrație - Algoritm de determinare a unui ciclu care conține x:**

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă

  până când $v_i = x$

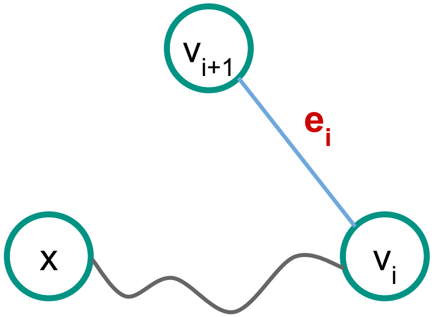


# Grafuri euleriene

**Demonstrație - Algoritm de determinare a unui ciclu care conține x:**

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
    - selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$
    - $E(C) = E(C) \cup \{e_i\}$
    - $i = i + 1$

  până când $v_i = x$

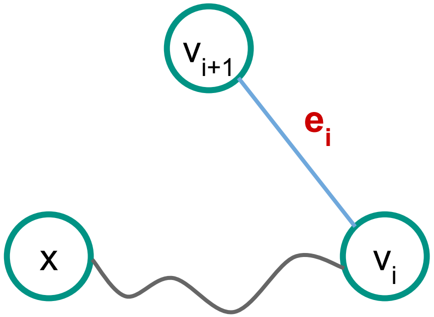


# Grafuri euleriene

**Demonstrație - Algoritm de determinare a unui ciclu care conține x:**

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$
  - $E(C) = E(C) \cup \{e_i\}$
  - $i = i + 1$

  până când $v_i = x$




**Algoritmul este corect deoarece:**

# Grafuri euleriene

**Demonstrație - Algoritm de determinare a unui ciclu care conține x:**

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$
  - $E(C) = E(C) \cup \{e_i\}$
  - $i = i + 1$

  până când $v_i = x$

**Dacă $v_i \neq x$, atunci $d_C(v_i)$ este impar. Din ipoteză, $d_G(v_i)$ este par, deci $d_{G-E(C)}(v_i)$ este impar deci $d_{G-E(C)}(v_i) > 0$**

**$\Rightarrow$ muchia $e_i$ există**

# Grafuri euleriene

**Demonstrație - Algoritm de determinare a unui ciclu care conține x:**

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$
  - $E(C) = E(C) \cup \{e_i\}$
  - $i = i + 1$

  până când $v_i = x$

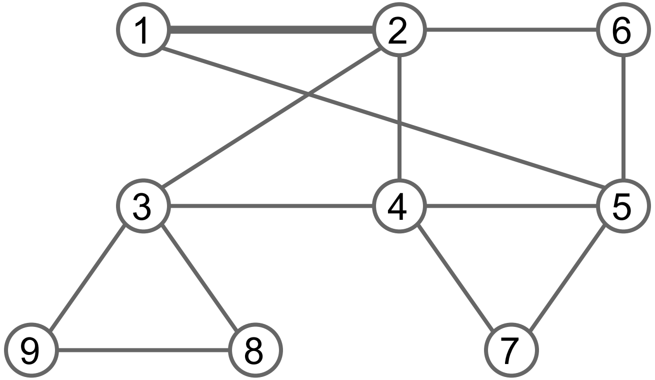
**Dacă $v_i \neq x$, atunci $d_C(v_i)$ este impar. Din ipoteză, $d_G(v_i)$ este par, deci $d_{G-E(C)}(v_i) > 0$**

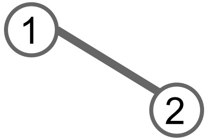
**$\Rightarrow$ muchia $e_i$ există**

**$|E(G)| < \infty$, deci algoritmul se termină ($v_i$ ajunge egal cu x)**

# Grafuri euleriene

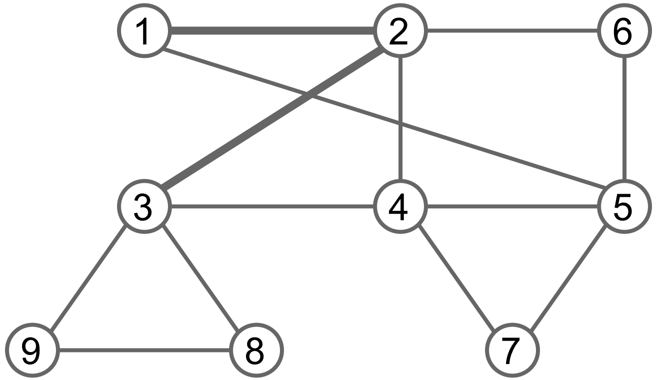
**x = 1**

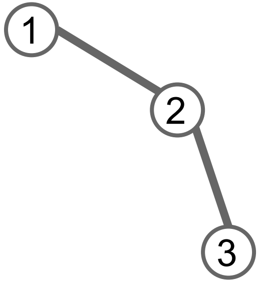


# Grafuri euleriene

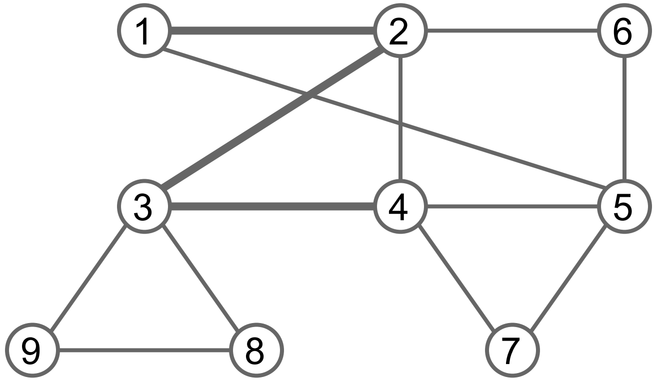
**x = 1**

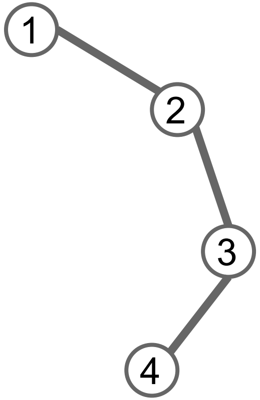


# Grafuri euleriene

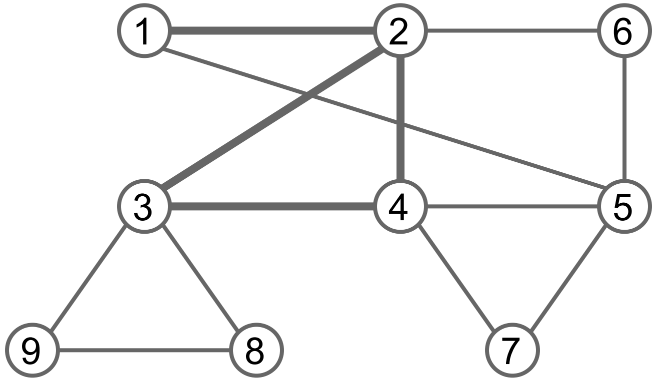
**x = 1**

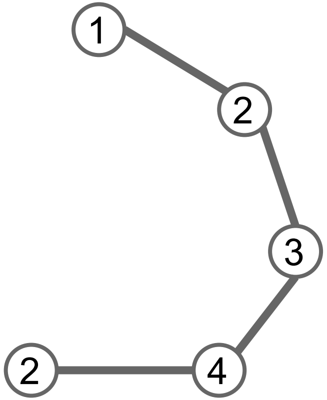


# Grafuri euleriene

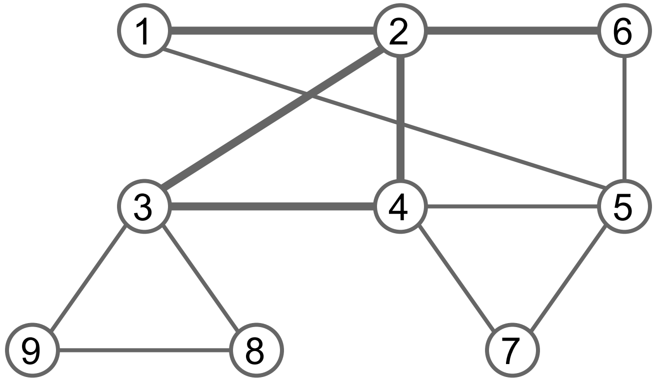
**x = 1**

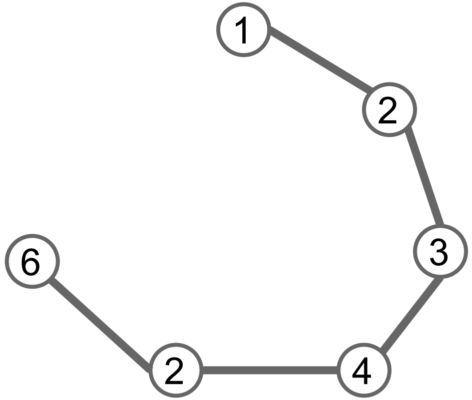


# Grafuri euleriene

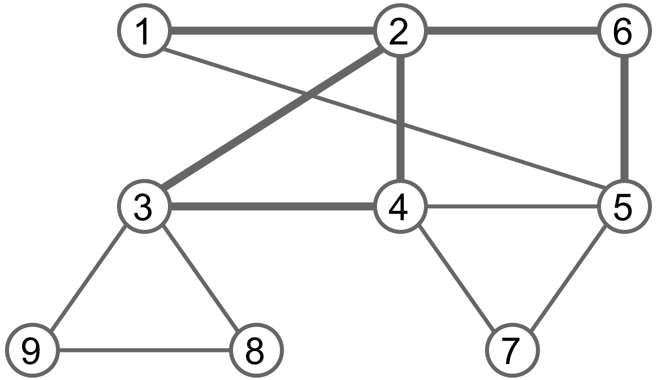
**x = 1**

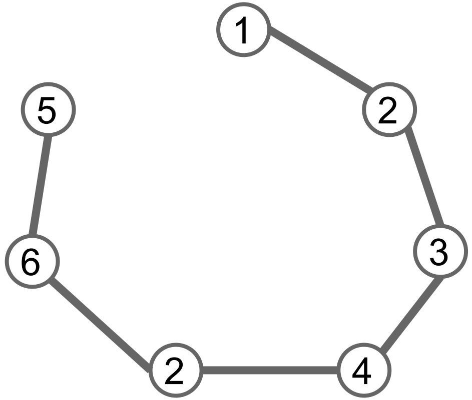


# Grafuri euleriene

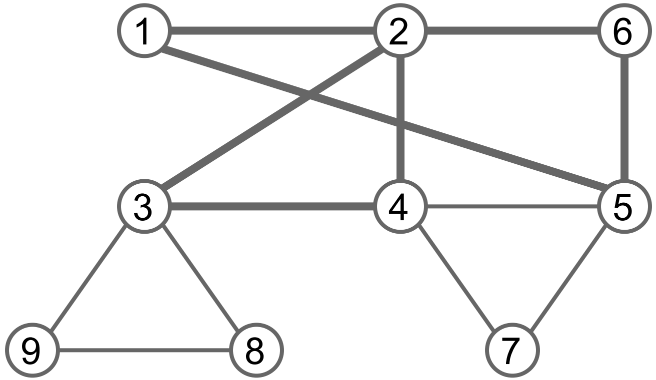
**x = 1**

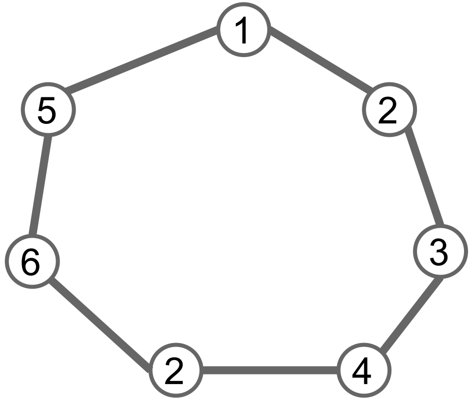


# Grafuri euleriene

**x = 1**



# Grafuri euleriene

**Teorema lui Euler**

Fie $G = (V, E)$ un (**multi**)**graf** neorientat, conex, cu $E \neq \emptyset$.

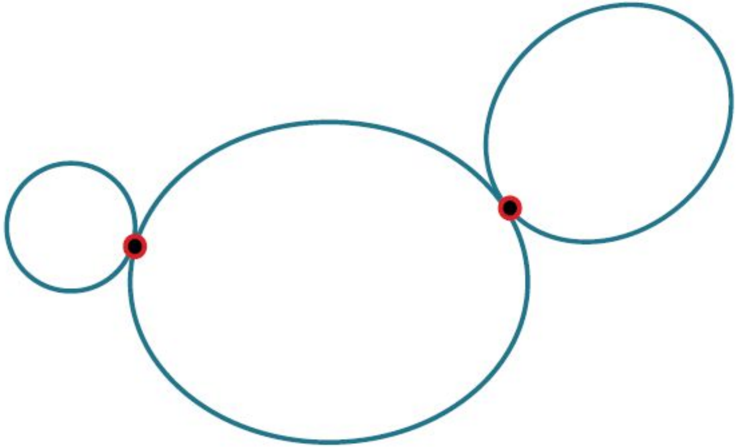
Atunci

$G$ este eulerian $\Leftrightarrow$ **orice vârf din G are grad par**

# Algoritmul lui Hierholzer

**Determinarea unui ciclu eulerian într-un graf conex** (sau un graf conex + vârfuri izolate) **cu toate vârfurile de grad par**.

- bazat pe ideea demonstrației teoremei lui Euler - **fuziune de cicluri (succesiv)**



# Algoritmul lui Hierholzer

- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)

# Algoritmul lui Hierholzer

- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)
- **Pasul 1**
  - `alege v ∈ V arbitrar`
  - `construiește C un ciclu în G care începe cu v (cu algoritmul din Lemă)`

# Algoritmul lui Hierholzer

- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)
- **Pasul 1**
  - alege $v \in V$ arbitrar
  - construiește C un ciclu în G care începe cu v (cu algoritmul din Lemă)
- **cât timp |E(C)| < |E(G)| execută**
  - selectează $v \in V(C)$ cu $d_{G-E(C)}(v) > 0$ (în care sunt indicate muchii care nu aparțin lui C)

# Algoritmul lui Hierholzer

- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)
- **Pasul 1**
  - `alege v ∈ V arbitrar`
  - `construiește C un ciclu în G care începe cu v (cu algoritmul din Lemă)`
- **cât timp |E(C)| < |E(G)| execută**
  - selectează v ∈ V(C) cu $d_{G-E(C)}(v) > 0$ (în care sunt indicate muchii care nu aparțin lui C)
  - `construiește C' un ciclu în G - E(C) care începe cu v`

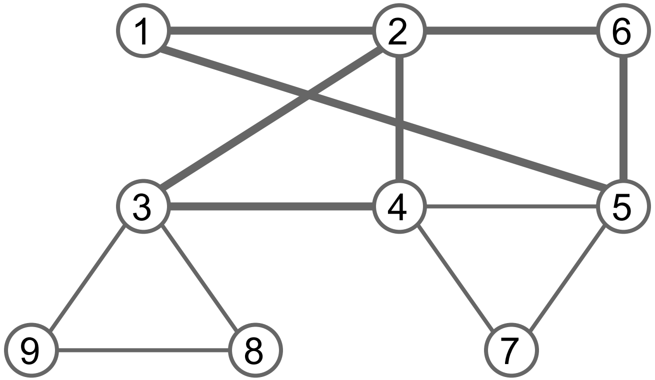
# Algoritmul lui Hierholzer

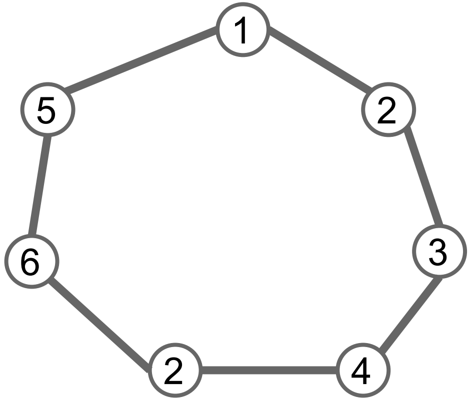
- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)
- **Pasul 1**
  - alege $v \in V$ arbitrar
  - construiește C un ciclu în G care începe cu v (cu algoritmul din Lemă)
- **cât timp |E(C)| < |E(G)| execută**
  - selectează $v \in V(C)$ cu $d_{G-E(C)}(v) > 0$ (în care sunt indicate muchii care nu aparțin lui C)
  - construiește C’ un ciclu în G - E(C) care începe cu v
  - C = ciclul obținut prin fuziunea ciclurilor C și C’ în v

# Algoritmul lui Hierholzer

- **Pasul 0** - verificare condiții (conex + vf. izolate, grade pare)
- **Pasul 1**
  - alege $v \in V$ arbitrar
  - construiește C un ciclu în G care începe cu v (cu algoritmul din Lemă)
- **cât timp |E(C)| < |E(G)| execută**
  - selectează $v \in V(C)$ cu $d_{G-E(C)}(v) > 0$ (în care sunt indicate muchii care nu aparțin lui C)
  - construiește C’ un ciclu în G - E(C) care începe cu v
  - C = ciclul obținut prin fuziunea ciclurilor C și C’ în v
- **scrie C**

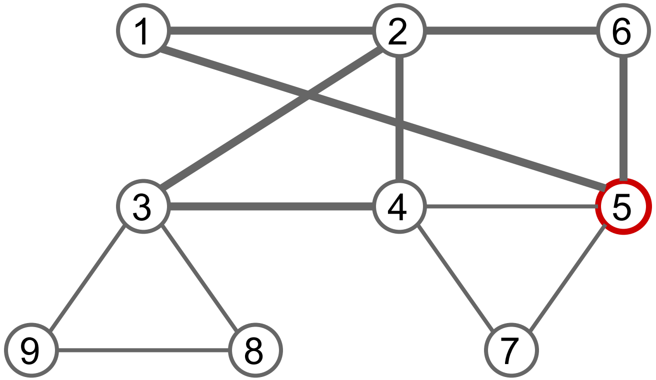
# Algoritmul lui Hierholzer



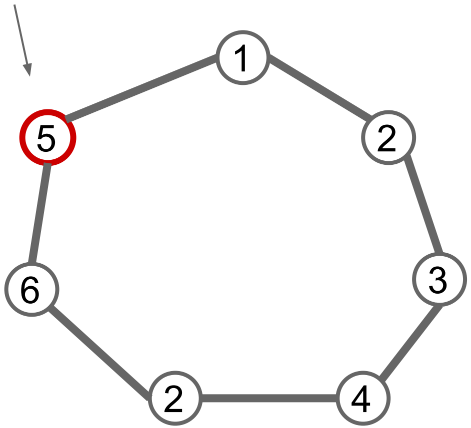




C = [1, 2, 3, 4, 2, 6, 5, 1]

# Algoritmul lui Hierholzer

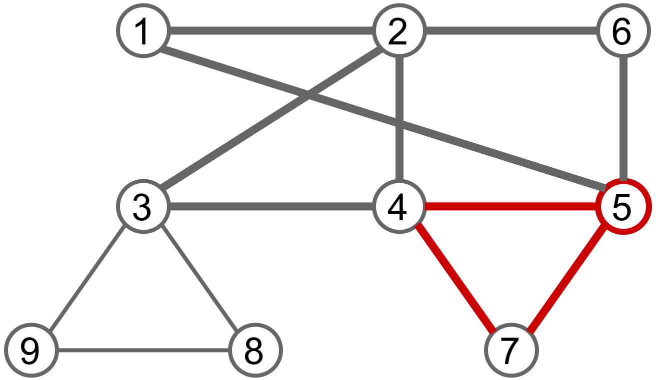



Alegem un vârf din C în care mai sunt incidente muchii, de exemplu v = 5

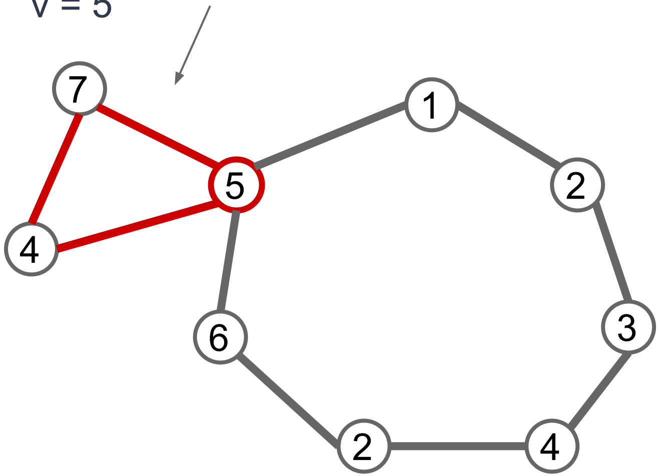



C = [1, 2, 3, 4, 2, 6, **5**, 1]

# Algoritmul lui Hierholzer



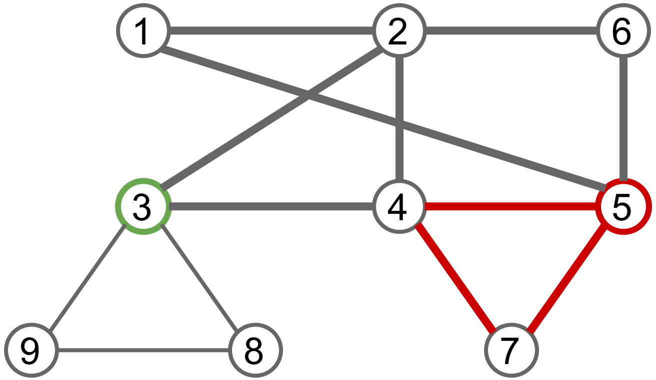

Construim un ciclu C’ cu muchiile rămase, care conține v = 5



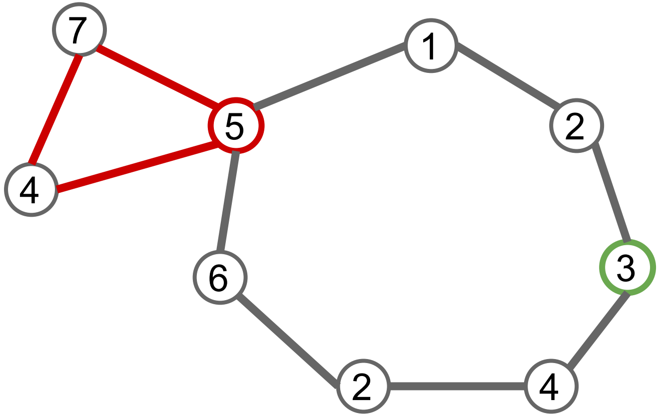

⇒ Un nou ciclu obținut prin fuziunea celor două cicluri

C = [1, 2, 3, 4, 2, 6, **5, 4, 7, 5**, 1]

# Algoritmul lui Hierholzer

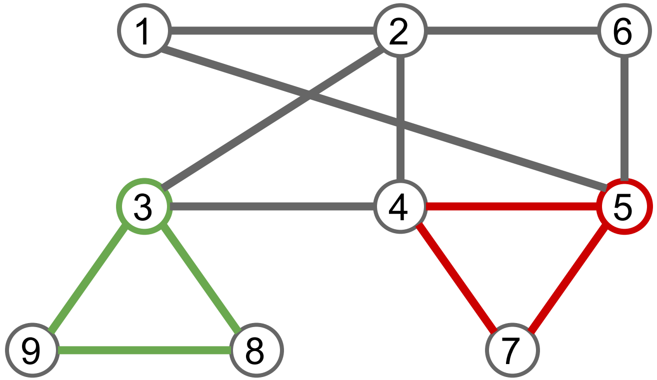



Alegem un vârf din C în care mai sunt incidente muchii, de exemplu v = 3

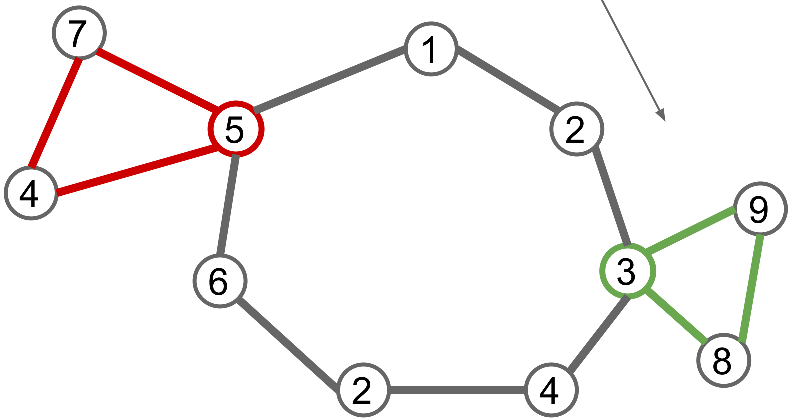



C = [1, 2, **3**, 4, 2, 6, 5, 4, 7, 5, 1]

# Algoritmul lui Hierholzer



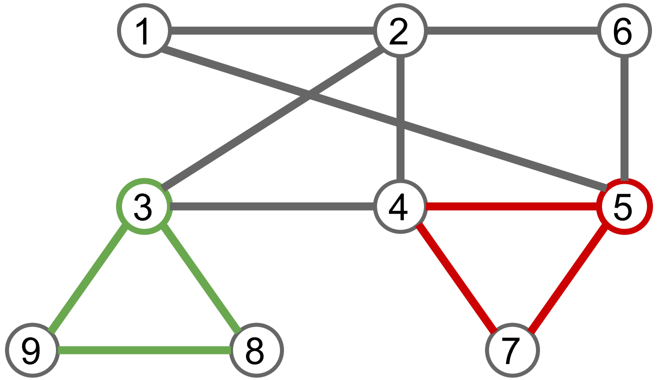

Construim un ciclu C’ cu muchiile rămase, care conține v = 3

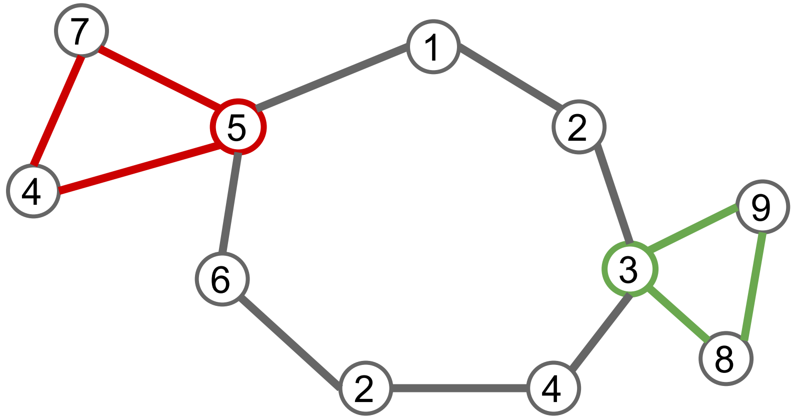



⇒ Un nou ciclu obținut prin fuziunea celor două cicluri

C = [1, 2, **3, 8, 9, 3**, 4, 2, 6, 5, 4, 7, 5, 1]

# Algoritmul lui Hierholzer




C = [1, 2, 3, 8, 9, 3, 4, 2, 6, 5, 4, 7, 5, 1]

Ciclul conține toate muchiile
⇒ este eulerian

# Algoritmul lui Hierholzer

**Complexitate - O(m)**

# Algoritmul lui Hierholzer

**Posibile implementări**

- **Varianta 1** - stiva (dfs)
- Muchiile folosite - marcate (nu neapărat șterse)

# Algoritmul lui Hierholzer

## Posibile implementări

- Varianta 2 - posibilă implementare recursivă

```
euler(nod n)
     cât timp d(v) > 0
          alege vw o muchie incidentă în v
          șterge muchia vw din G
          euler(w)
     C = C + v // adăugăm v la ciclul C
```

**Inițial**

```
C = ∅
euler(1)  // pornim construcția din vârful 1
```

# Algoritmul lui Hierholzer

## Posibile implementări

- **Varianta 2 - posibilă implementare recursivă**

```
euler(nod n)
     cât timp d(v) > 0
          alege vw o muchie incidentă în v
          șterge muchia vw din G
          euler(w)
     C = C + v // adăugăm v la ciclul C
```

**Observație** - putem alege muchiile incidente în v, de exemplu, în ordinea dată de listele de adiacență

```
cât timp d(v) > 0
          alege vw o muchie incidentă în v
          șterge muchia vw din G
          euler(w)
```

$\Rightarrow$

```
pentru vw ∈ E
     șterge muchia vw din G
     euler(w)
```

# Algoritmul lui Hierholzer

## Posibile implementări

- **Varianta 2 - posibilă implementare recursivă**

```
euler(nod n)
     cât timp d(v) > 0
          alege vw o muchie incidentă în v
          șterge muchia vw din G
          euler(w)
     C = C + v // adăugăm v la ciclul C
```

### Inițial

```
C = ∅
euler(1)  // pornim construcția din vârful 1
```

https://www.infoarena.ro/problema/ciclueuler

# Lanțuri euleriene

**Teorema lui Euler**

Fie $G = (V, E)$ un graf neorientat, conex, cu $E \neq \varnothing$.

Atunci

G are un lanț eulerian $\Leftrightarrow$ **G are cel mult două vârfuri de grad impar**

# Descompuneri euleriene în lanțuri (suplimentar)

**k-descompunere euleriană în lanțuri** a unui graf G =

o mulțime de k lanțuri simple, muchie-disjuncte

$\Delta = \{P_1, P_2, \ldots, P_k\}$

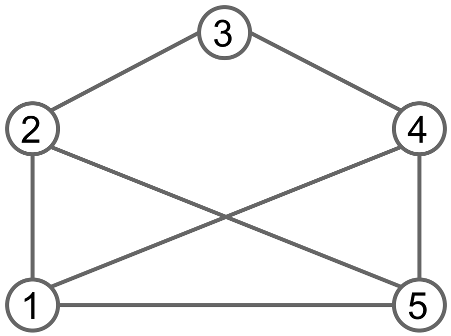
ale căror muchii induc o k-partiție a lui E(G)

$E(G) = E(P_1) \cup E(P_2) \cup \ldots \cup E(P_k)$

# Descompuneri euleriene în lanțuri (suplimentar)

**Interpretare**

De câte ori (minim) trebuie să ridicăm creionul de pe hârtie, pentru a desena diagrama?



# Descompuneri euleriene în lanțuri (suplimentar)

**Teoremă - Descompunerea euleriană**

Fie G = (V, E) un graf orientat, conex (= graful neorientat asociat este conex), cu exact **2k vârfuri de grad impar** (k > 0).

Atunci există o k-descompunere euleriană a lui G și k este cel mai mic cu această proprietate.

# Grafuri orientate euleriene

**Teorema lui Euler**

Fie $G = (V, E)$ un graf orientat, conex (= graful neorientat asociat este conex), cu $E \neq \emptyset$.

Atunci

$$G \text{ este eulerian} \iff \forall\, v \in V,\ \ d^-_G(v) = d^+_G(v)$$

# Lanțuri euleriene

**Teorema lui Euler**

Fie $G = (V, E)$ un graf orientat, conex (= graful neorientat asociat este conex), cu $E \neq \varnothing$.

Atunci

G are un drum eulerian $\Leftrightarrow$

$\forall\ v \in V,\ \ d^-_G(v) = d^+_G(v)$ SAU

$\exists\ x \in V$ cu $d^-_G(x) = d^+_G(x) - 1$,

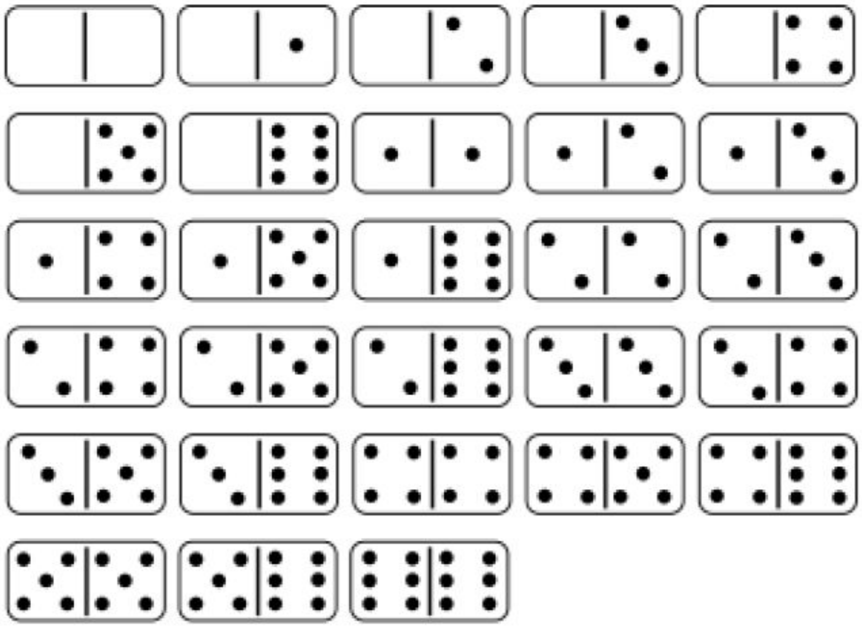
$\exists\ y \in V$ cu $d^-_G(y) = d^+_G(y) + 1$,

$\forall\ v \in V-\{x, y\}\ \ d^-_G(v) = d^+_G(v)$

# Grafuri euleriene

**Problemă - joc domino**

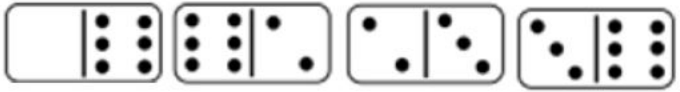
Piesă de domino - două fețe, numere 0, …, n, de obicei n = 6



# Grafuri euleriene

**Problemă - joc domino**

Șir de piese de domino - respectă regula de construcție: primul număr de pe piesa adăugată la șir = al doilea număr de pe ultima piesă din șir



# Grafuri euleriene

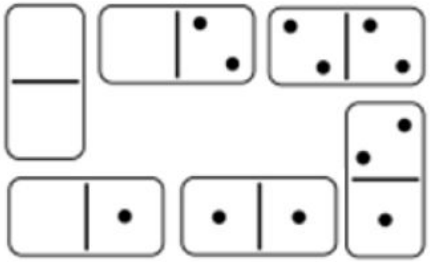
**Problemă - joc domino**

Se poate forma un șir de piese de domino care să conțină toate piesele + să se termine cu același număr cu care a început (un șir circular)?

# Grafuri euleriene

**Problemă - joc domino**

Exemplu - dacă folosim doar piese cu numere 0, …, 2, putem forma un ciclu



# Grafuri euleriene

**Problemă - joc domino**

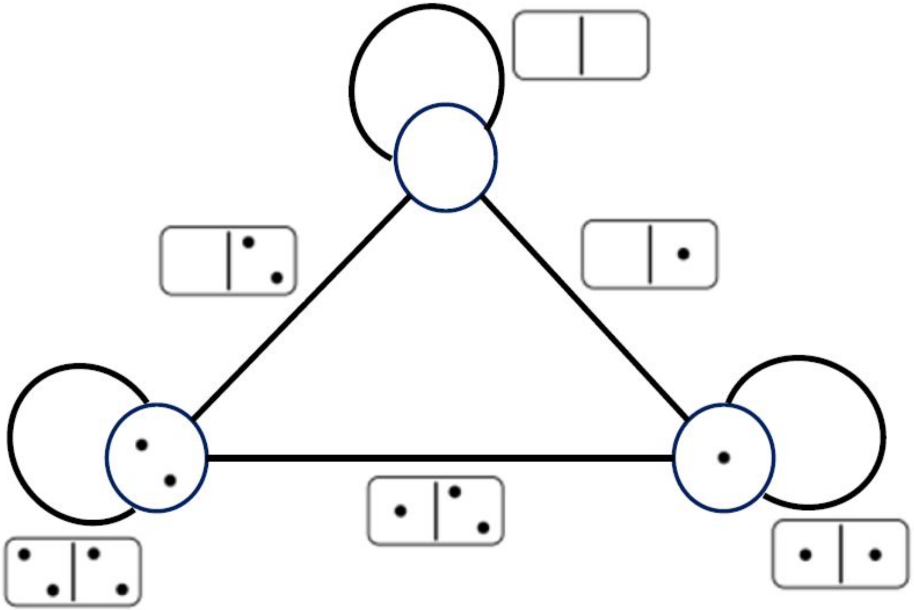
Graf asociat

- vârfuri - numerele de pe piese
- muchii - perechi de numere (piesele)
- se pot lipi doar piese asociate muchiilor adiacente

# Grafuri euleriene

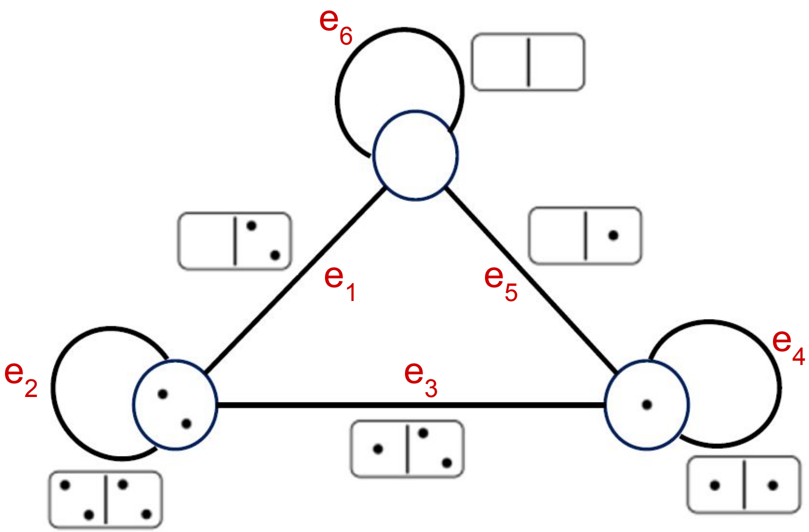
**Problemă - joc domino**

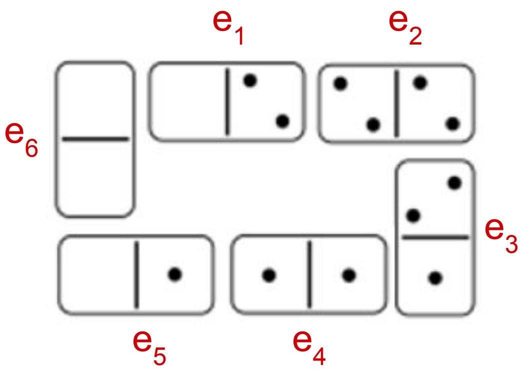
n = 2



# Grafuri euleriene

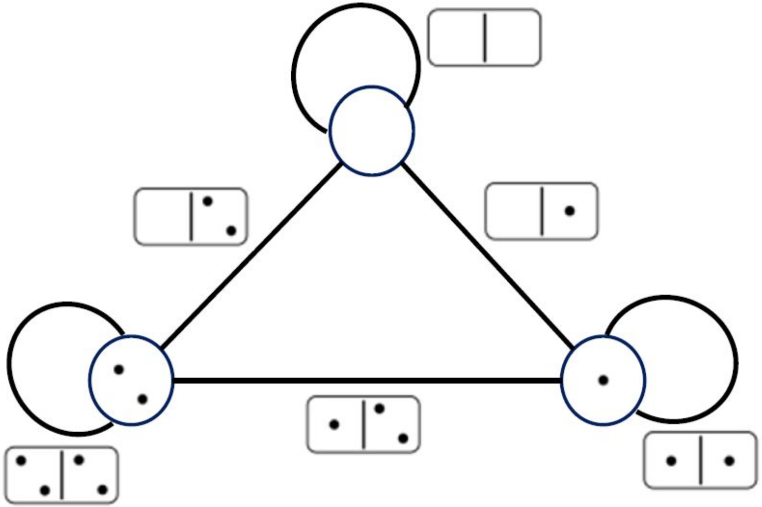
**Există ciclu de piese ⇔ există ciclu eulerian în (multi)graf**



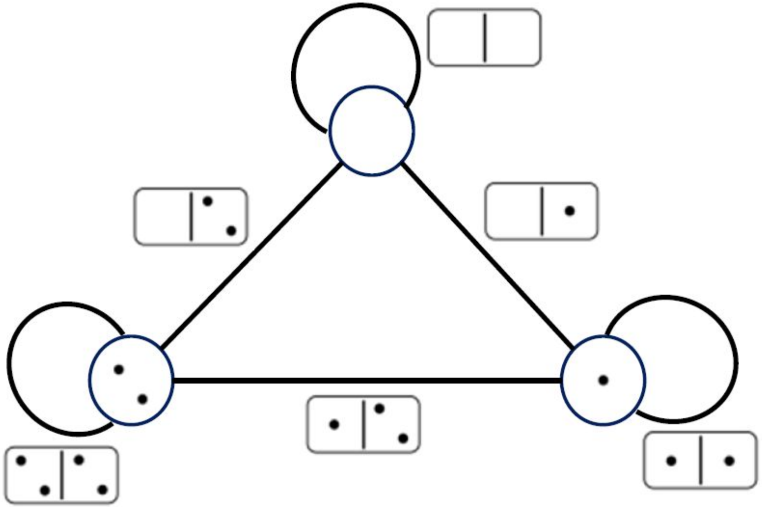
# Grafuri euleriene

**Pentru ce valori ale lui n există un ciclu eulerian în (multi)graful asociat?**



# Grafuri euleriene

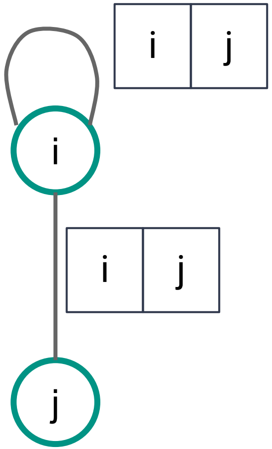
**Pentru ce valori ale lui n există un ciclu eulerian în (multi)graful asociat?**



**d(i) = ?, pentru i = 0, …, n**

# Grafuri euleriene

**Pentru ce valori ale lui n există un ciclu eulerian în (multi)graful asociat?**




**$d(i) = n + 2$**

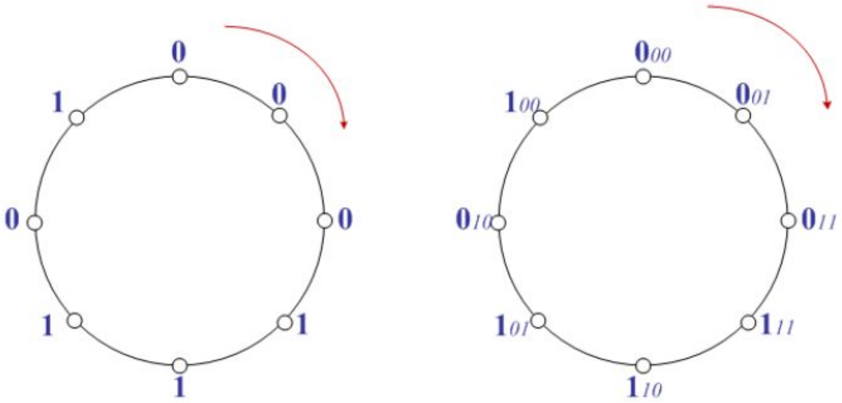
muchiile incidente în i sunt:
bucla neetichetată (i, i) și
muchiile etichetate $\{i, j\}$ cu $i \neq j$, $j \notin \{0, \dots, n\}$

**⇒ trebuie ca n să fie par**

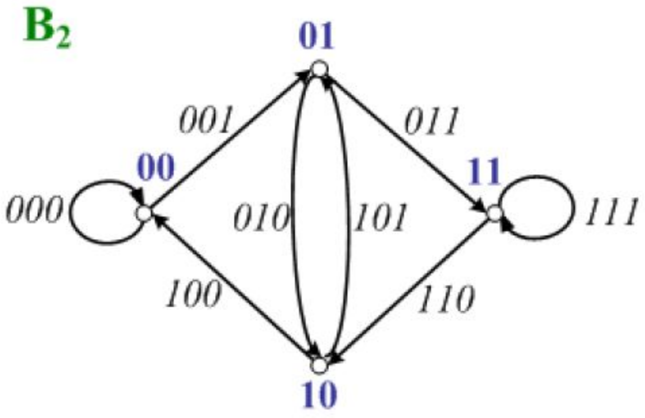
# Grafuri euleriene

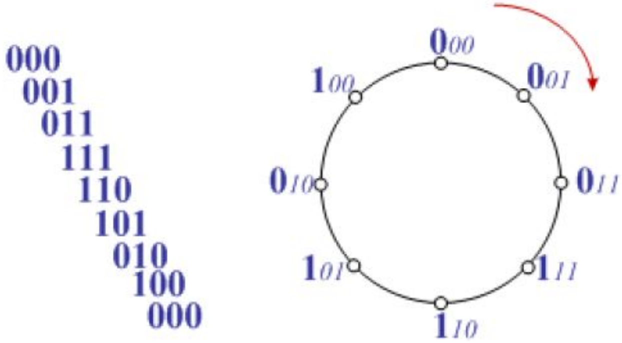
**Problema lui POSTHUMUS (Suplimentar)**

- f(n) = numărul minim de cifre de 0 și 1 care se pot dispune circular a.î. între cele f(n) secvențe de lungime n de cifre succesive apar toți cei $2^n$ vectori de lungime n peste {0, 1} (citite în același sens).
- Evident, $f(n) \geq 2^n$. **Are loc chiar egalitate?**



# Grafuri de Bruijn




**Prima cifră din etichetele arcelor unui circuit eulerian în $B_{n-1}$ - soluție pentru problema lui Posthumus**

# Grafuri de Bruijn

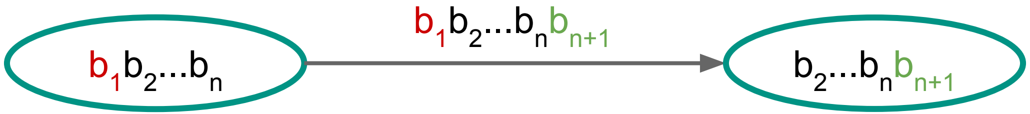
- Multigraf
- $V(B_n) = \{0, 1\}^n$ (mai general $\{0, 1, \ldots, p\}^n$)

  (sau cuvinte de lungime n peste un alfabet finit)

- $E(B_n)$ etichetate cu $\{0, 1\}^{n+1}$ ($\{0, 1, \ldots, p\}^{n+1}$)

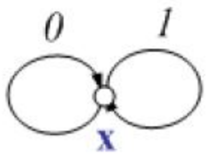
  $b_1b_2...b_nb_{n+1}$ etichetează arcul de la

  $b_1\mathbf{b_2...b_n}$ la $\mathbf{b_2...b_n}b_{n+1}$

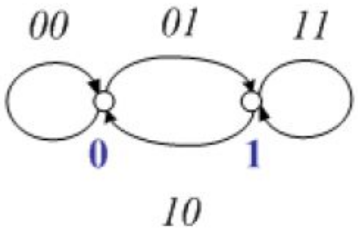


# Grafuri de Bruijn

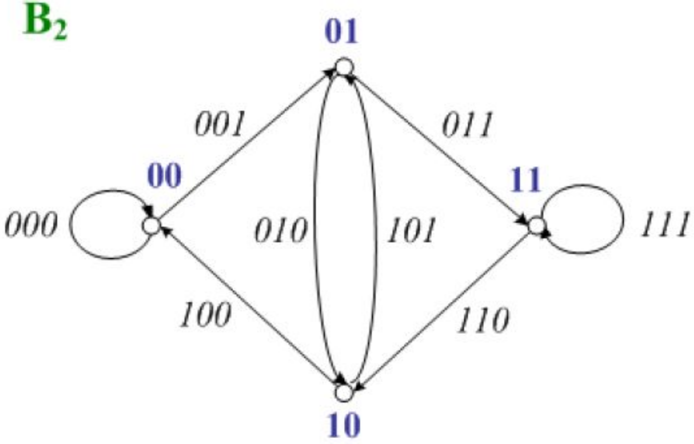
$\mathbf{B_0}$



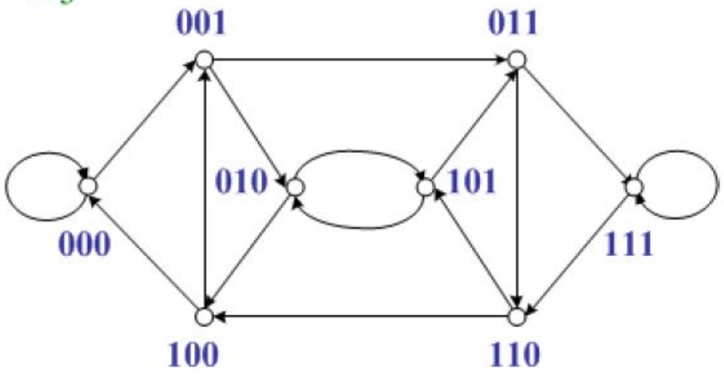

$\mathbf{B_1}$




$\mathbf{B_2}$




$\mathbf{B_3}$

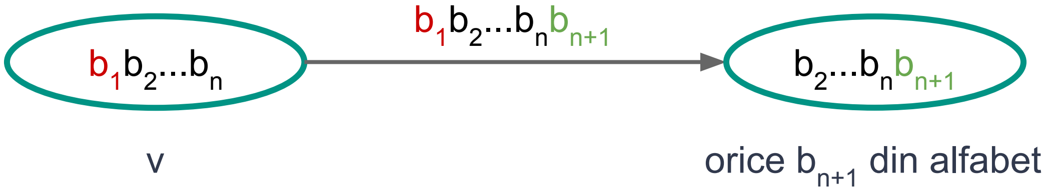


# Grafuri de Bruijn

$B_n$ este eulerian?

$d^+(v) = ?$

$d^-(v) = ?$

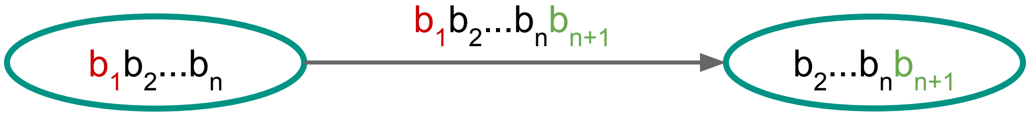


# Grafuri de Bruijn

$B_n$ este eulerian

$d^+(v) = |\{0, 1\}| = 2$ (mai general = $|\{0, 1, \ldots, p\}|$)

$d^-(v) = d^+(v)$




v

orice $b_{n+1}$ din alfabet

# Grafuri de Bruijn

- **Prima cifră din etichetele arcelor unui circuit eulerian în $B_{n-1}$ - soluție pentru problema lui Posthumus**

  **$\Rightarrow f(n) = 2^n$**

- **Observație**

  Circuit eulerian în $B_{n-1}$ $\leftrightarrow$ circuit hamiltonian în $B_n$

# Grafuri de Bruijn

**Aplicație -** genetică (*Genome Assembly*)