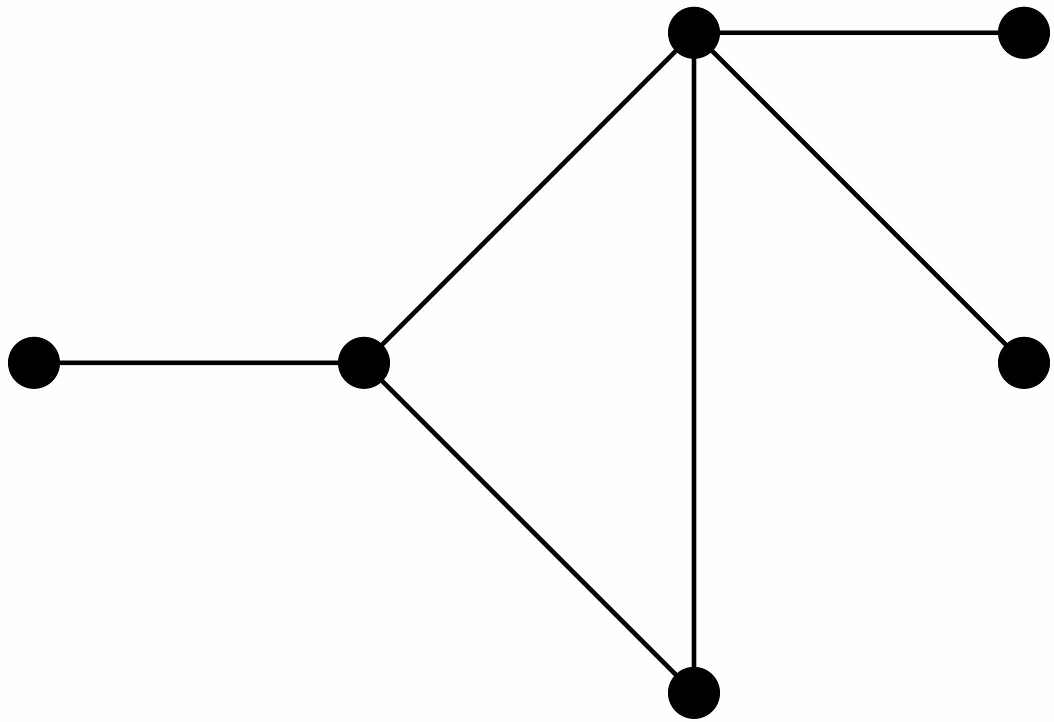
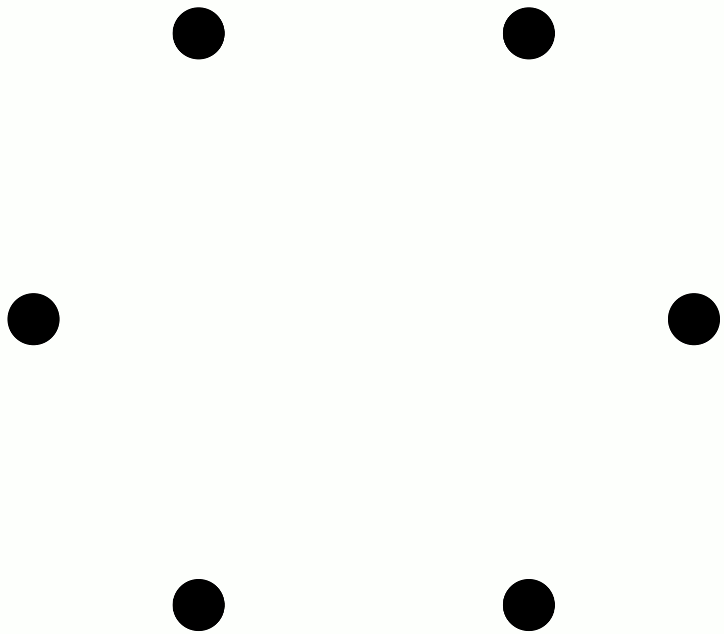
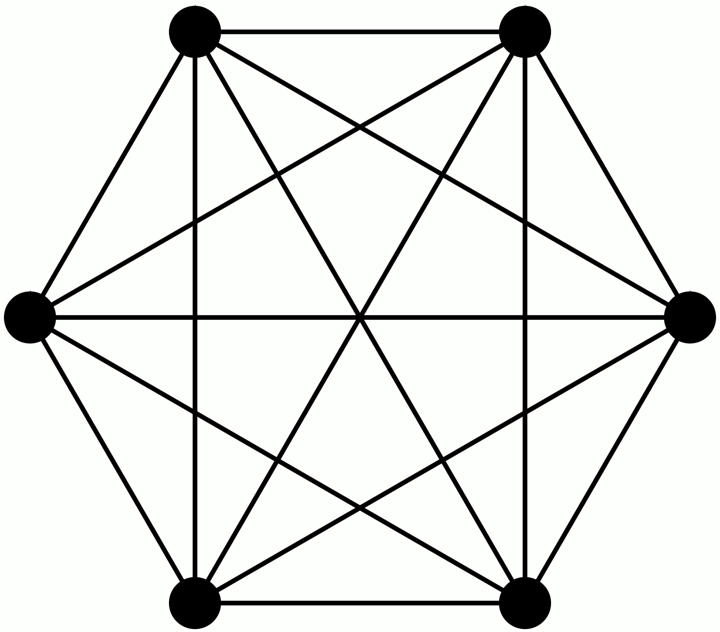


Graf prosty

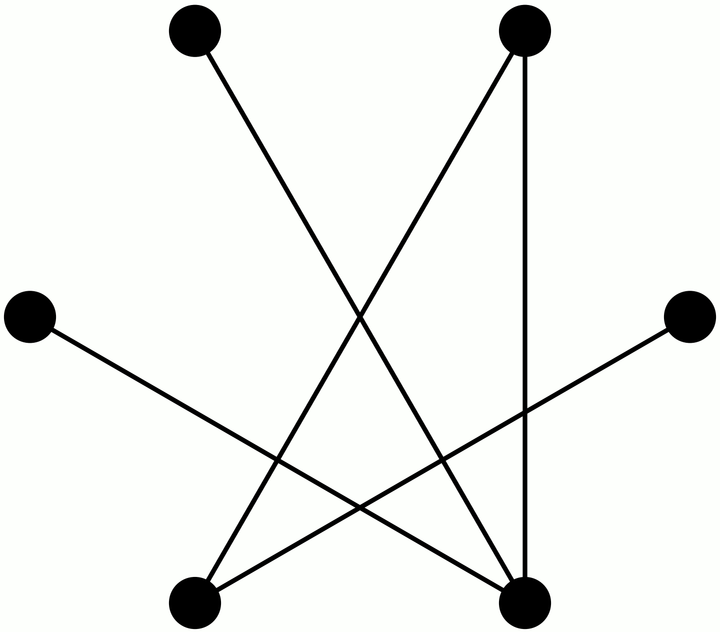
Podstawowe typy grafów: graf pusty



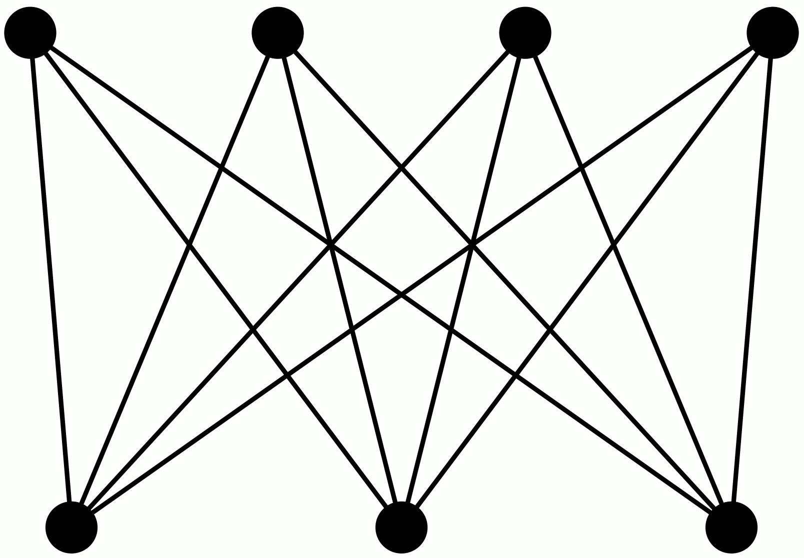
Podstawowe typy grafów: graf pełny

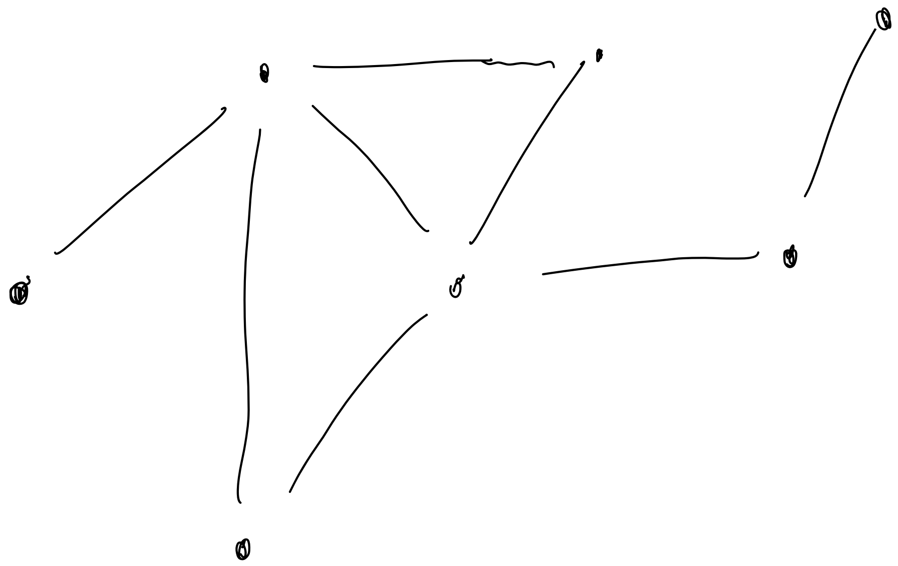


Podstawowe typy grafów: graf dwudzielny



Podstawowe typy grafów: pełny graf dwudzielny





Podstawowe definicje

- **Spacer** od v do w:

$$vv_1,\ \ v_1v_2,\ \ \ldots,\ \ v_{k-1}w.$$

Oznaczenie:

$$v \to v_1 \to v_2 \to \cdots \to v_{k-1} \to w.$$

- **Długością** spaceru jest liczba jego krawędzi.
- **Spacer zamknięty**: $v = w$.

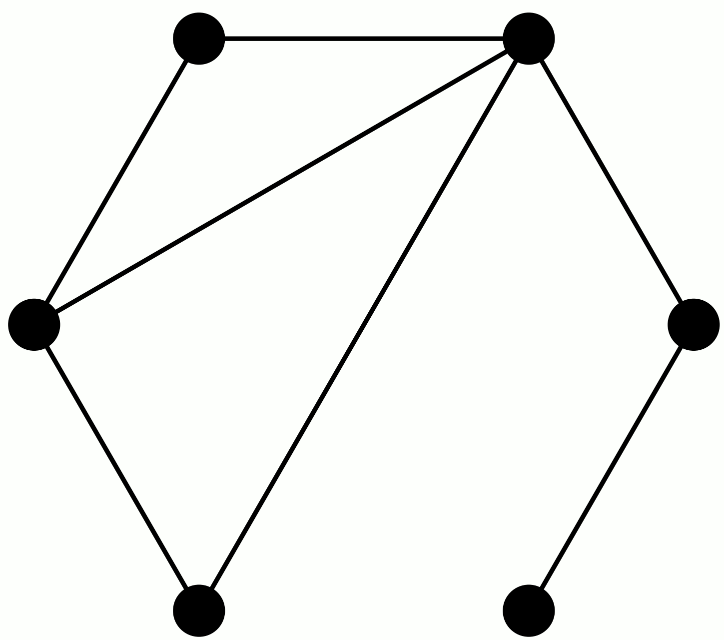
Podstawowe definicje

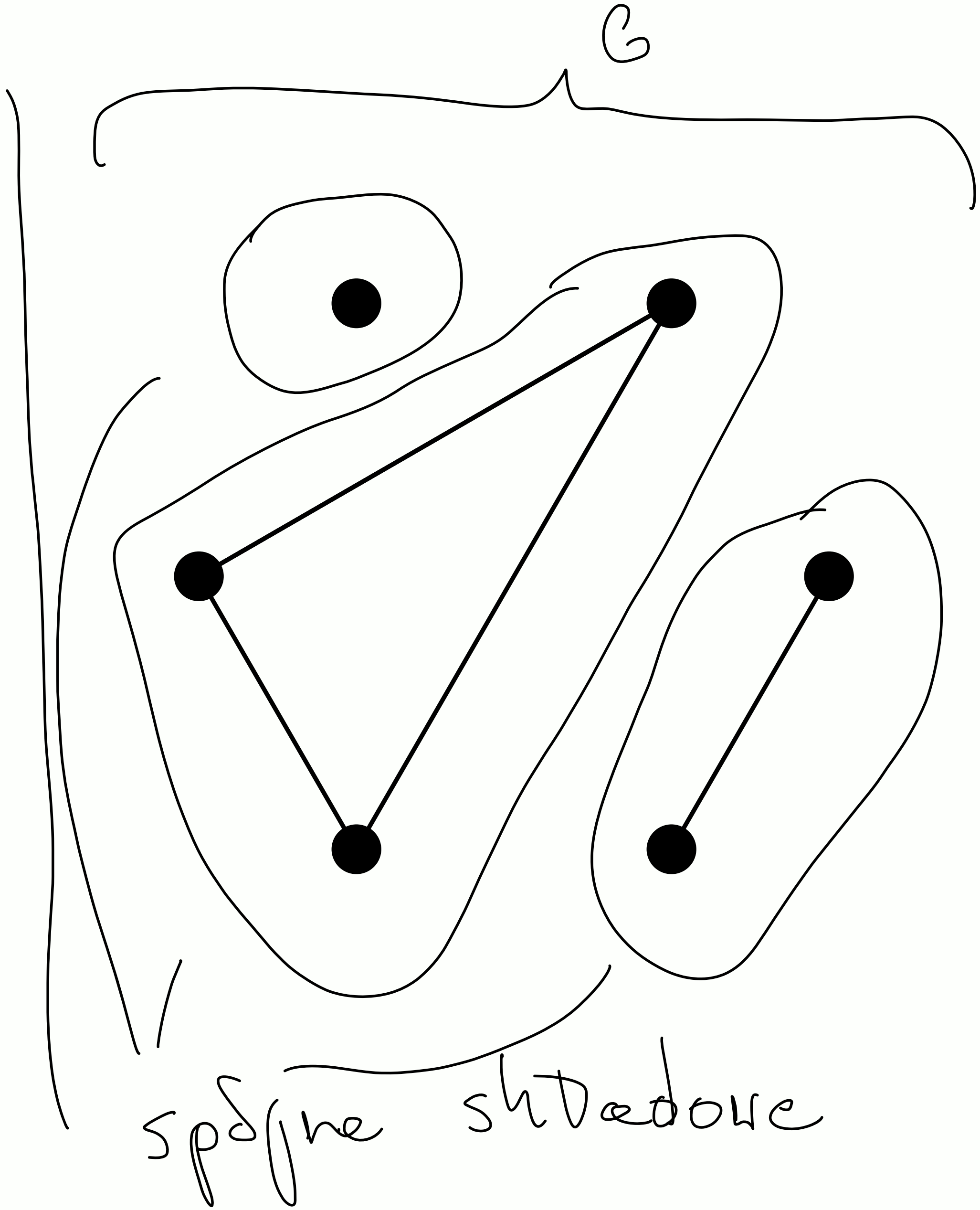
- **Droga**: spacer bez powtarzających się krawędzi.
- **Ścieżka**: spacer bez powtarzających się wierzchołków.
- **Cykl**: ścieżka długości $\geqslant 2$, w której pierwszy i ostatni wierzchołek są połączone krawędzią.

spacer < droga < ścieżka

Podstawowe definicje

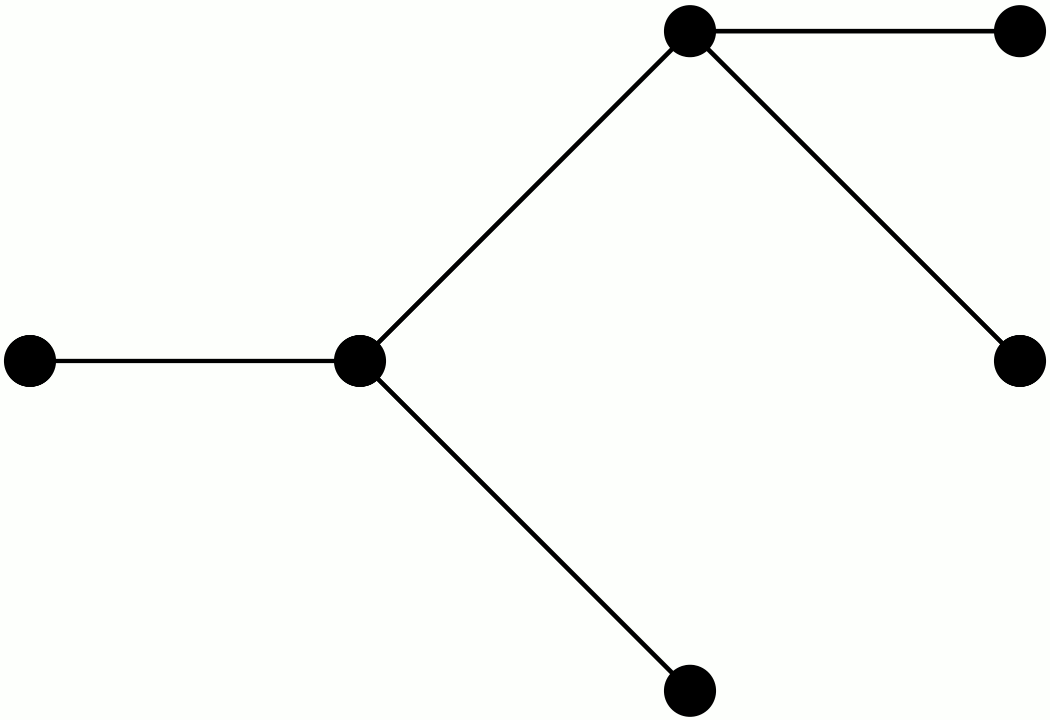
⇝ **Graf spójny**: między każdymi dwoma wierzchołkami istnieje spacer.





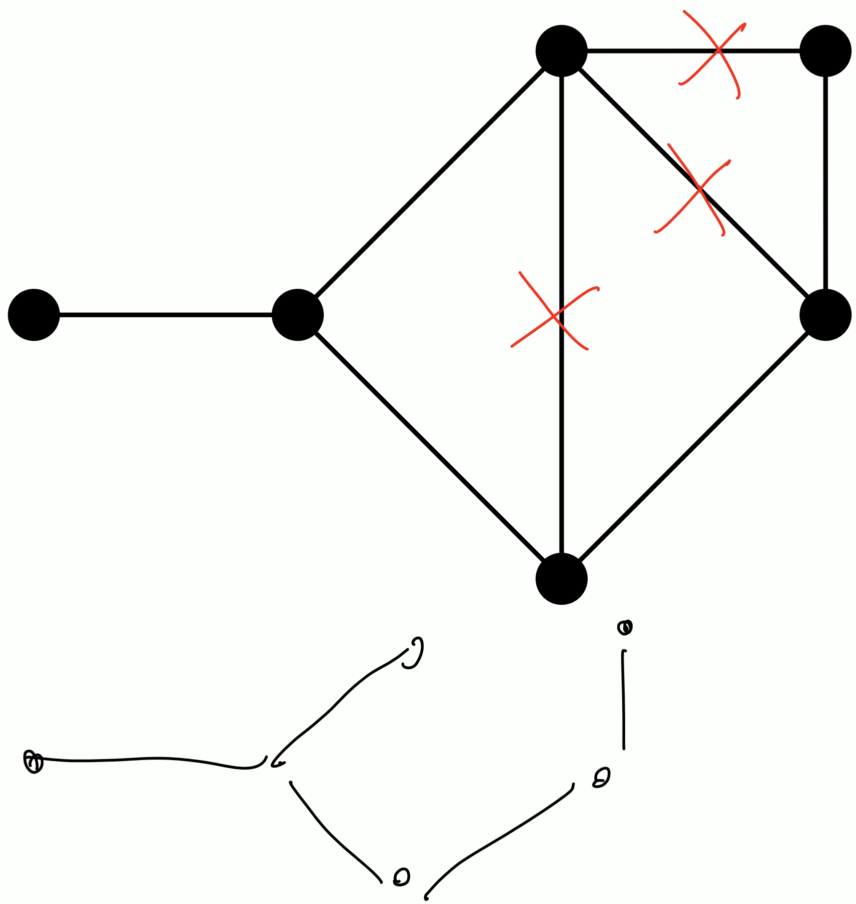
Podstawowe definicje

⇝ **Drzewo**: graf spójny bez cykli.

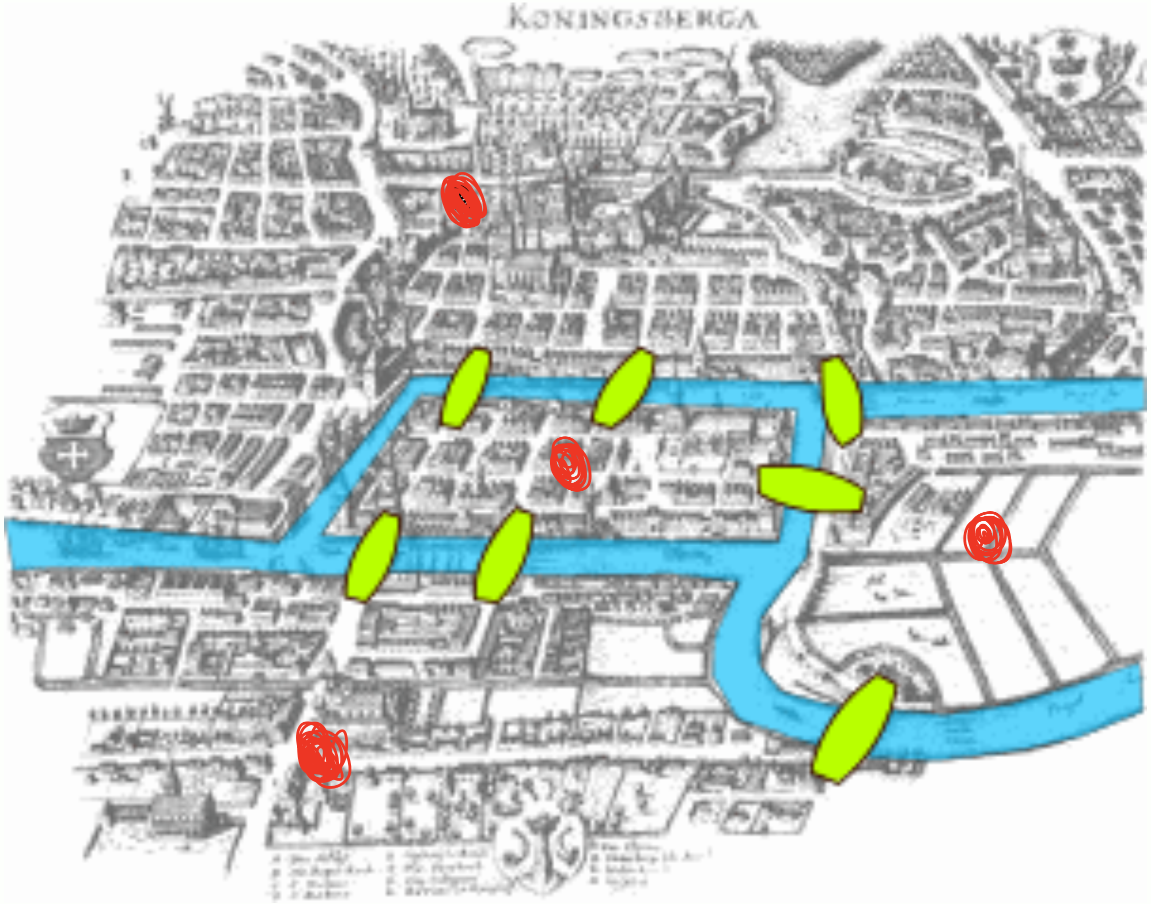


Podstawowe definicje

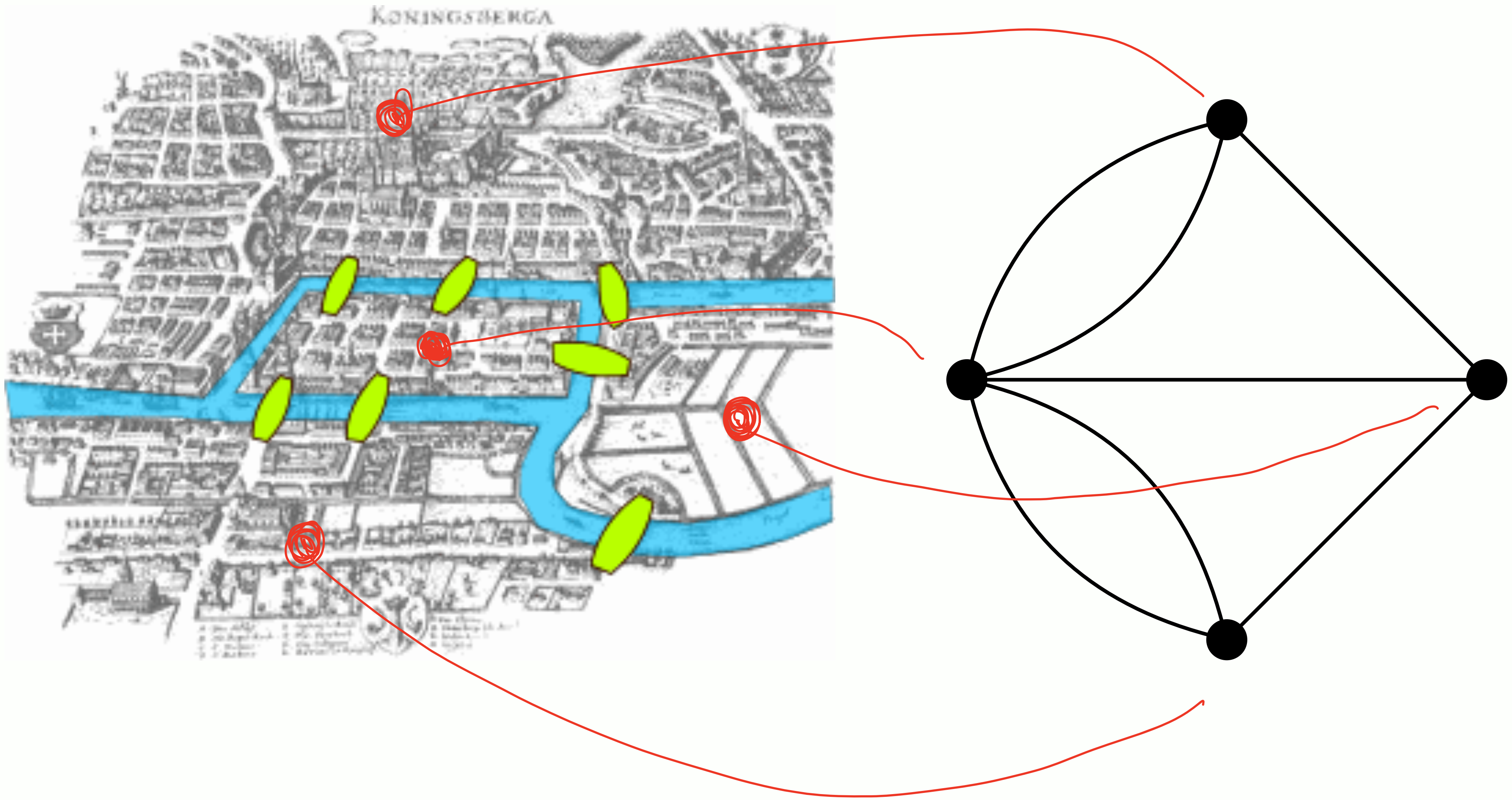
- **Drzewo rozpinające**: podgraf, który zawiera wszystkie wierzchołki i jest drzewem.



Mosty królewieckie

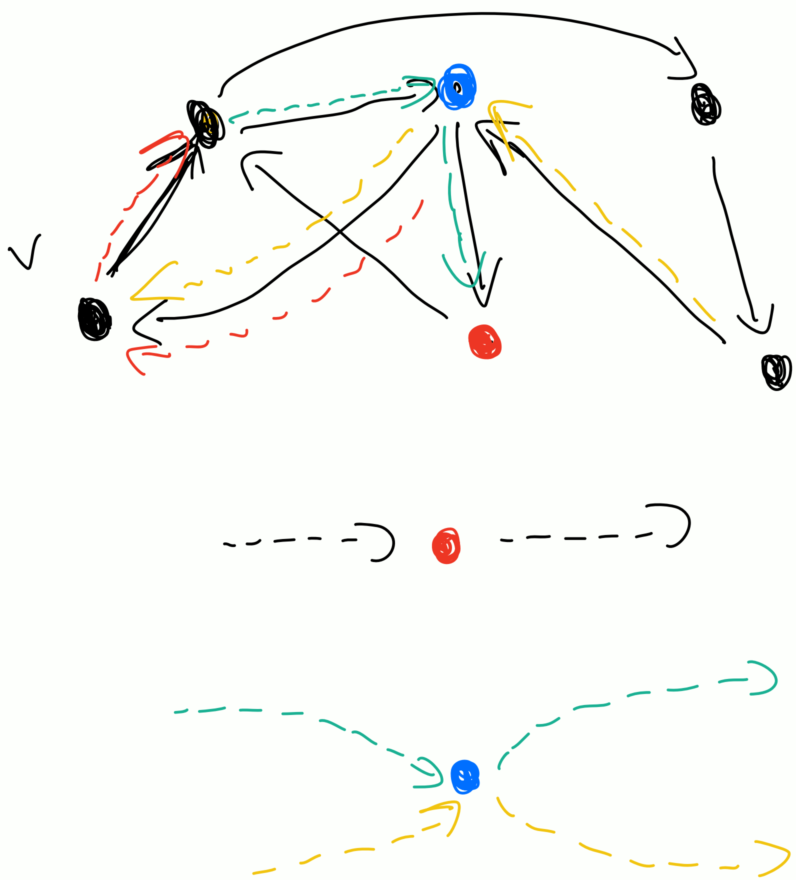


Mosty królewieckie



Grafy eulerowskie

- **Droga Eulera**: droga przechodząca przez wszystkie krawędzie.
- **Obchód Eulera**: zamknięta droga Eulera.
- **Graf eulerowski**: graf posiadający obchód Eulera.



Każdy wierzchołek w grafie eulerowskim ma parzysty stopień.

Grafy eulerowskie

- **Droga Eulera**: droga przechodząca przez wszystkie krawędzie.
- **Obchód Eulera**: zamknięta droga Eulera.
- **Graf eulerowski**: graf posiadający obchód Eulera.

Twierdzenie

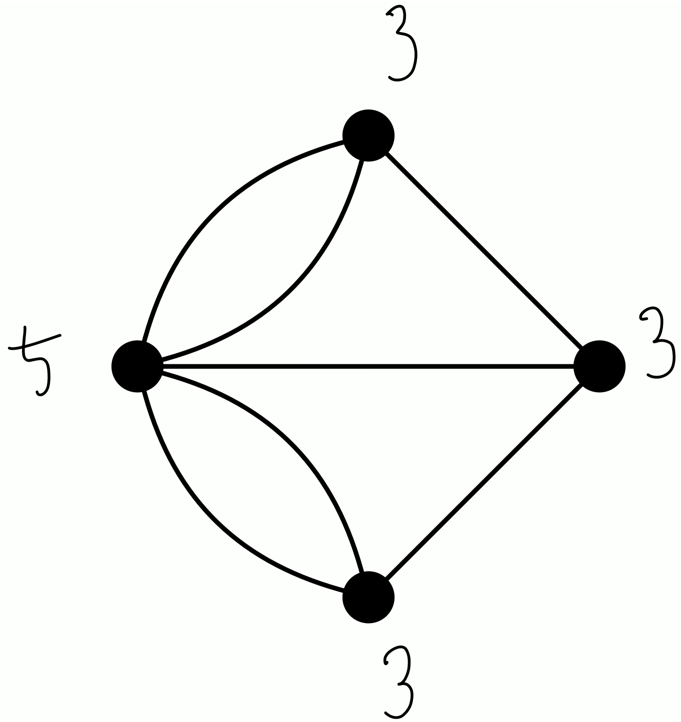
W grafie eulerowskim każdy wierzchołek jest stopnia parzystego.

Grafy eulerowskie

- **Droga Eulera**: droga przechodząca przez wszystkie krawędzie.
- **Obchód Eulera**: zamknięta droga Eulera.
- **Graf eulerowski**: graf posiadający obchód Eulera.

Twierdzenie

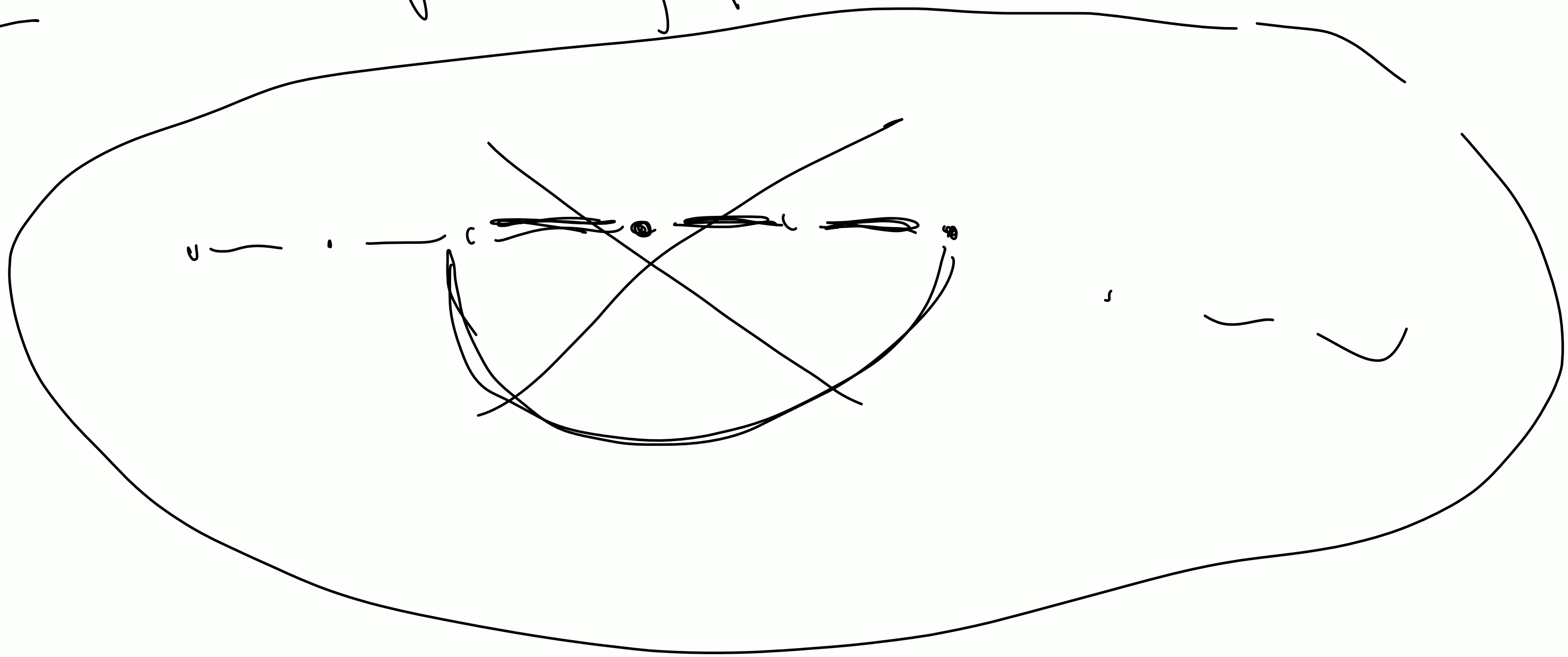
W grafie eulerowskim każdy wierzchołek jest stopnia parzystego.



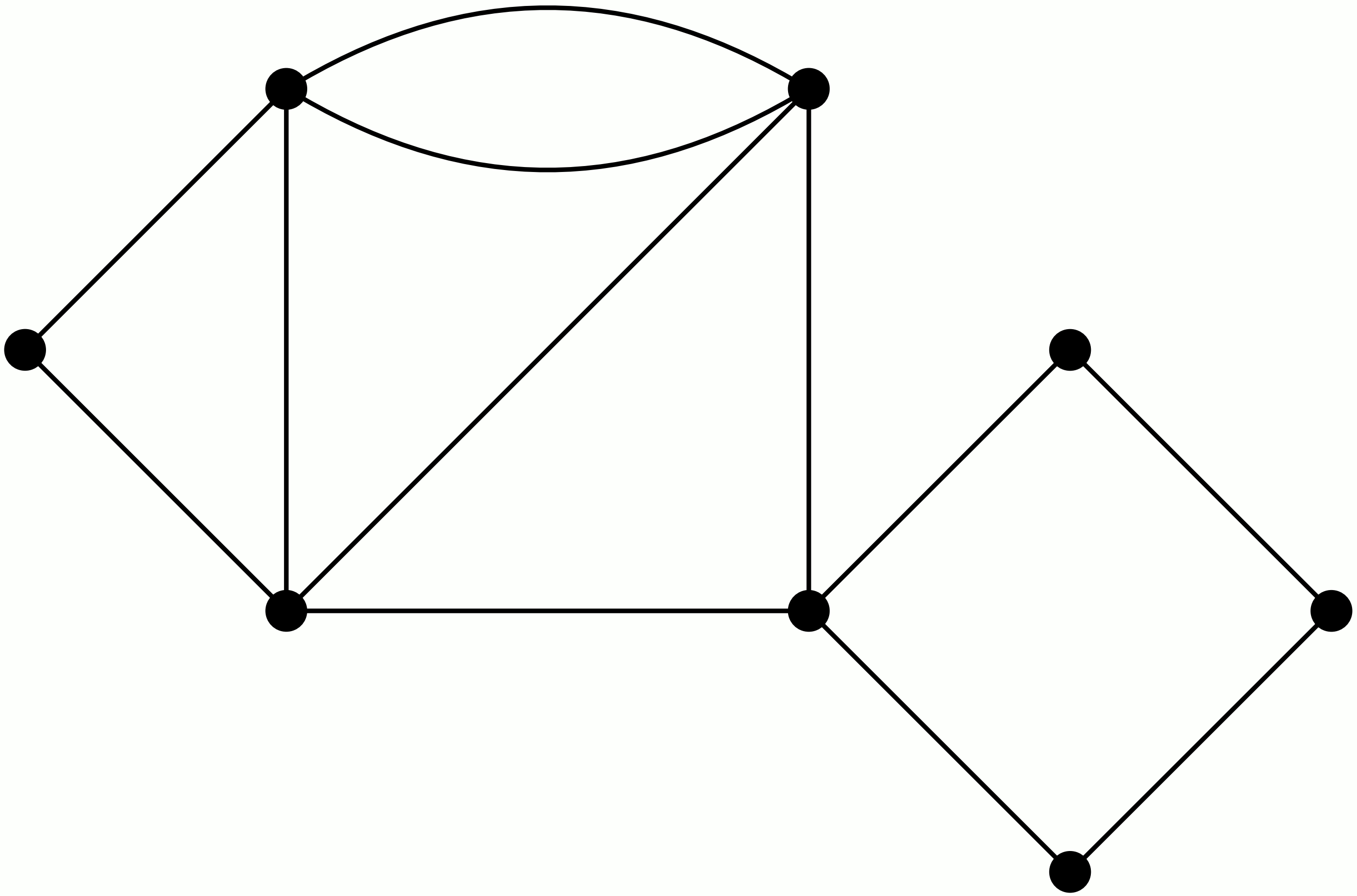
Twierdzenie Eulera

Graf jest eulerowski wtedy i tylko wtedy, gdy jest spójny i każdy jego wierzchołek ma stopień parzysty.

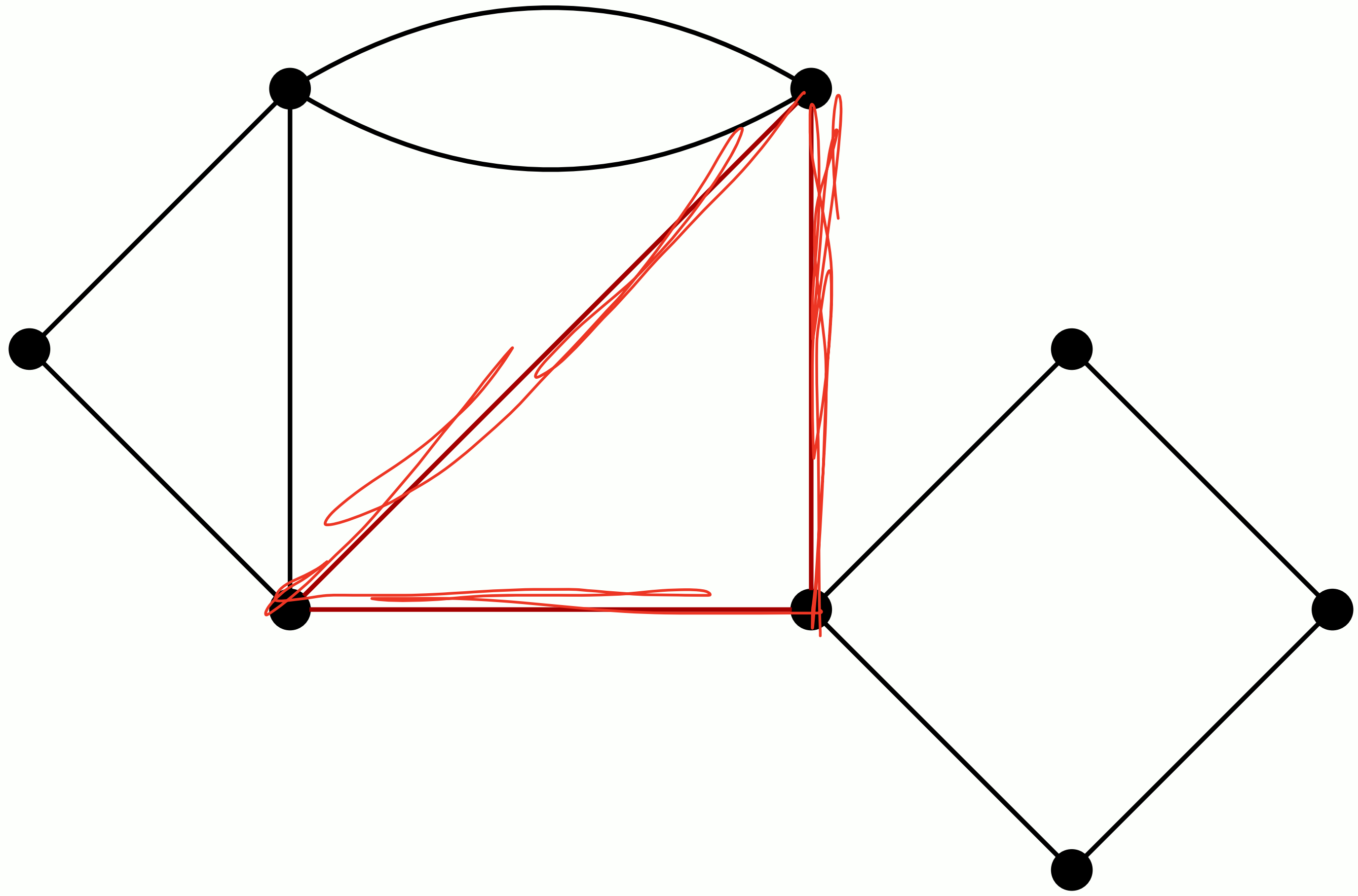
Dow. Indukcja względem liczby krawędzi.



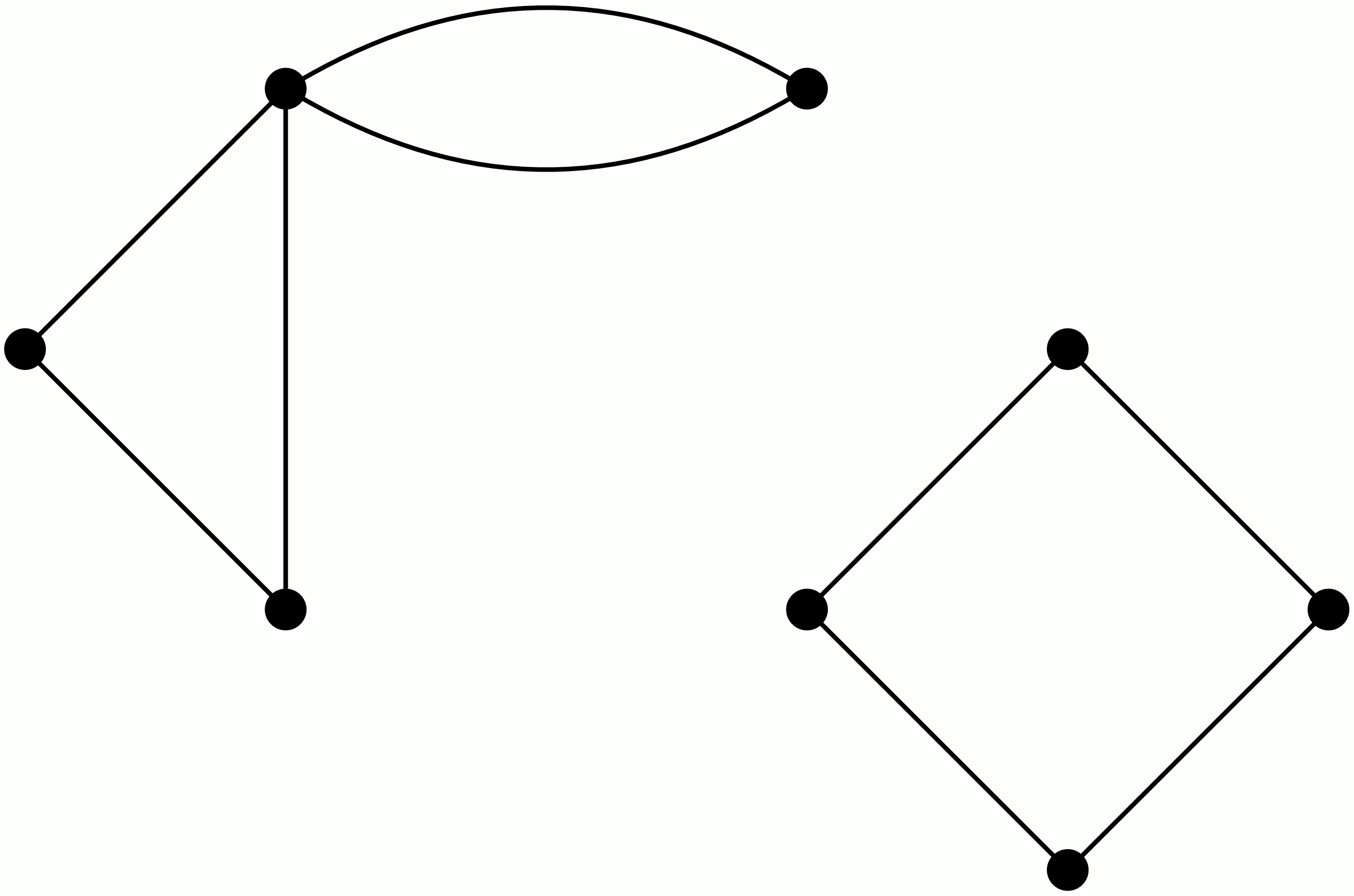
Twierdzenie Eulera



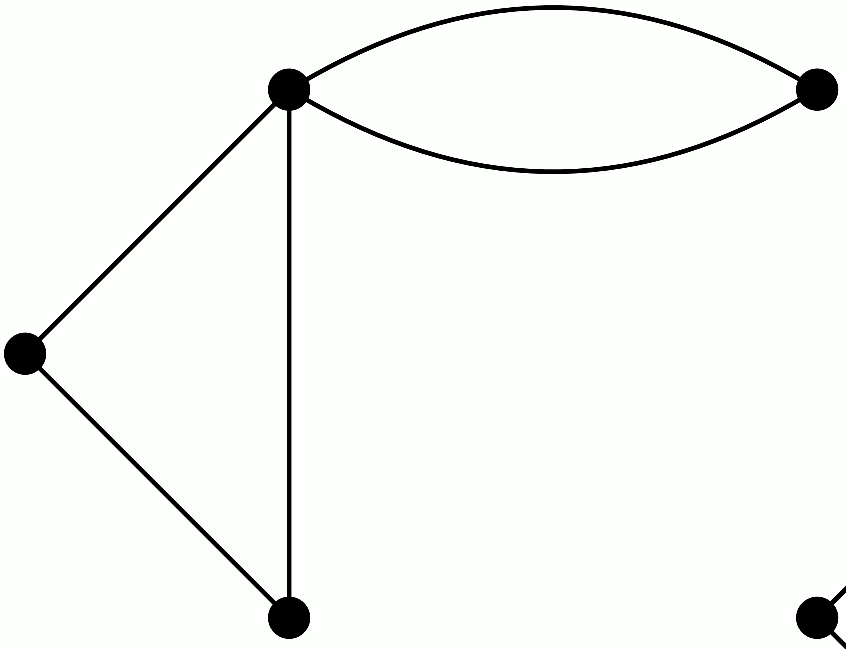
Twierdzenie Eulera

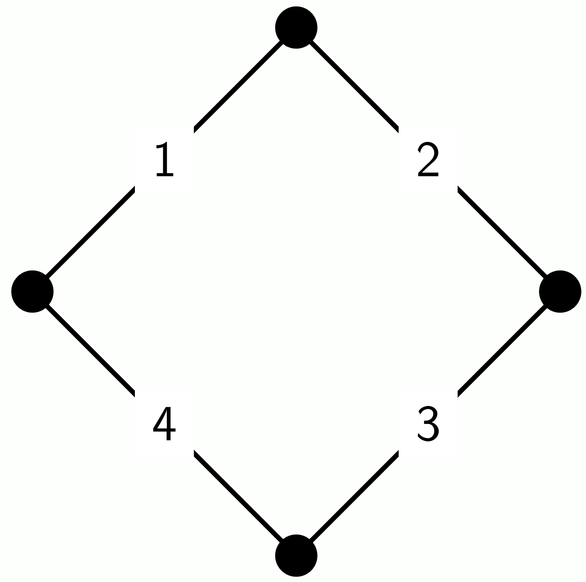


Twierdzenie Eulera

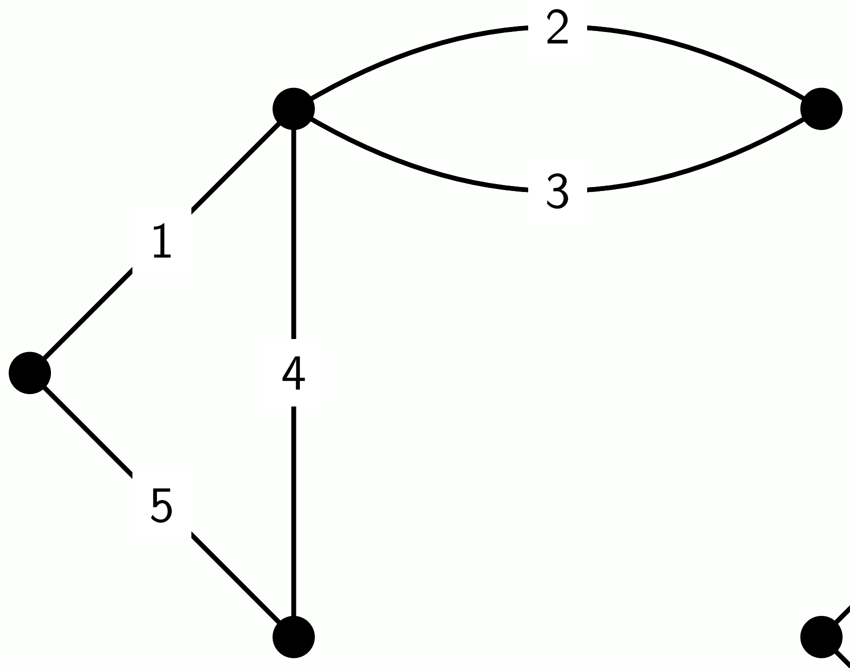


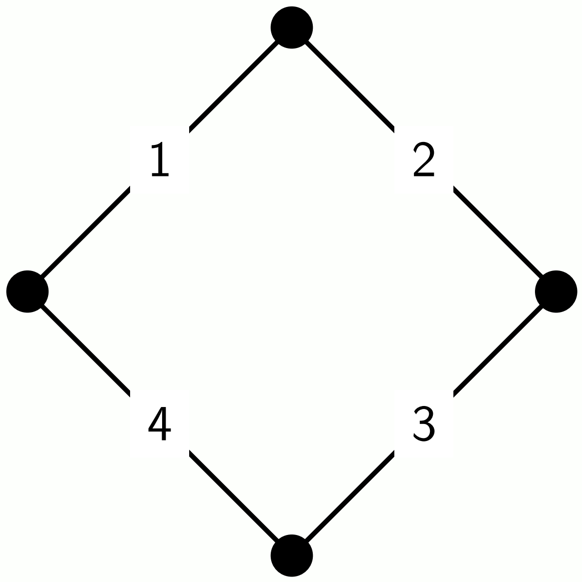
Twierdzenie Eulera



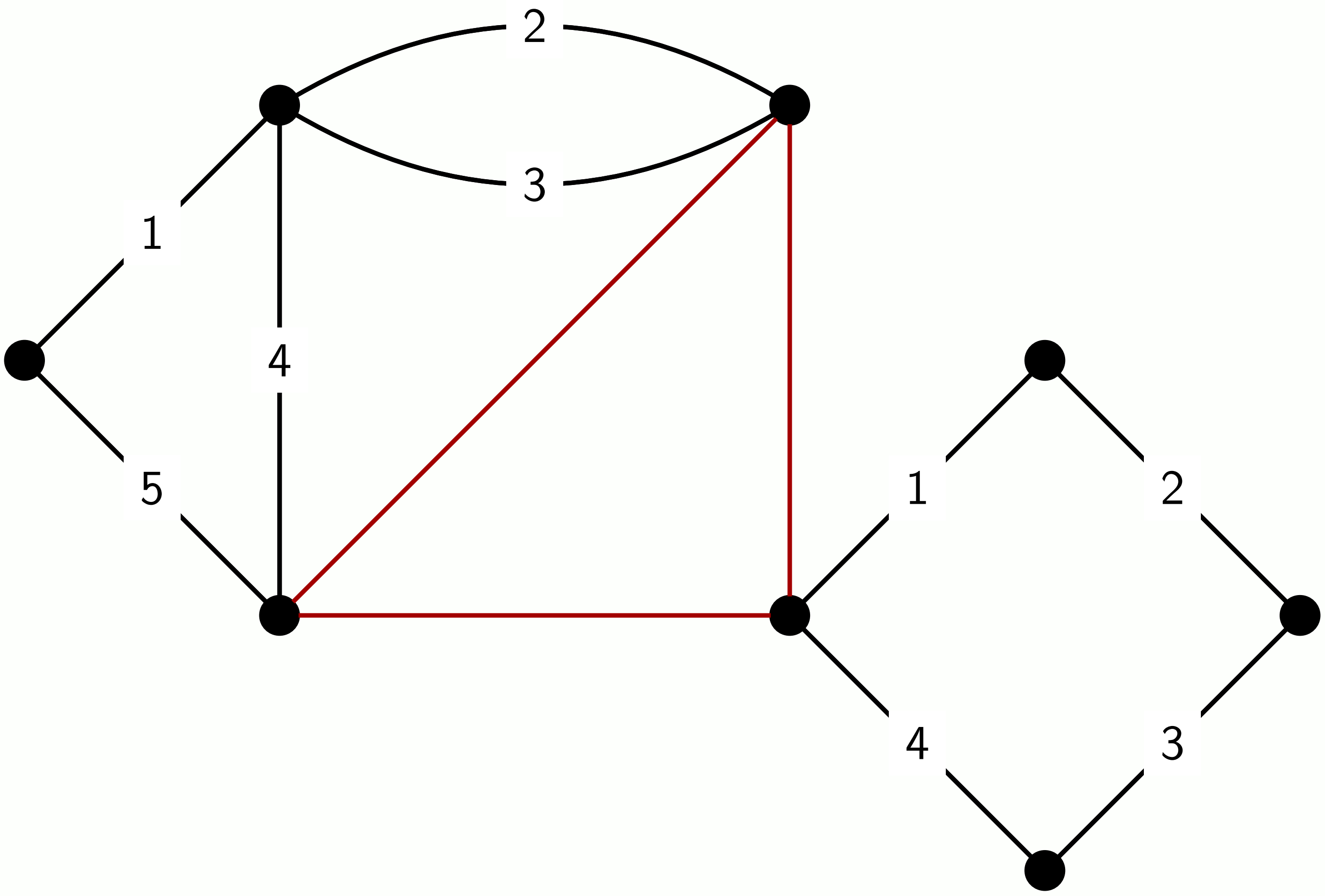


Twierdzenie Eulera

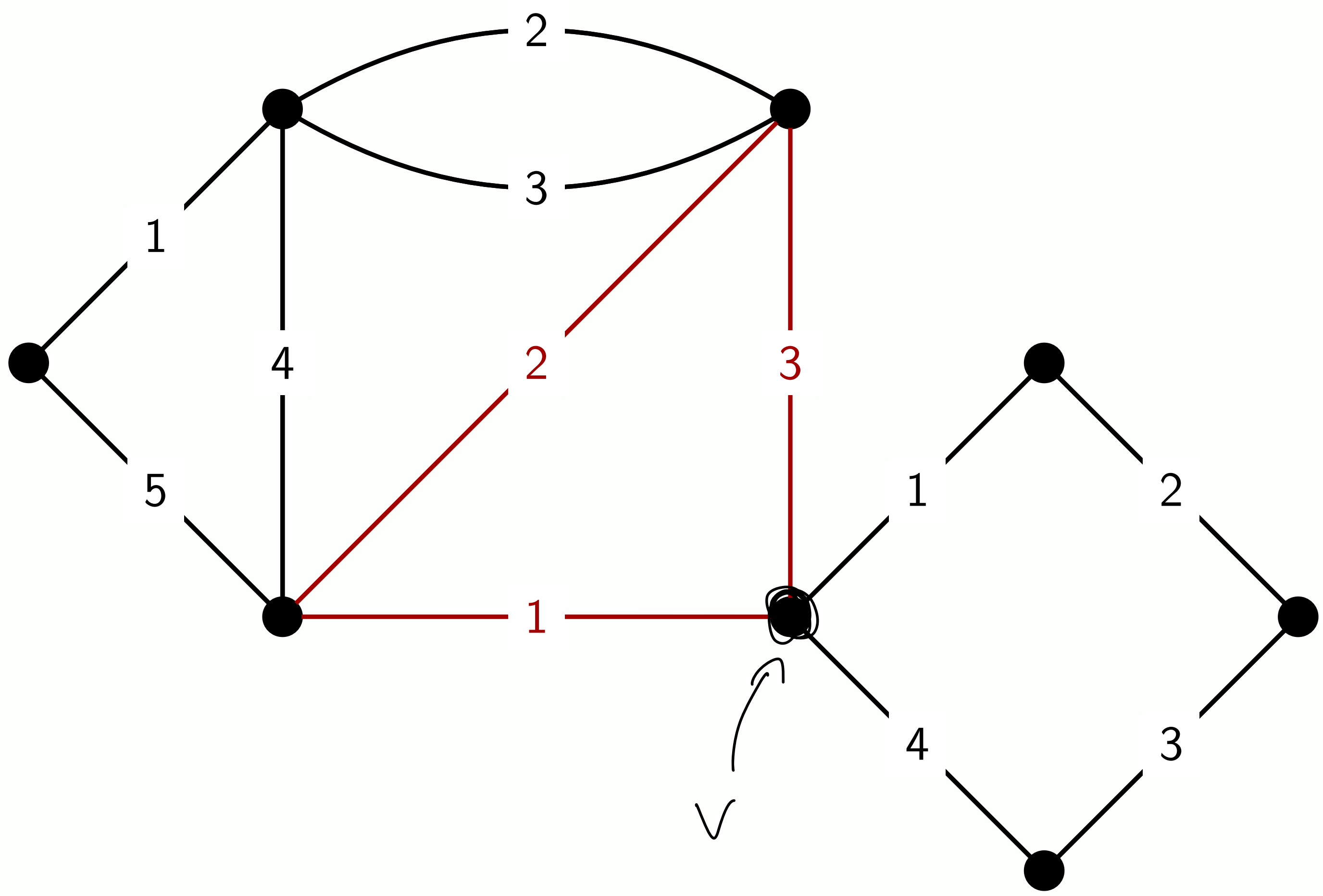




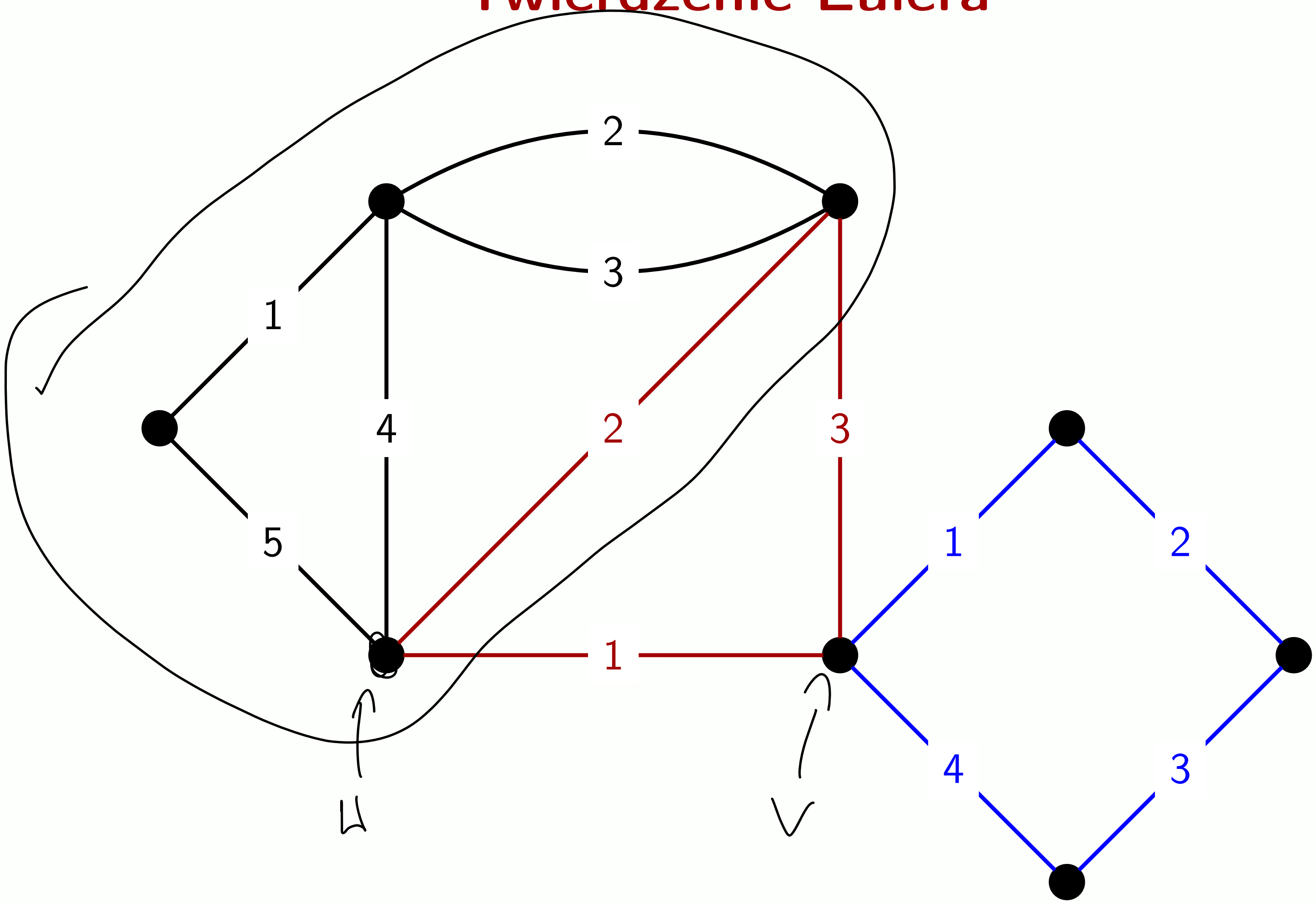
Twierdzenie Eulera



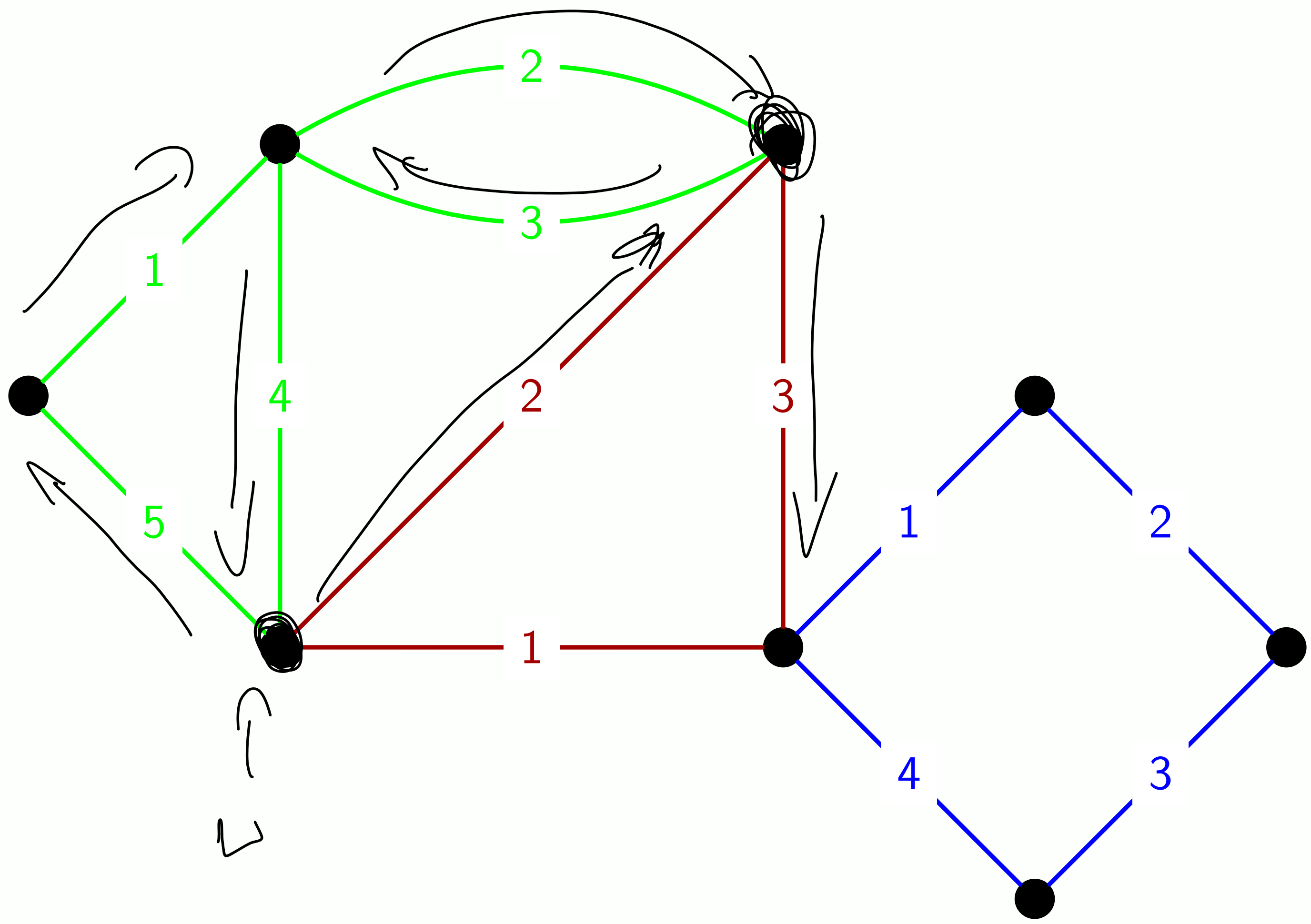
Twierdzenie Eulera



Twierdzenie Eulera



Twierdzenie Eulera



Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.

Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.

Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.
   - $\rightsquigarrow$ Jeżeli z wierzchołka v nie wychodzi żadna krawędź, to przerwij.

Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.
   - $\leadsto$ Jeżeli z wierzchołka v nie wychodzi żadna krawędź, to przerwij.
   - $\leadsto$ Jeżeli pozostała dokładnie jedna krawędź $e = vw$ wychodząca z wierzchołka v do w, to usuń ten wierzchołek i tę krawędź.

Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.
   - ⇝ Jeżeli z wierzchołka v nie wychodzi żadna krawędź, to przerwij.
   - ⇝ Jeżeli pozostała dokładnie jedna krawędź $e = vw$ wychodząca z wierzchołka v do w, to usuń ten wierzchołek i tę krawędź.
   - ⇝ Jeżeli pozostała więcej niż jedna krawędź wychodząca z v, to wybierz taką krawędź $e = vw$, po usunięciu której graf pozostanie spójny, a następnie usuń tę krawędź.

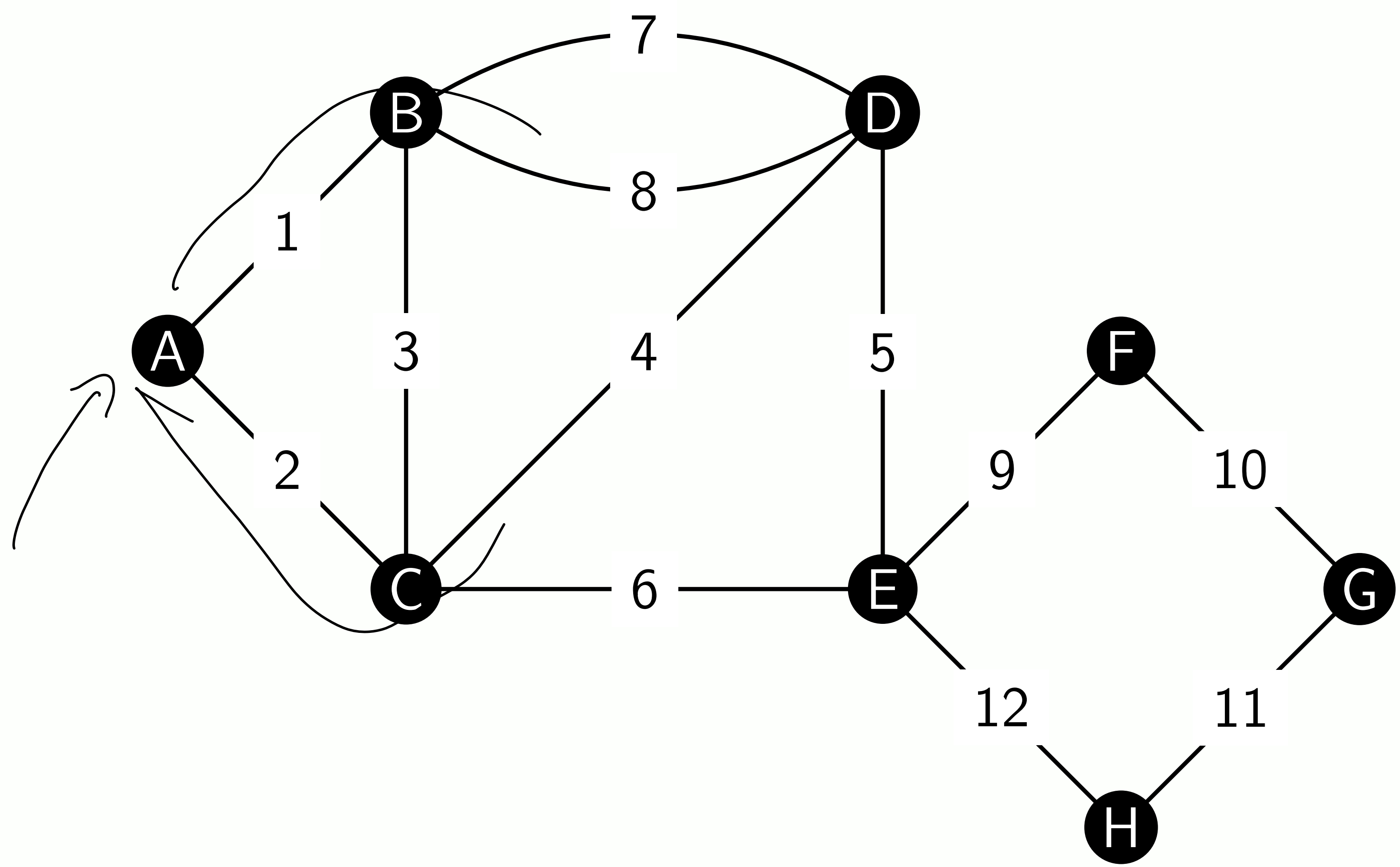
Algorytm Fleury'ego

1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.
   - Jeżeli z wierzchołka v nie wychodzi żadna krawędź, to przerwij.
   - Jeżeli pozostała dokładnie jedna krawędź $e = vw$ wychodząca z wierzchołka v do w, to usuń ten wierzchołek i tę krawędź.
   - Jeżeli pozostała więcej niż jedna krawędź wychodząca z v, to wybierz taką krawędź $e = vw$, po usunięciu której graf pozostanie spójny, a następnie usuń tę krawędź.
3. Dodaj e do E_E.

Algorytm Fleury'ego

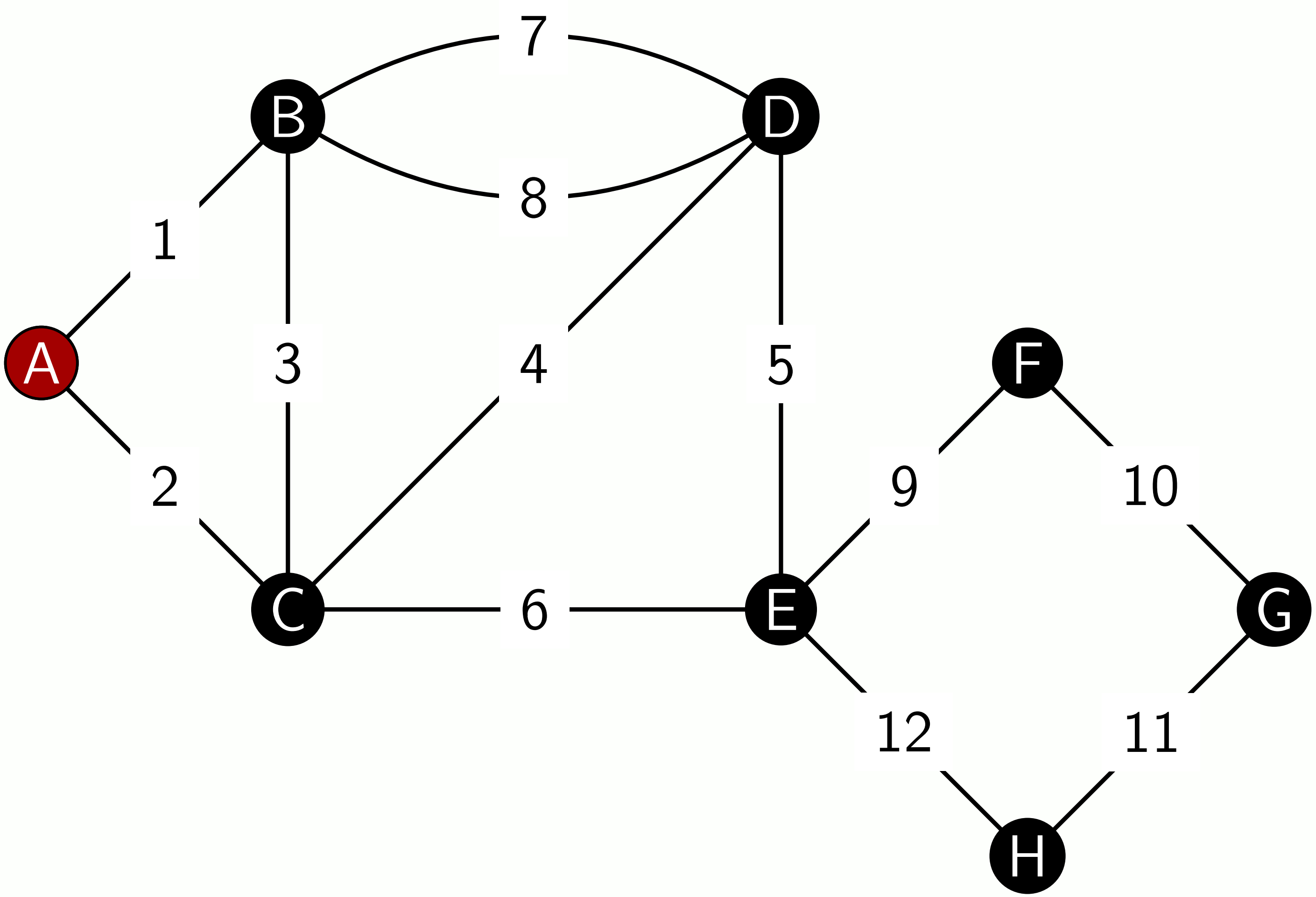
1. E_E: poszukiwany ciąg krawędzi.
2. Jeżeli w grafie istnieje jakiś wierzchołek v stopnia nieparzystego, to go wybierz. Jeżeli taki wierzchołek nie istnieje, to wybierz dowolny wierzchołek v.
   - ⇝ Jeżeli z wierzchołka v nie wychodzi żadna krawędź, to przerwij.
   - ⇝ Jeżeli pozostała dokładnie jedna krawędź $e = vw$ wychodząca z wierzchołka v do w, to usuń ten wierzchołek i tę krawędź.
   - ⇝ Jeżeli pozostała więcej niż jedna krawędź wychodząca z v, to wybierz taką krawędź $e = vw$, po usunięciu której graf pozostanie spójny, a następnie usuń tę krawędź.
3. Dodaj e do E_E.
4. Zastąp v przez w i wróć do kroku 2.

Algorytm Fleury'ego



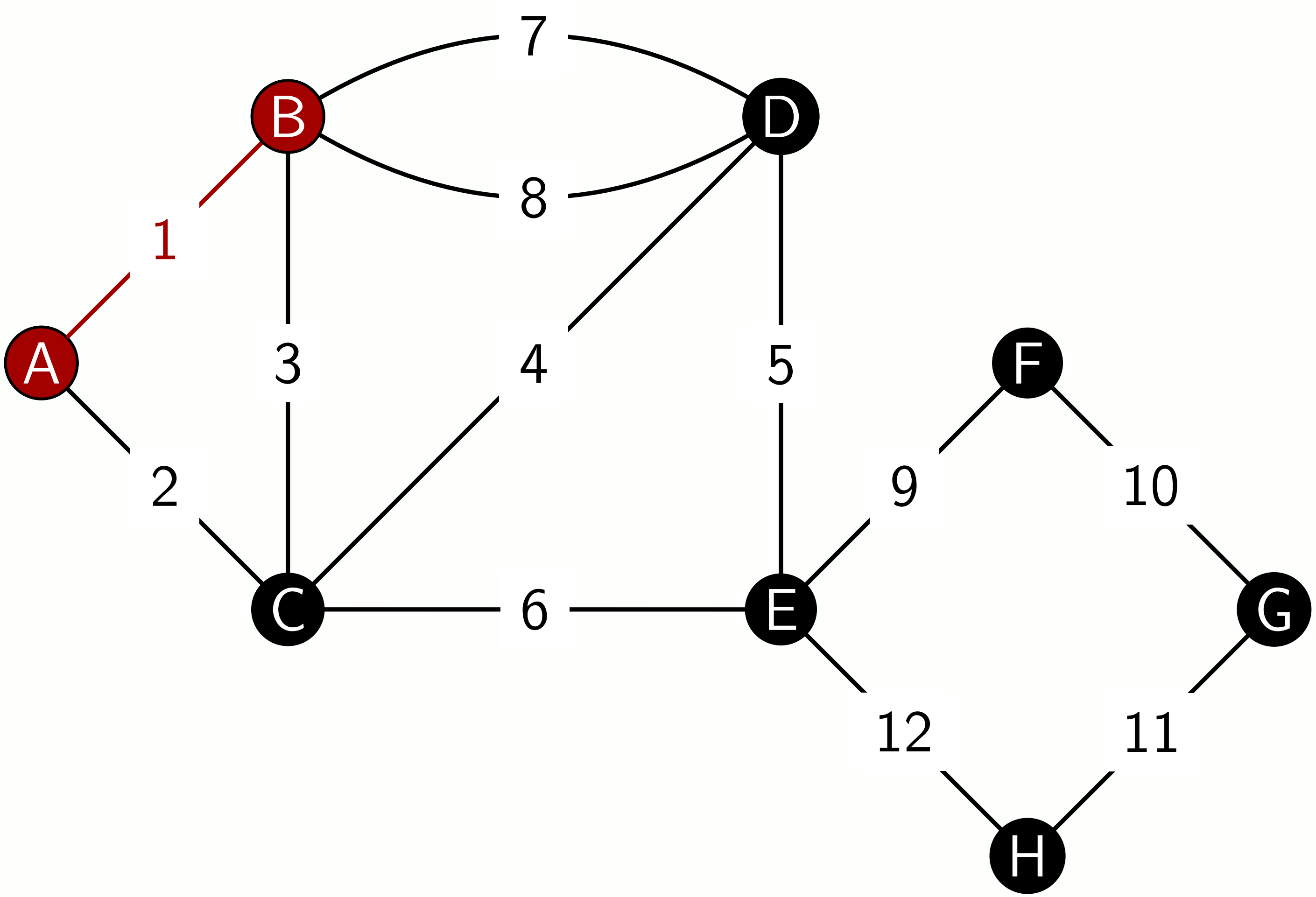
$$E_E =$$

Algorytm Fleury'ego



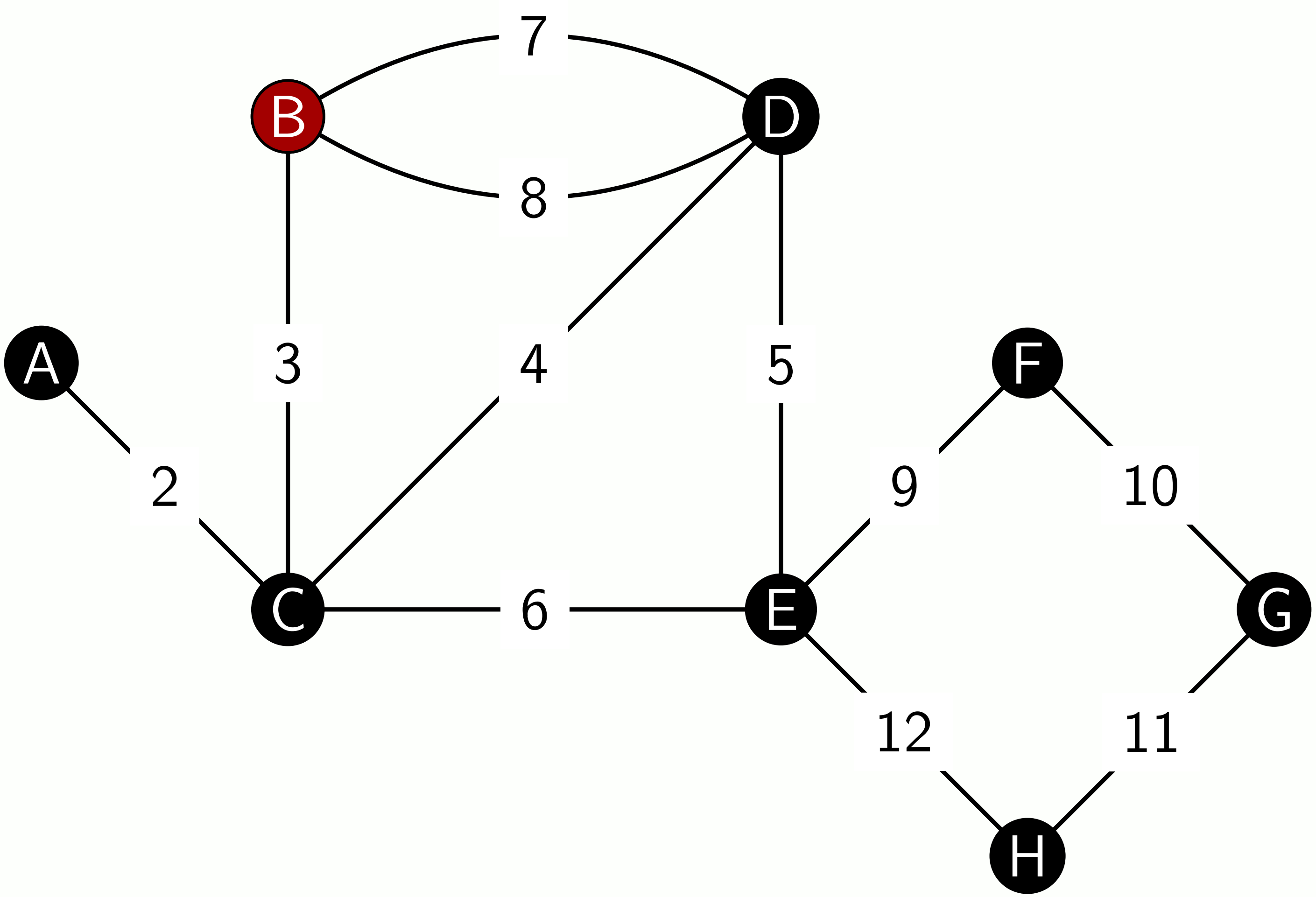
$$E_E =$$

Algorytm Fleury'ego



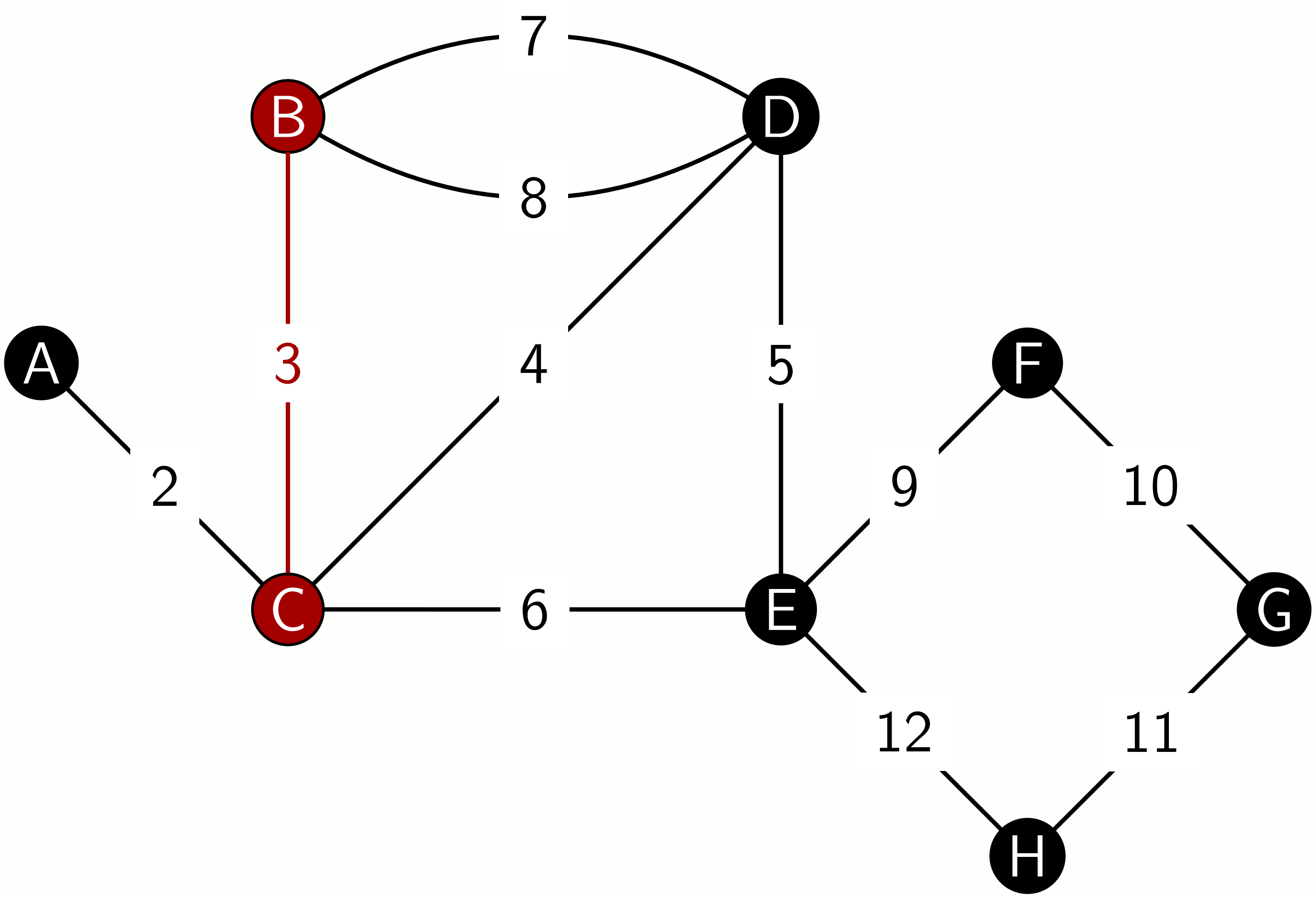
$E_E =$

Algorytm Fleury'ego



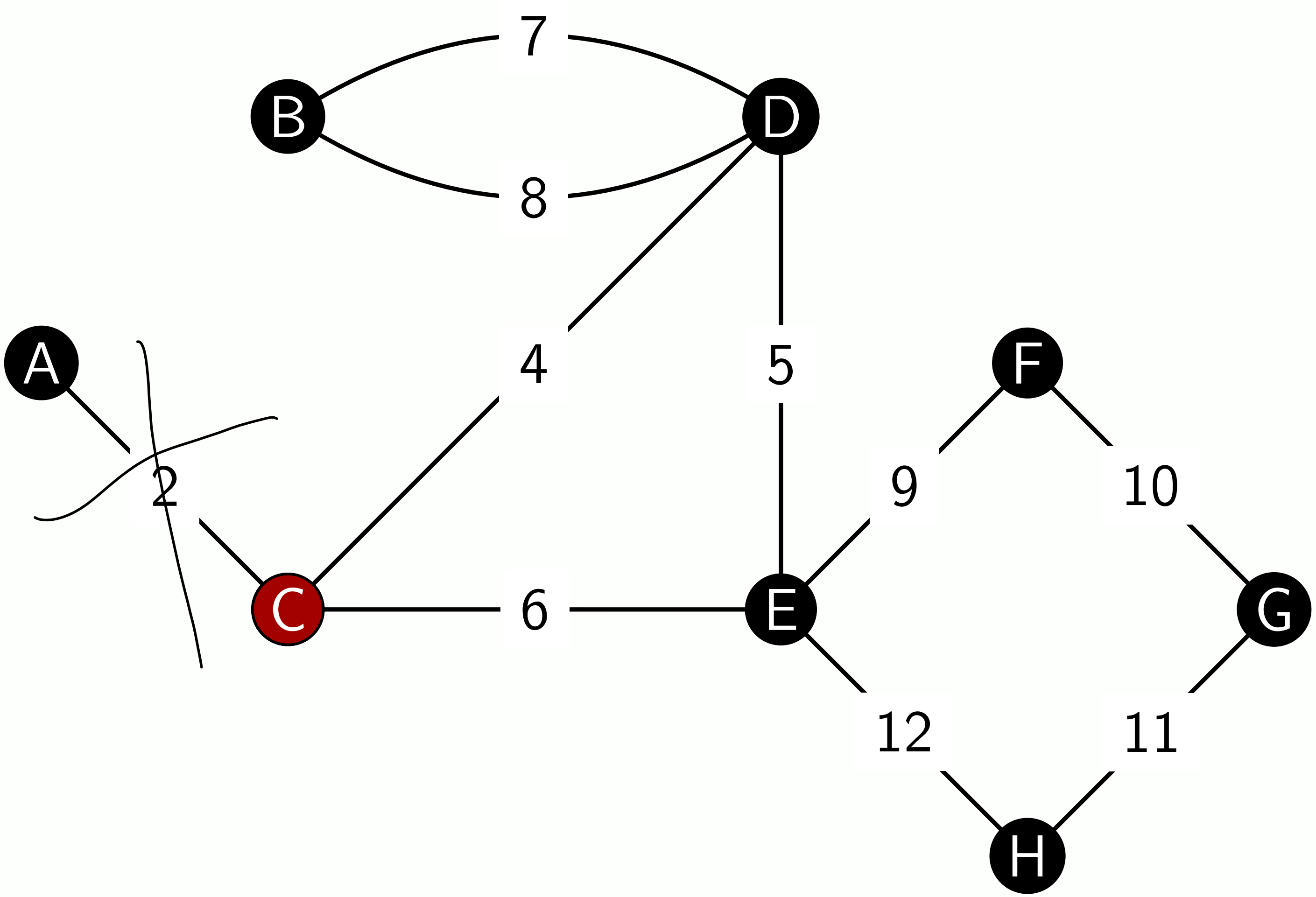
$E_E = 1$

Algorytm Fleury'ego



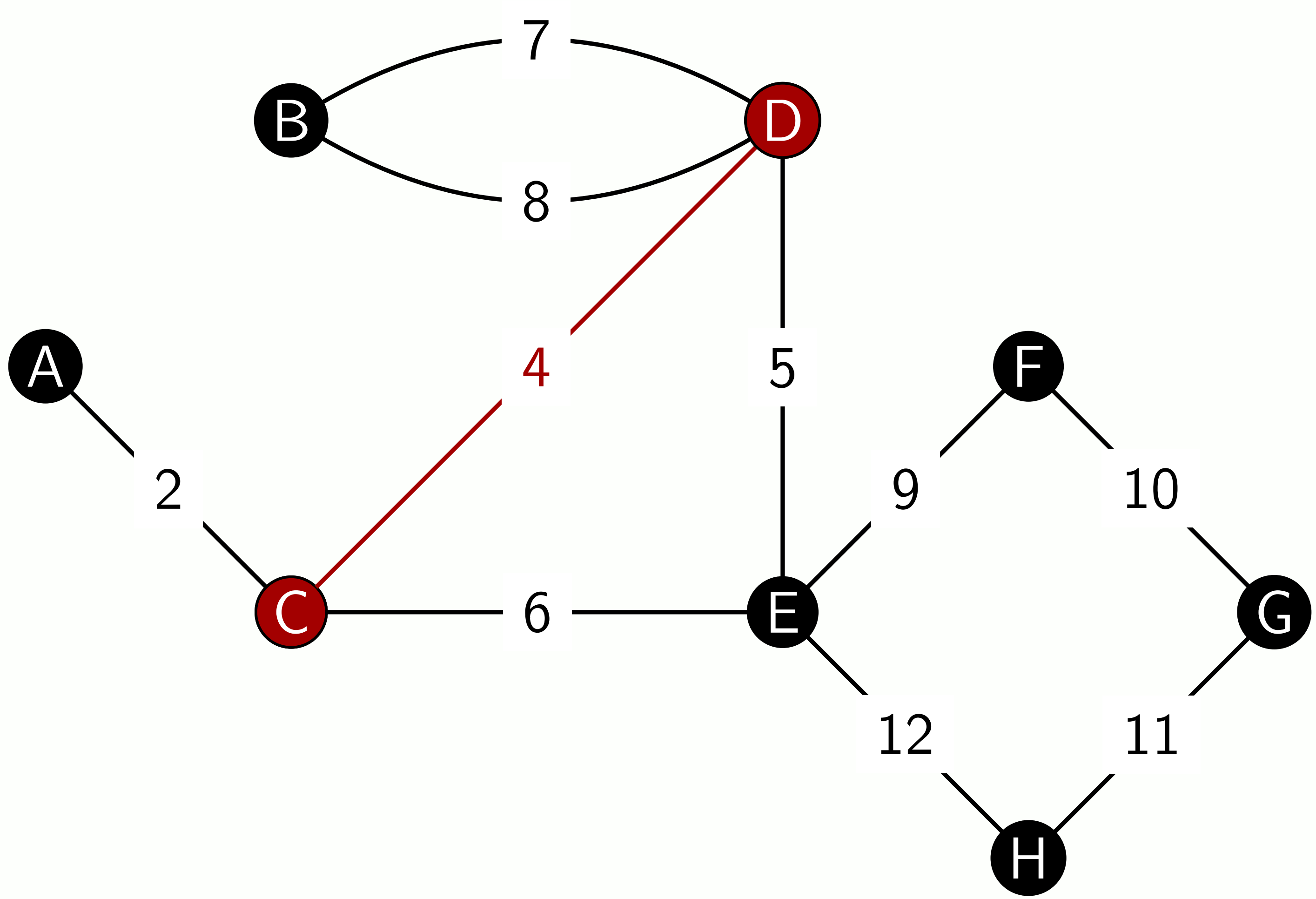
$E_E = 1$

Algorytm Fleury'ego



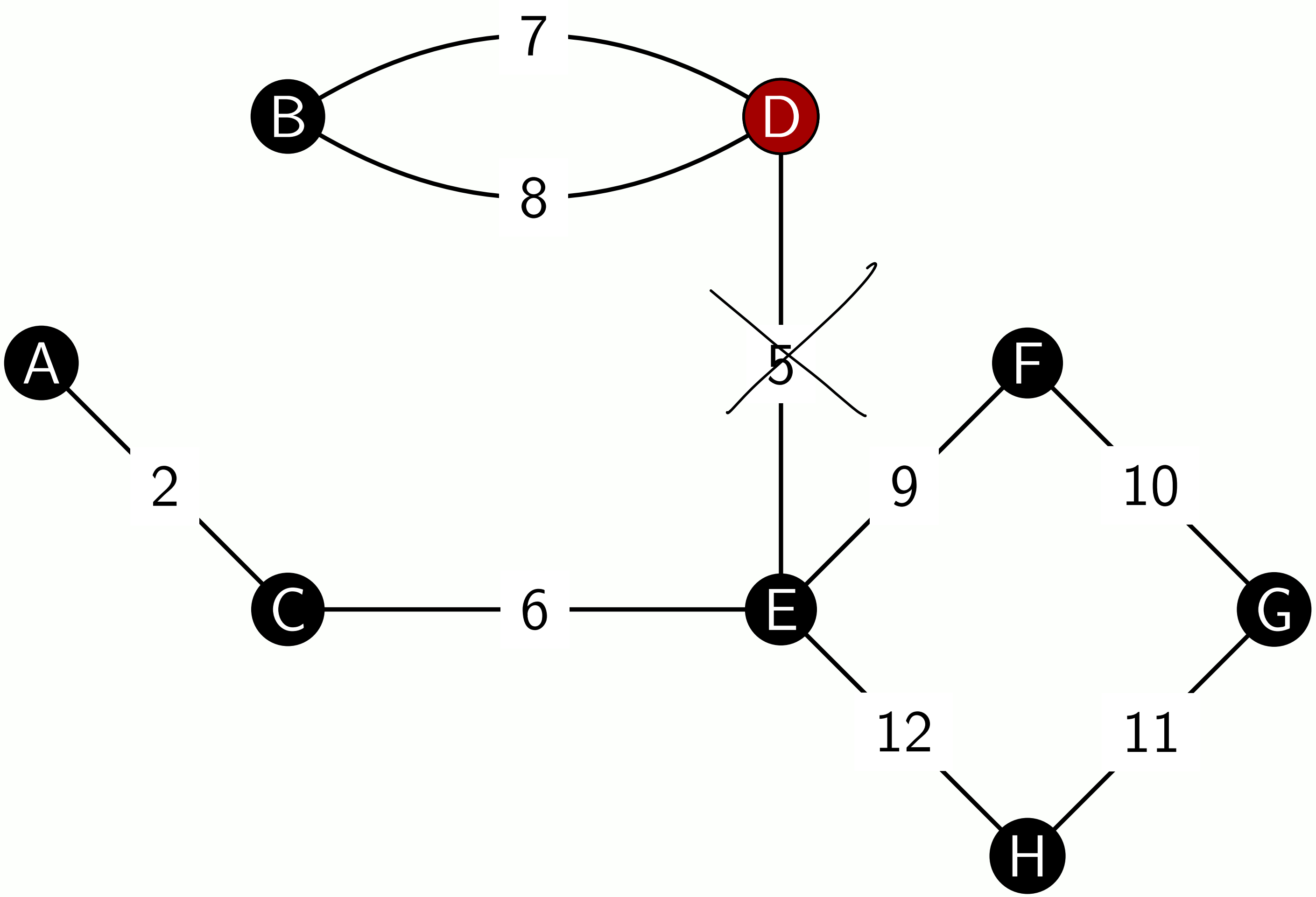
$E_E = 1, 3$

Algorytm Fleury'ego



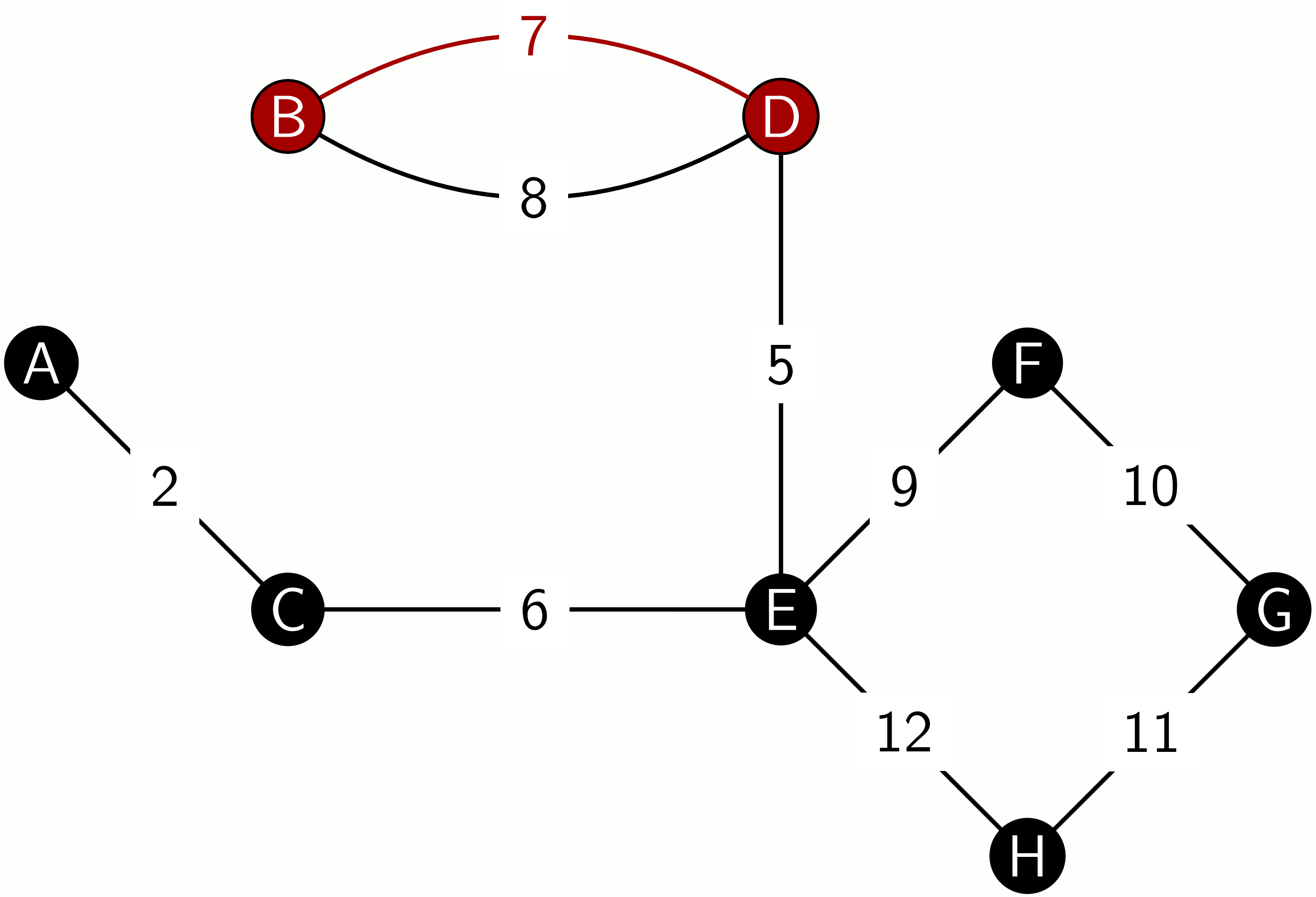
$E_E = 1, 3$

Algorytm Fleury'ego



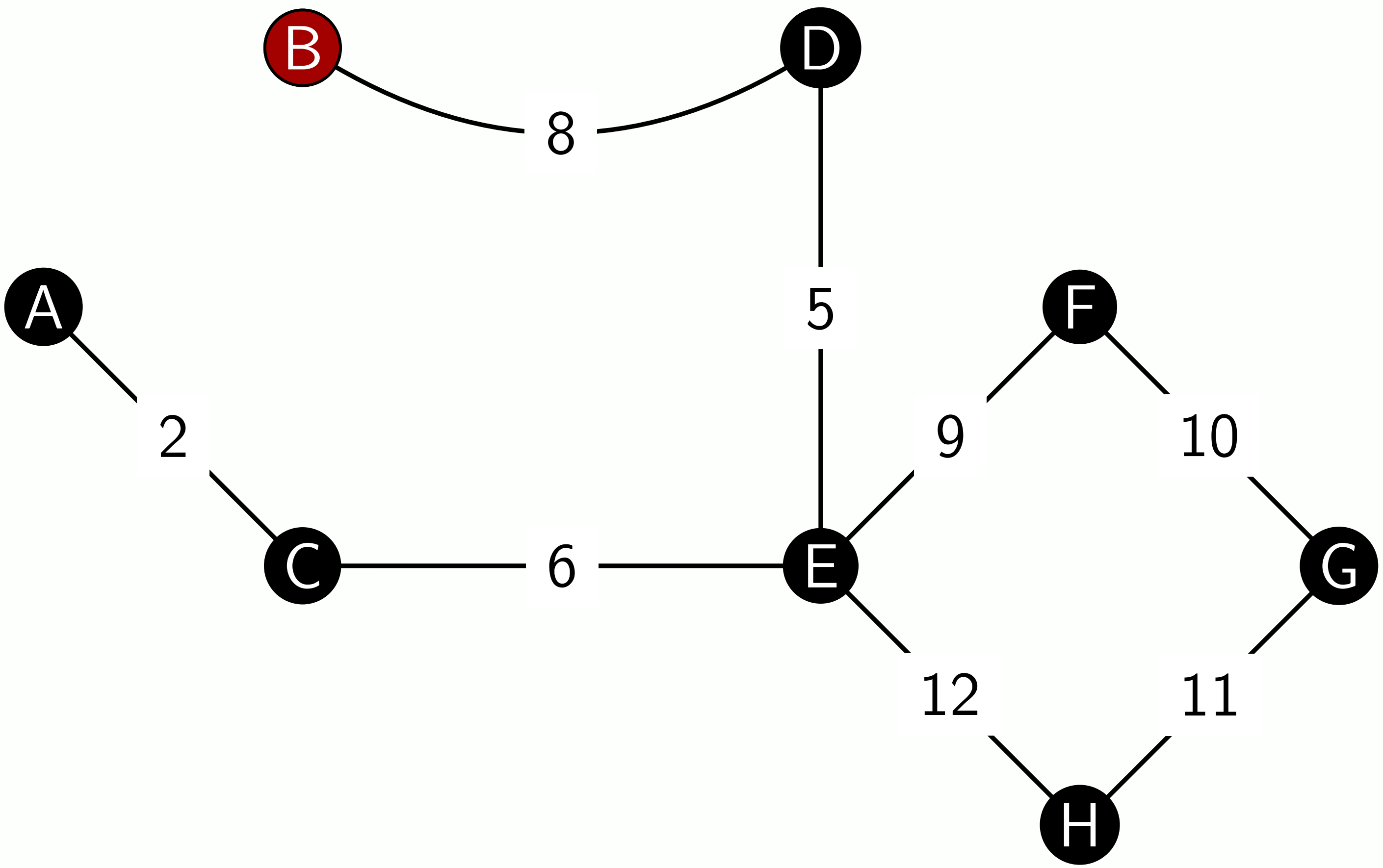
$E_E = 1, 3, 4, \cancel{7}$

Algorytm Fleury'ego



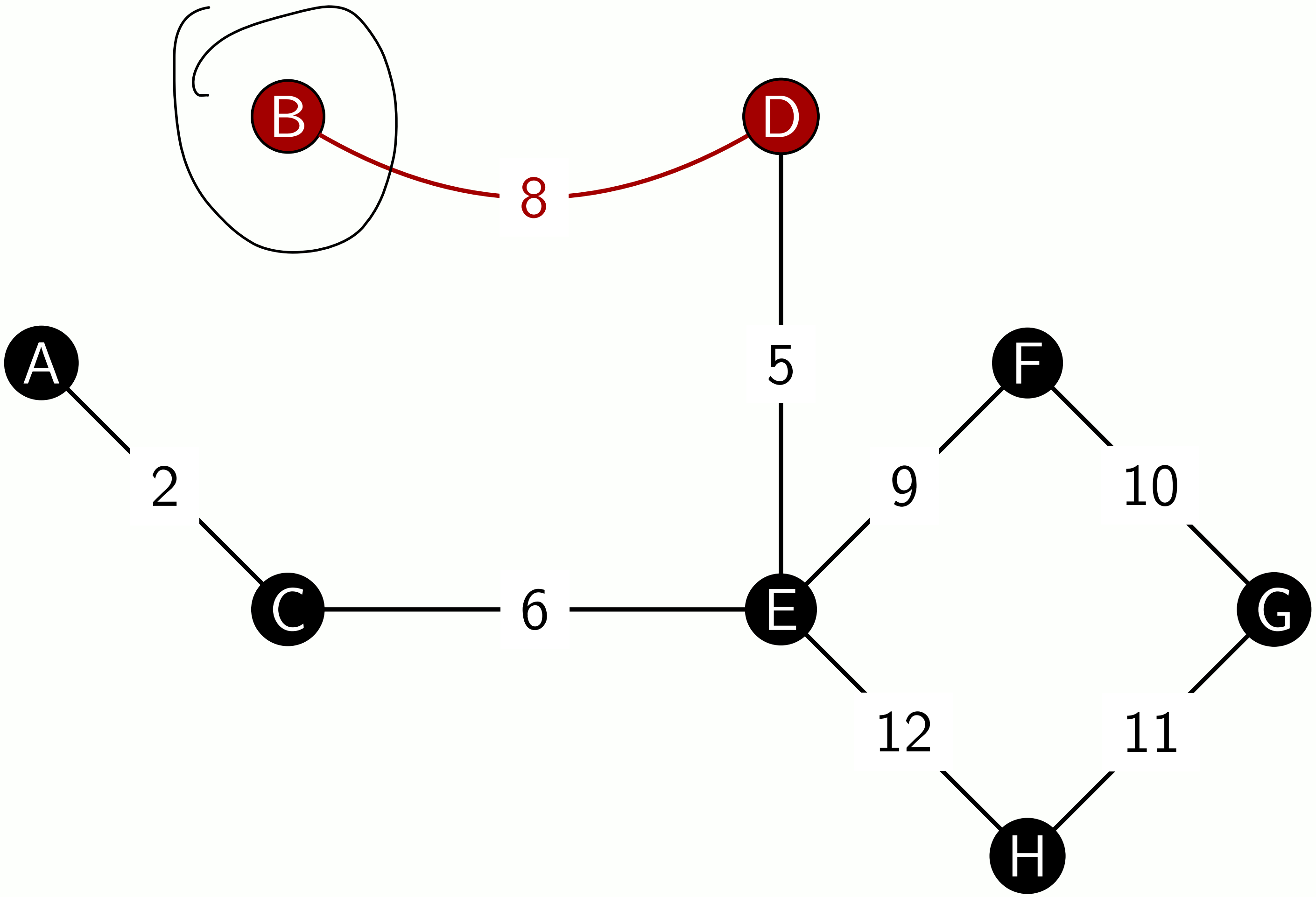
$E_E = 1, 3, 4, 7$

Algorytm Fleury'ego



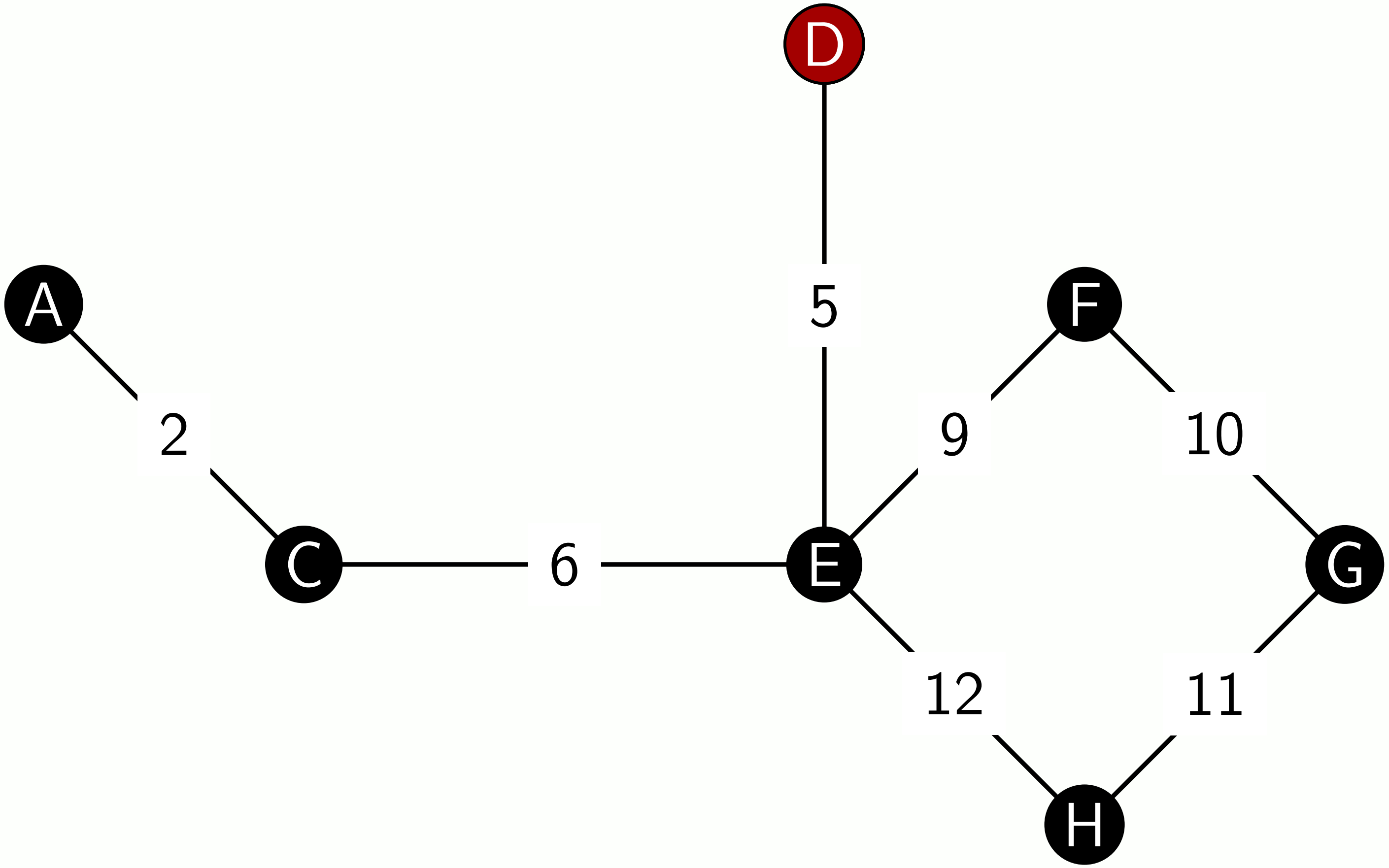
$E_E = 1, 3, 4, 7$

Algorytm Fleury'ego



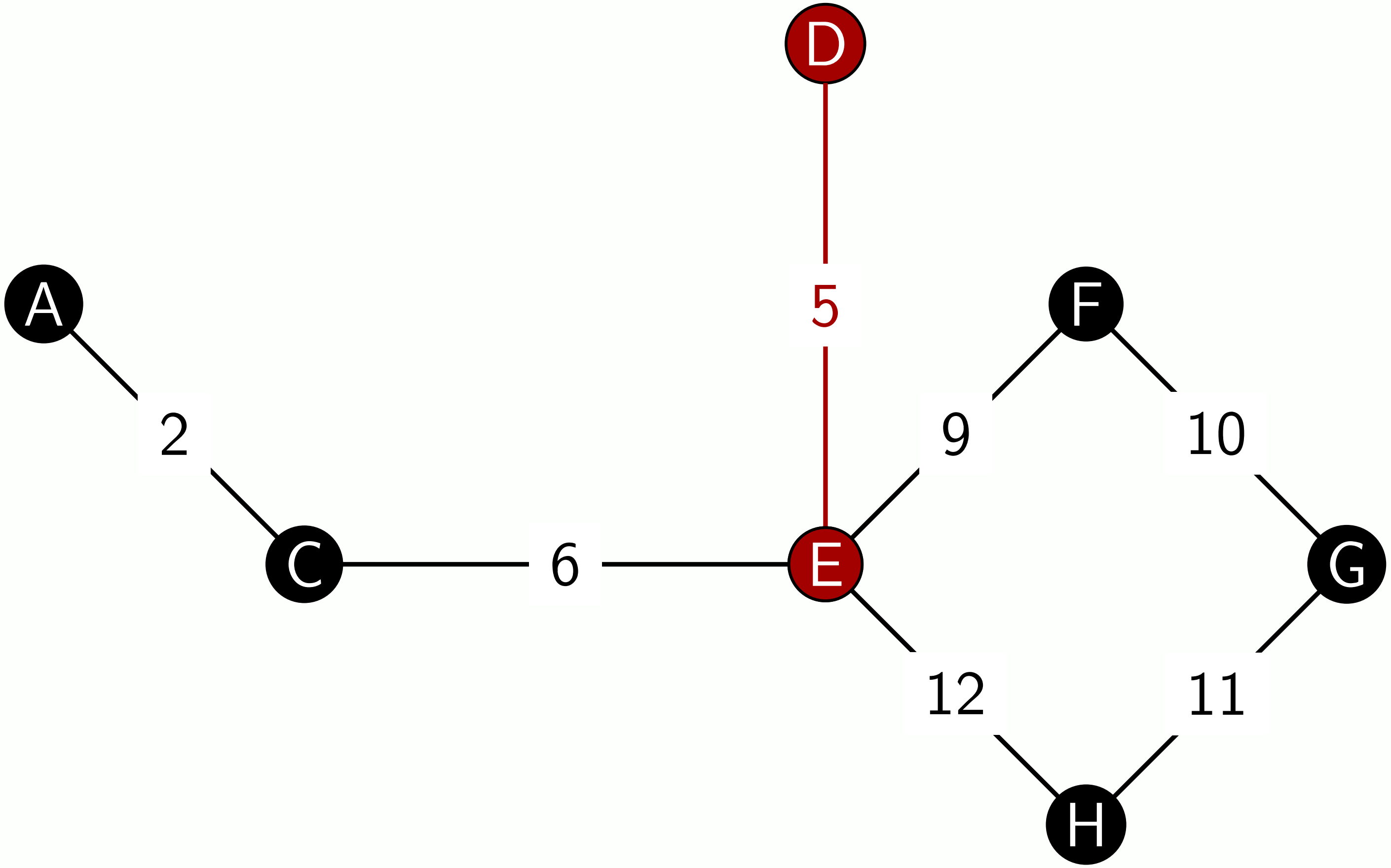
$E_E = 1, 3, 4, 7$

Algorytm Fleury'ego



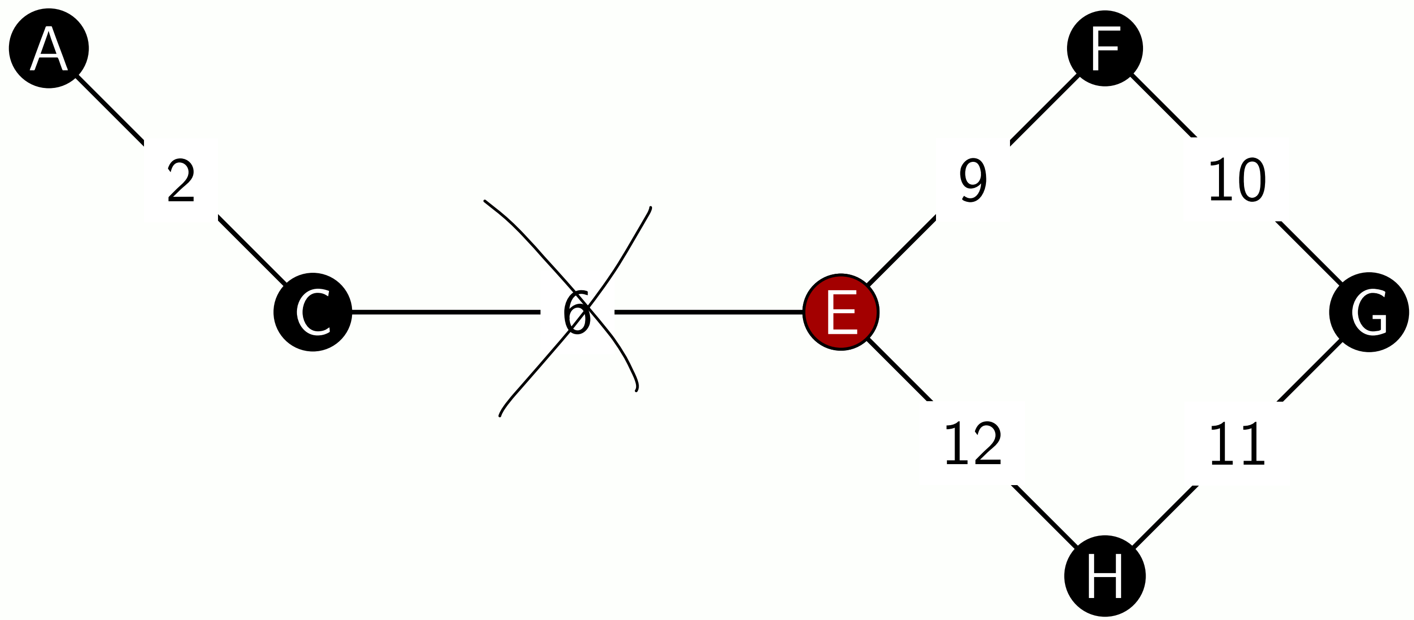
$E_E = 1, 3, 4, 7, 8$

Algorytm Fleury'ego



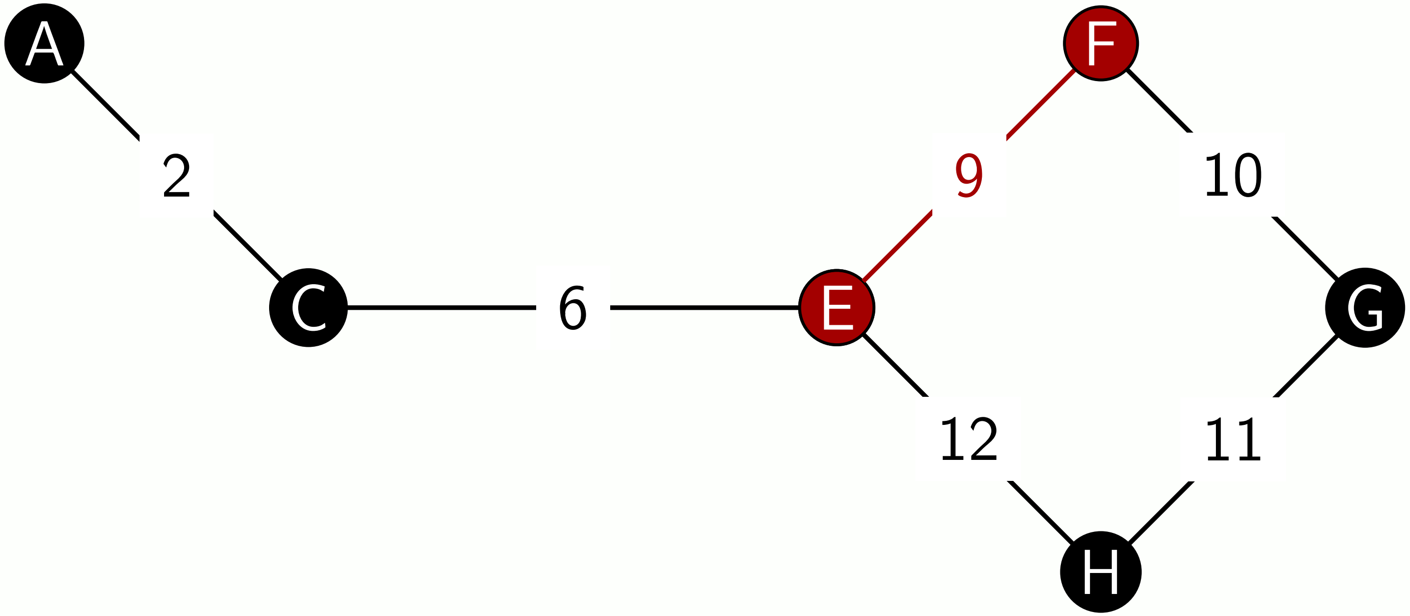
$$E_E = 1, 3, 4, 7, 8$$

Algorytm Fleury'ego



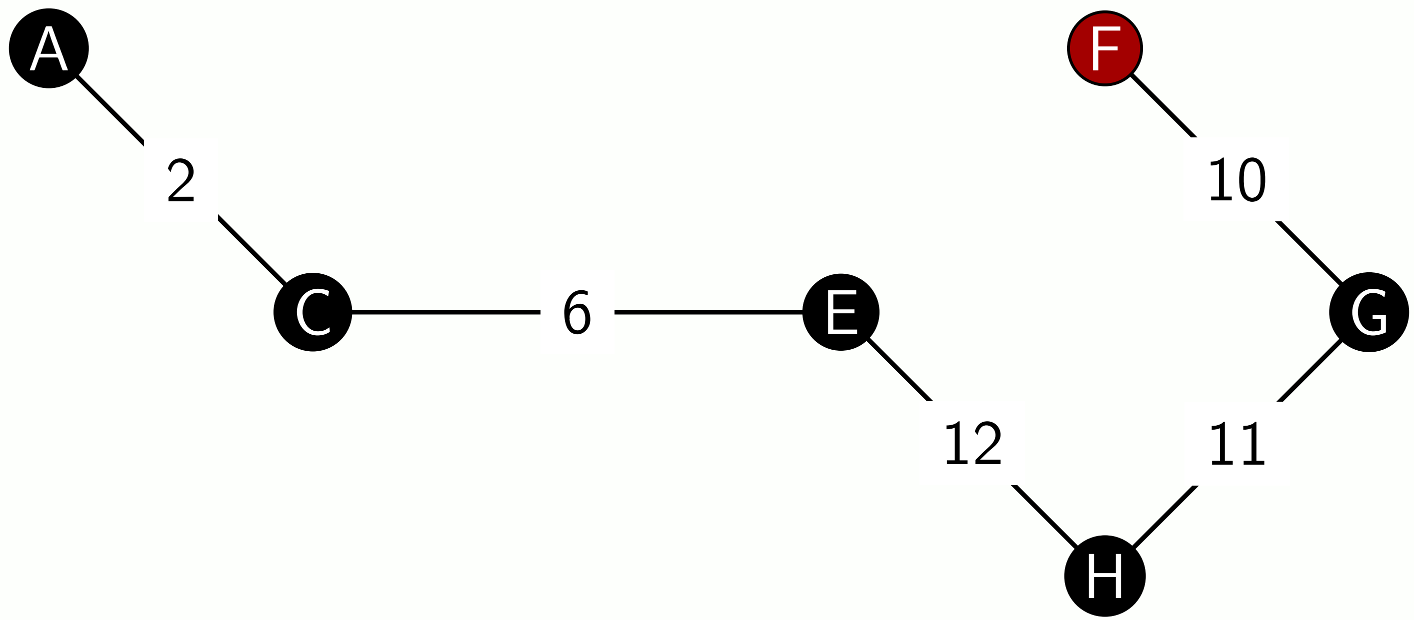
$E_E = 1, 3, 4, 7, 8, 5$

Algorytm Fleury'ego



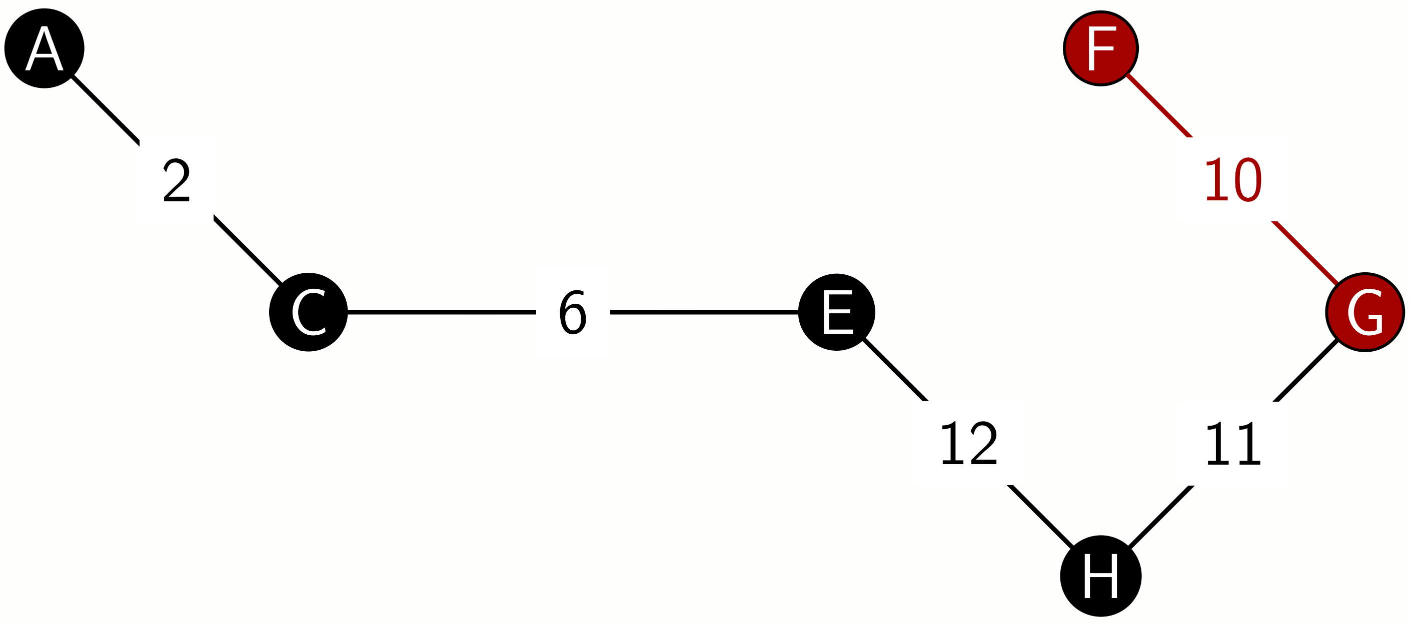
$E_E = 1, 3, 4, 7, 8, 5$

Algorytm Fleury'ego



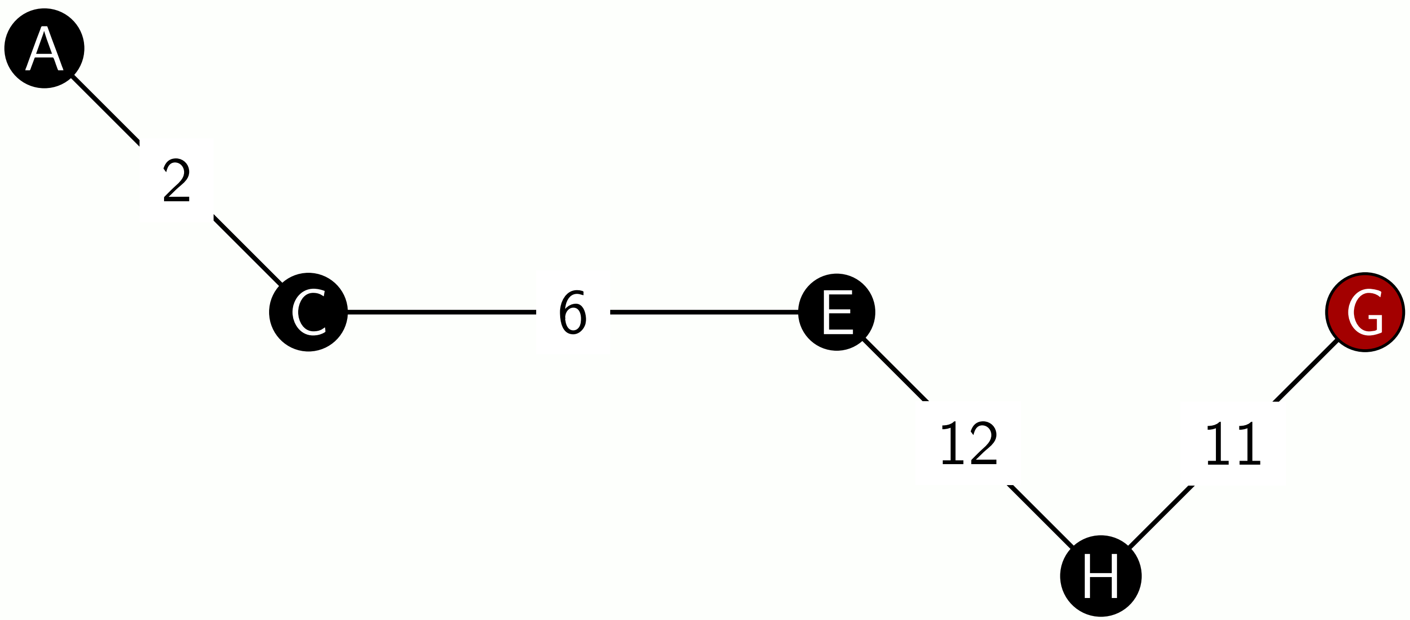
$E_E = 1, 3, 4, 7, 8, 5, 9$

Algorytm Fleury'ego



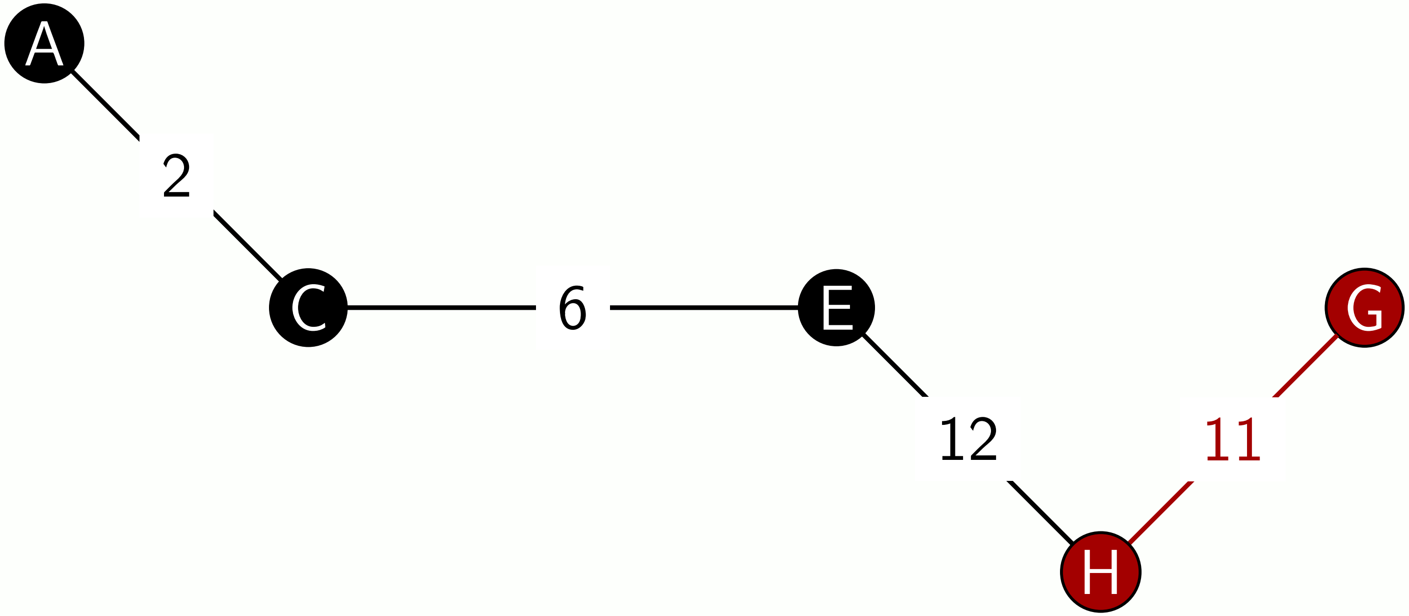
$$E_E = 1, 3, 4, 7, 8, 5, 9$$

Algorytm Fleury'ego



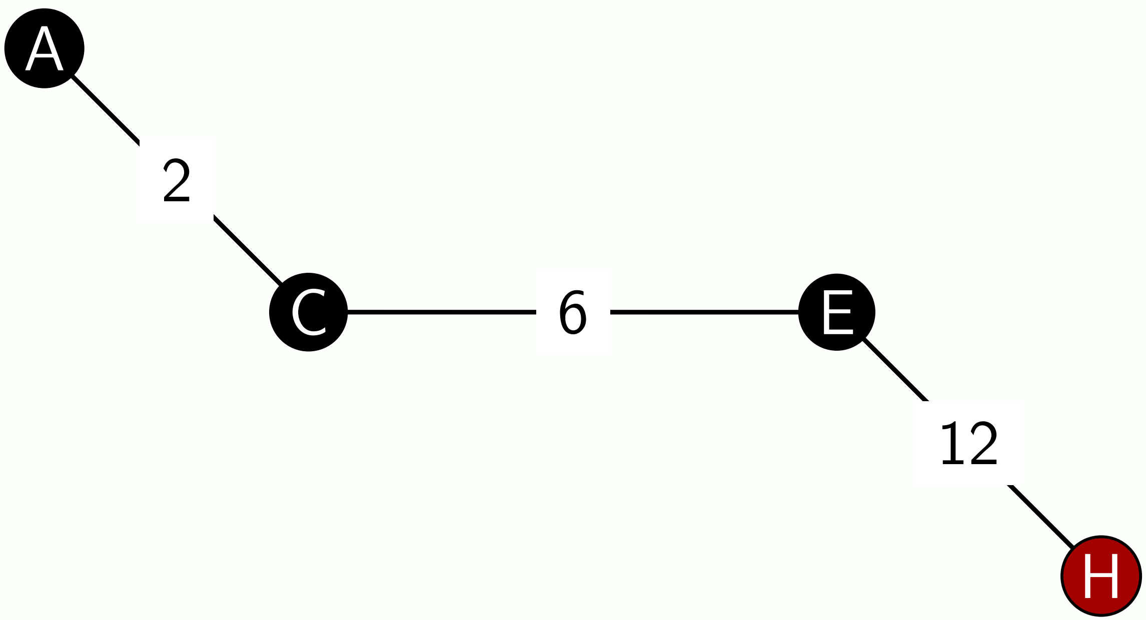
$E_E = 1, 3, 4, 7, 8, 5, 9, 10$

Algorytm Fleury'ego



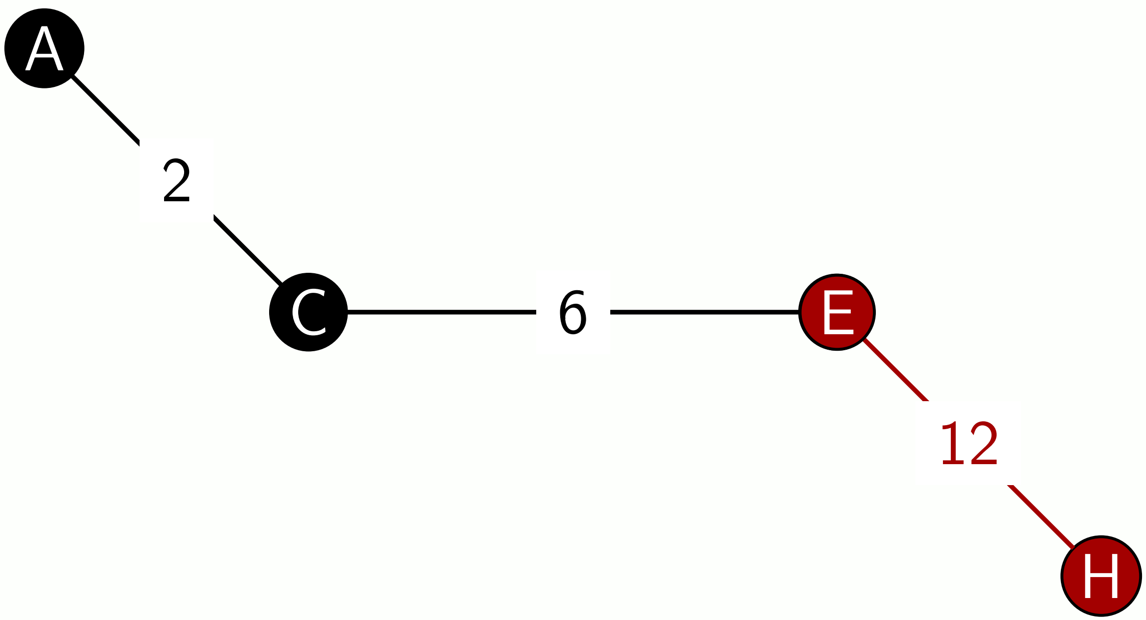
$$E_E = 1, 3, 4, 7, 8, 5, 9, 10$$

Algorytm Fleury'ego



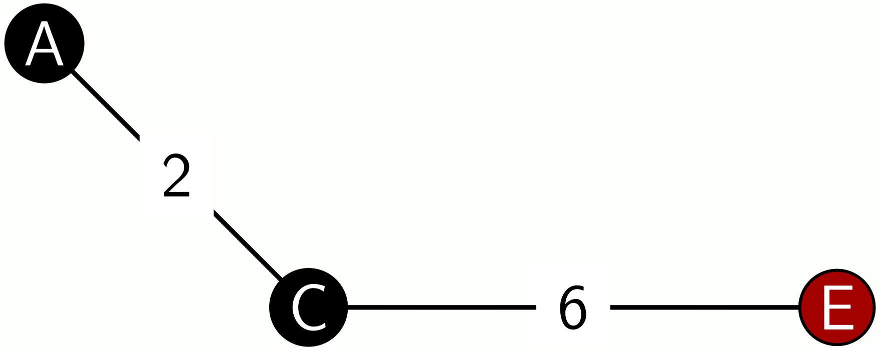
$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11$$

Algorytm Fleury'ego



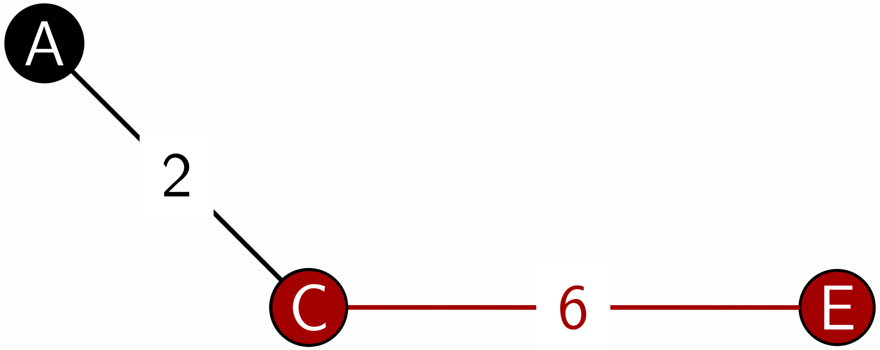
$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11$$

Algorytm Fleury'ego



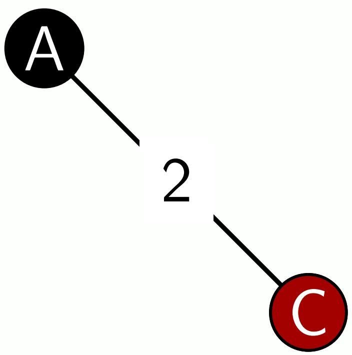
$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12$$

Algorytm Fleury'ego



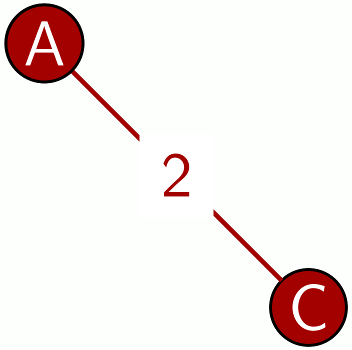
$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12$$

Algorytm Fleury'ego



$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12, 6$$

Algorytm Fleury'ego

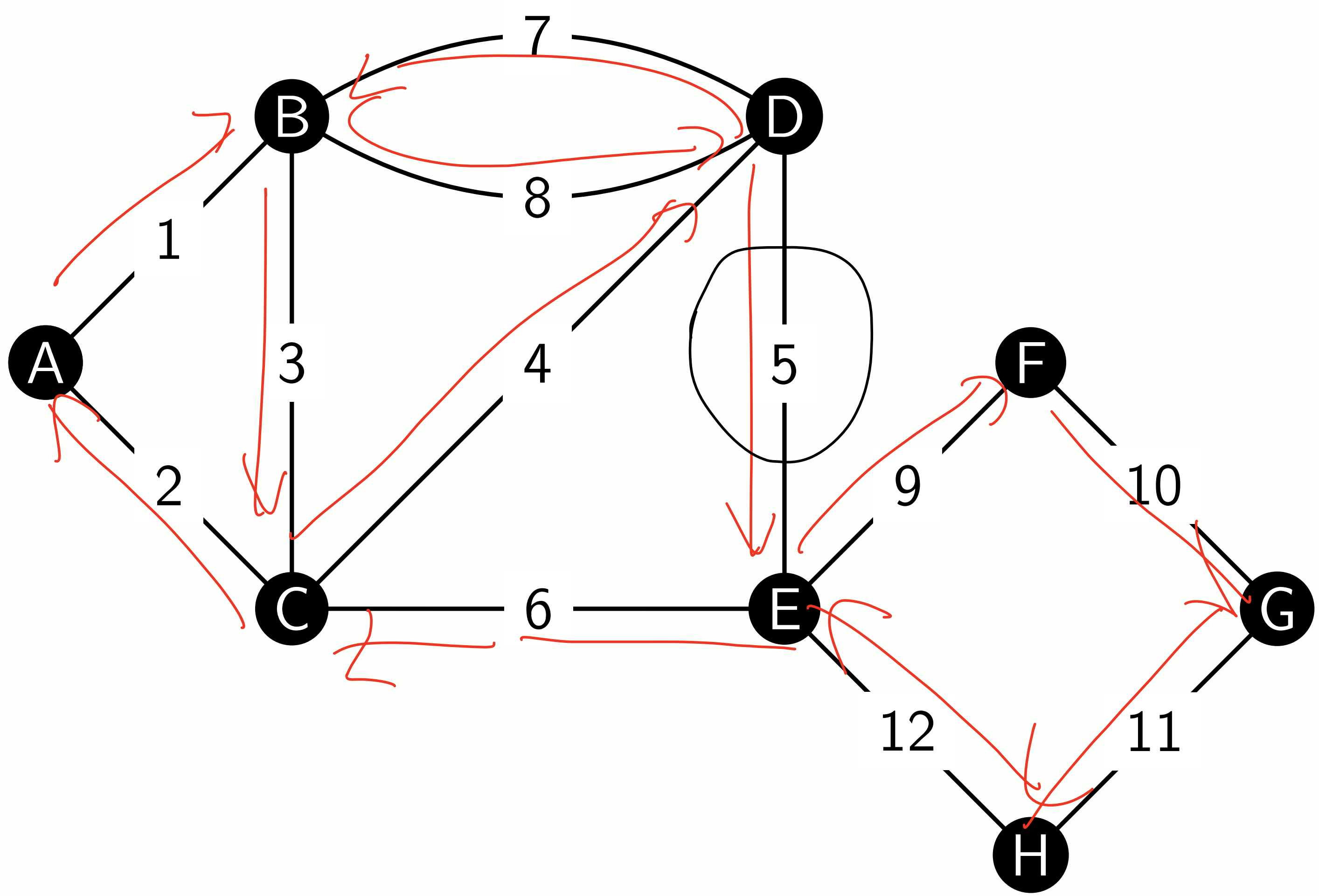


$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12, 6$$

Algorytm Fleury'ego

$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12, 6, 2$$

Algorytm Fleury'ego

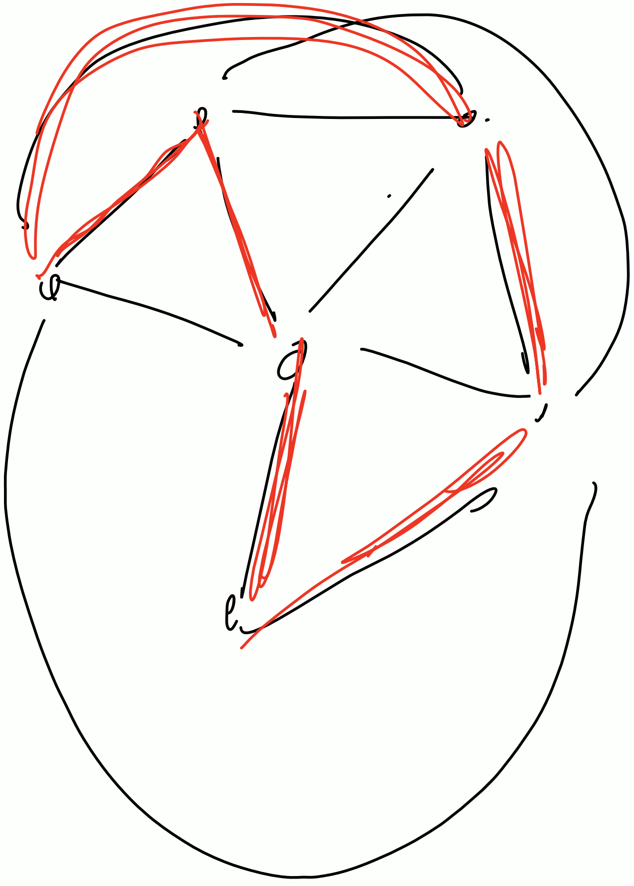


$$E_E = 1, 3, 4, 7, 8, 5, 9, 10, 11, 12, 6, 2$$

Grafy hamiltonowskie

⇝ **Cykl Hamiltona**: cykl zawierający wszystkie wierzchołki grafu.

⇝ **Graf hamiltonowski**: graf zawierający cykl Hamiltona.



Grafy hamiltonowskie

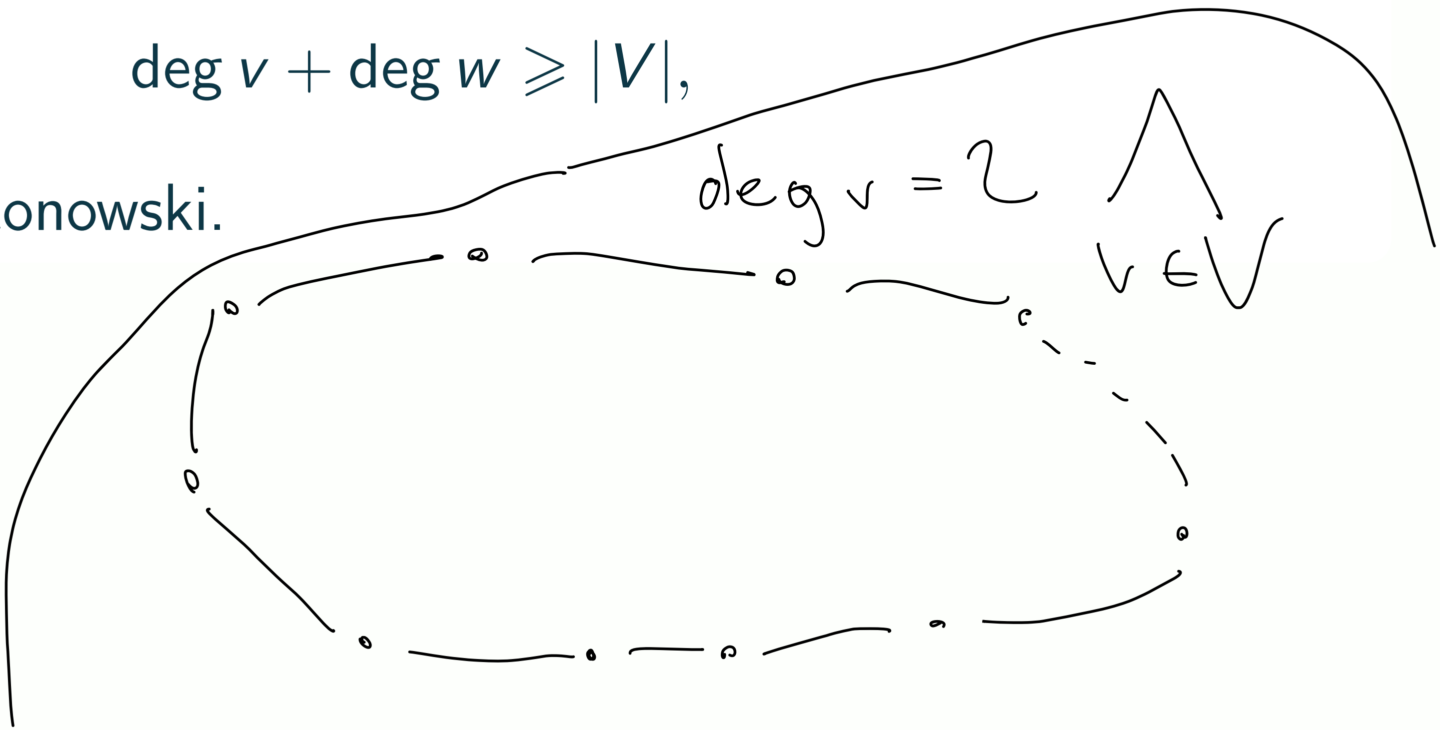
- **Cykl Hamiltona**: cykl zawierający wszystkie wierzchołki grafu.
- **Graf hamiltonowski**: graf zawierający cykl Hamiltona.

Twierdzenie Orego

Jeżeli graf $G = (V, E)$ ma przynajmniej trzy wierzchołki oraz dla każdej pary niesąsiednich wierzchołków v i w zachodzi

$$\deg v + \deg w \geqslant |V|,$$

to graf G jest hamiltonowski.

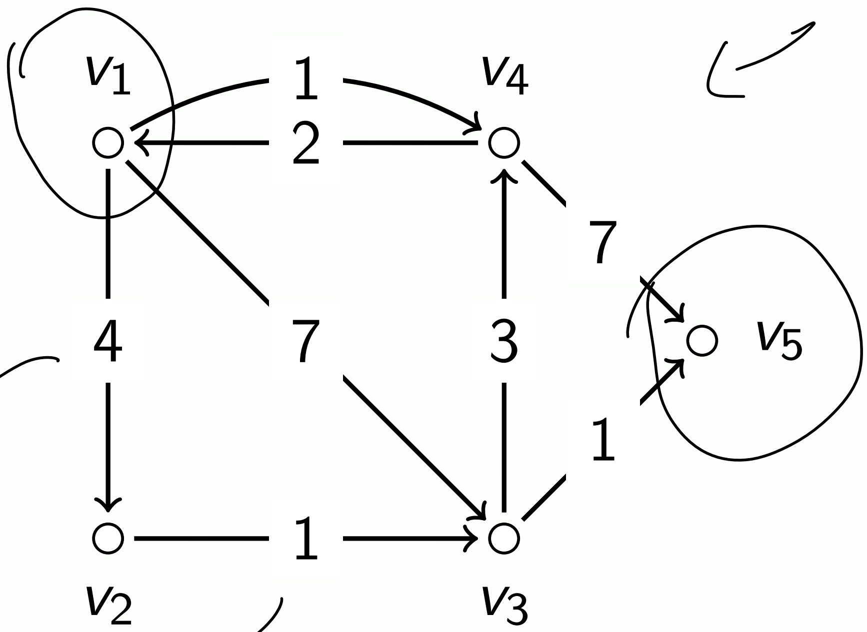


Poszukiwanie ścieżek

E
$\Downarrow$
$\{v, w\}$

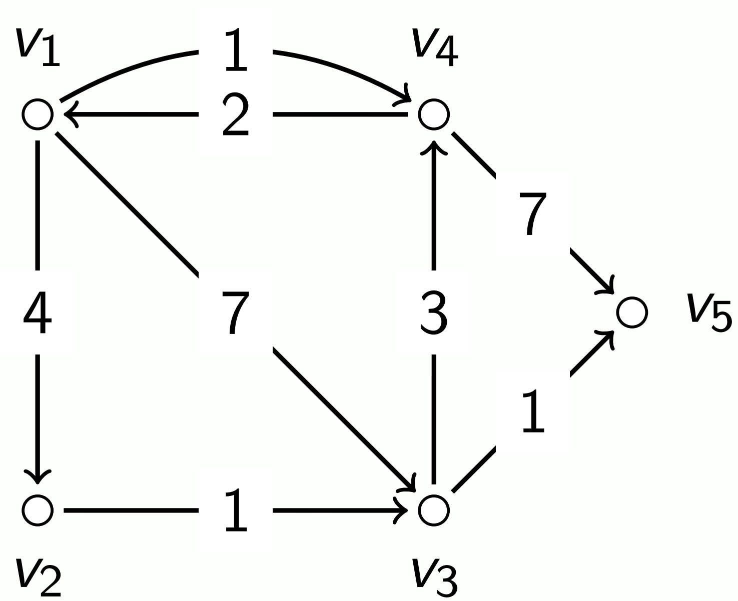
graf skierowany

E
$\Downarrow$
(v, w)



wagi
(koszt)

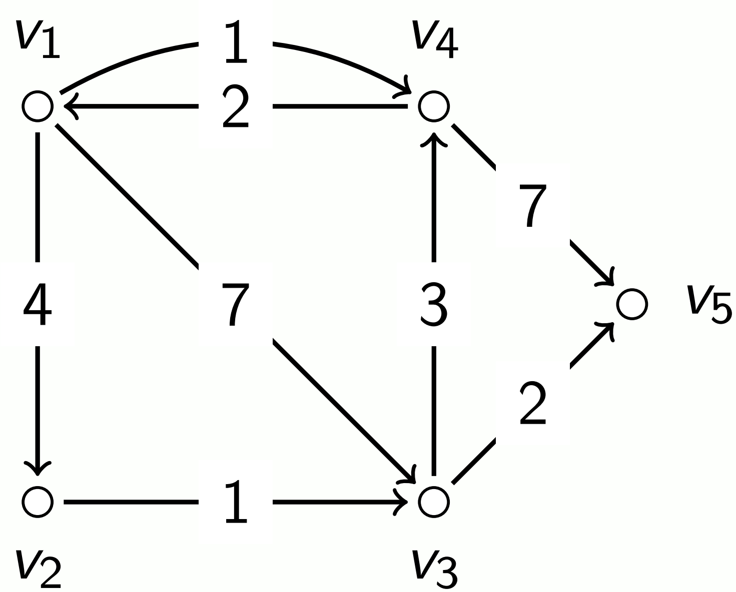
Poszukiwanie ścieżek



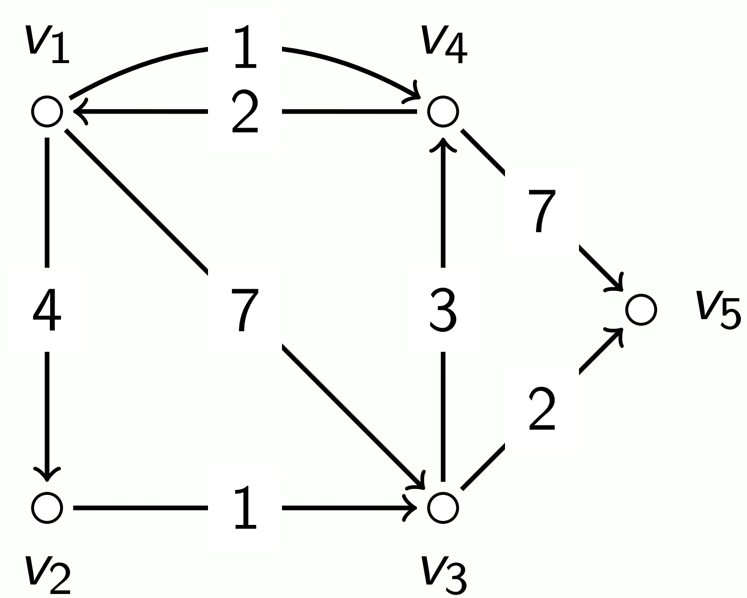
W	v_1	v_2	v_3	v_4	v_5
v_1	∞	4	7	1	∞
v_2	∞	$\cancel{1}$	1	∞	∞
v_3	∞	∞	∞	3	1
v_4	2	∞	∞	∞	7
v_5	∞	∞	∞	∞	∞

Algorytm Dijkstry

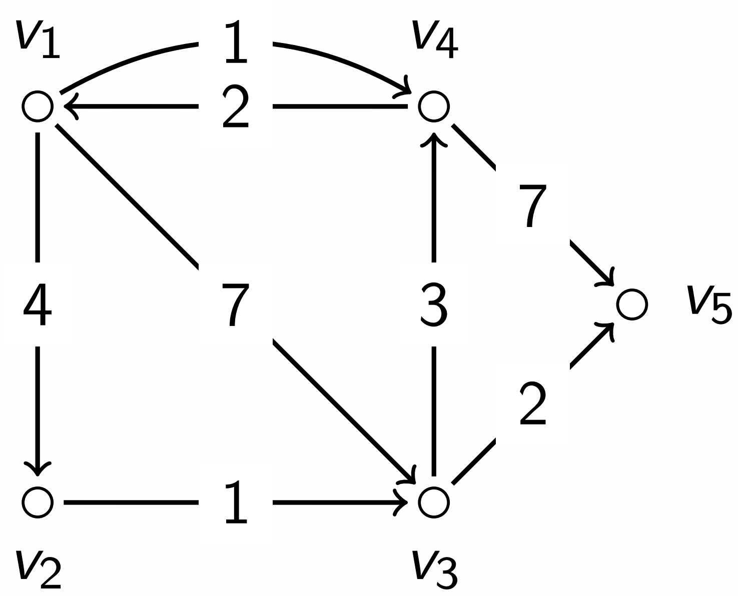
input: graf skierowany $G = (\{1, \ldots, n\}, E)$, wagi $W = W(v, w)$
output: długość D_j najkrótszej ścieżki od 1 do j, $j \in \{2, \ldots, n\}$
$L \leftarrow \emptyset$
$V \leftarrow \{2, \ldots, n\}$
for $i \in V$ **do**
    $D_i \leftarrow W(1, i)$
end for
while $V \setminus L \neq \emptyset$ **do**
    wybierz $k \in V \setminus L$ o najmniejszym $\cancel{D_j}$ D_k
    $L \leftarrow L \cup \{k\}$
    for $j \in V \setminus L$ **do**
        if $D_j > D_k + W(k, j)$ **then**
            $D_j \leftarrow D_k + W(k, j)$
        end if
    end for
end while



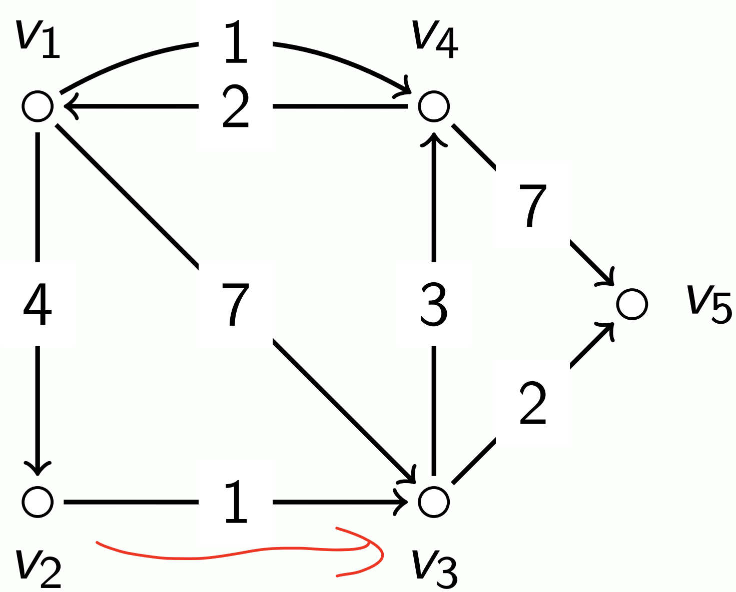
v_1
v_4
v_5
v_2
v_3
1
2
4
7
3
7
2
1



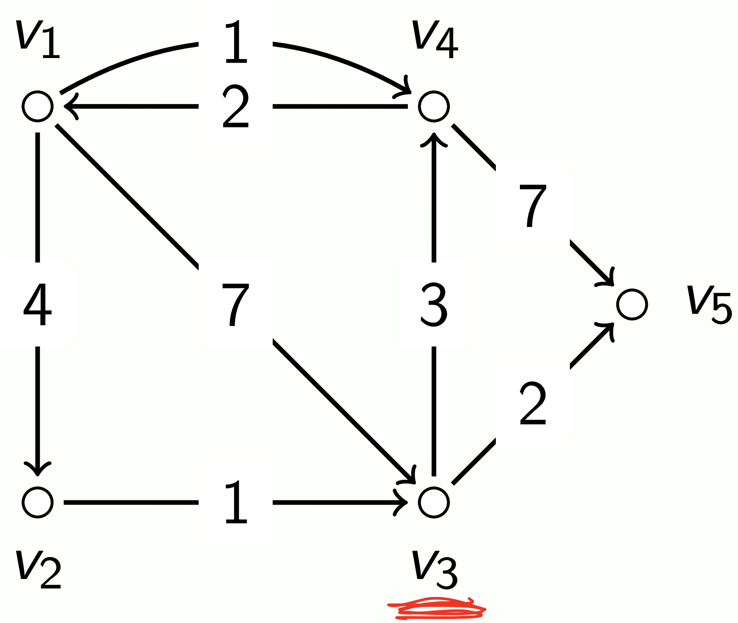
L	D_2	D_3	D_4	D_5



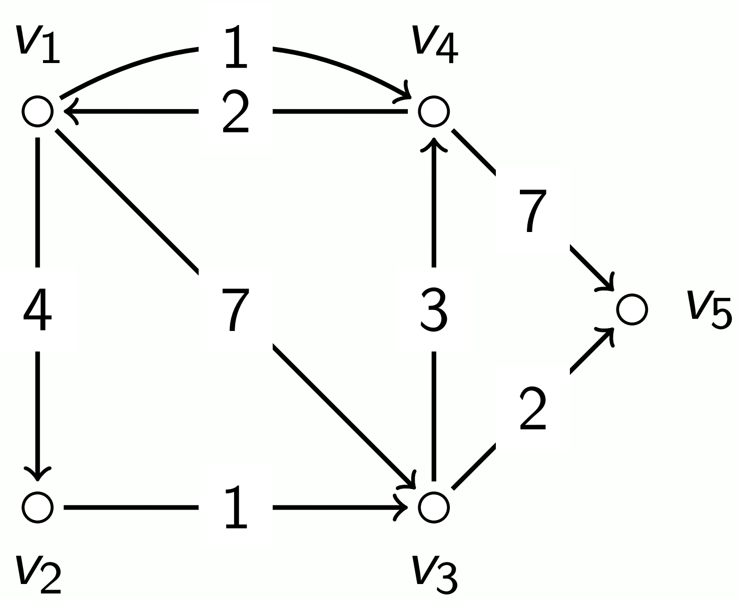
L	D_2	D_3	D_4	D_5
$\emptyset$	4	7	1	∞
$\{4\}$	4	7	1	8



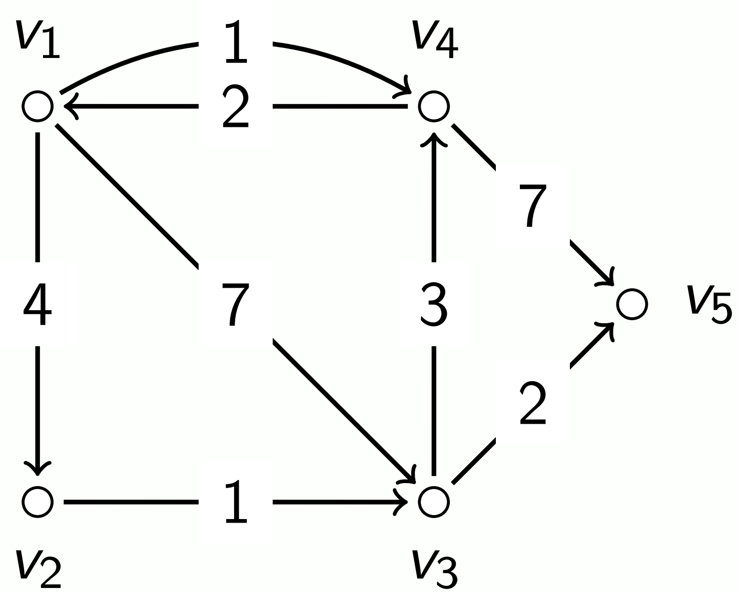
L	D_2	D_3	D_4	D_5
$\emptyset$	4	7	1	
$\{4\}$	4	7	1	8
$\{2,4\}$		5		



L	~~D_2~~	D_3	~~D_4~~	D_5
$\emptyset$	4	7	1	
$\{4\}$	4	7	1	8
$\{2,4\}$	4	5	1	8
				7



L	D_2	D_3	D_4	D_5
$\emptyset$	4	7	1	
$\{4\}$	4	7	1	8
$\{2,4\}$	4	5	1	8
$\{2,3,4\}$	4	5	1	6



L	D_2	D_3	D_4	D_5
$\emptyset$	4	7	1	
$\{4\}$	4	7	1	8
$\{2,4\}$	4	5	1	8
$\{2,3,4\}$	4	5	1	6
$\{2,3,4,5\}$	4	5	1	6

Algorytm Dijkstry

```
input: G = ({1,2,...,n},E), W = W(v,w)
output: D(j)
L := ∅, V := {2,...,n}
for i ∈ V do D(i) := W(1,i)
# 1. dla i w L, D(i) jest dlugoscia najkrotszej
#    sciezki 1->i
# 2. dla i ∉ L, D(i) jest dlugoscia najkrotszej
#    sciezki 1->i w L
while V \ L ≠ ∅ do
    wybierz k ∈ V \ L o minimalnym D(k)
    L := L ∪ {k}
    for j ∈ V \ L do
        if D(j) > D(k) + W(k,j) then
            D(j) := D(k) + W(k,j)
```