

Chinese Postman Problem

Chinese Postman Problem

- This is the **Chinese Postman Problem**, named in honor of the Chinese mathematician Guan Meigu [1962], who proposed it.
- A mail carrier must traverse **all edges** in a road network, starting and ending at the Post Office.
- The edges has **nonnegative weights** representing **distance or time**.
- We seek a **closed walk** of **minimum total length** that uses **all the edges exactly once**.

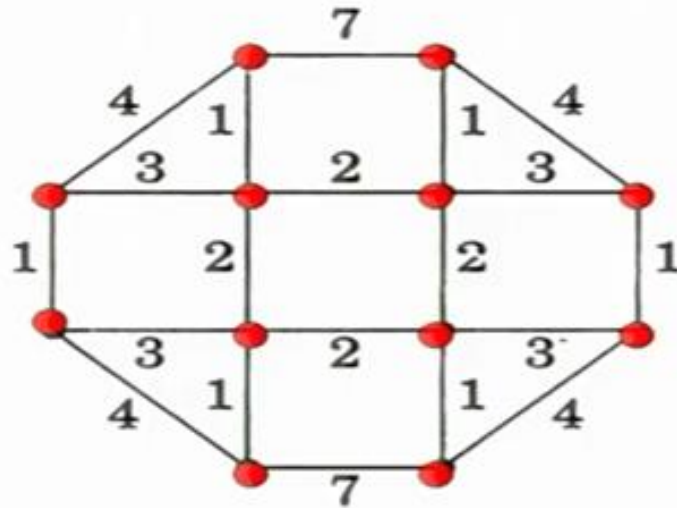
Eulerian cycle

- **If every vertex is even**, then the graph is **Eulerian** and the answer is the **sum of the edge weights**.
- Otherwise, **we must repeat edges**.
- Every traversal is an Eulerian circuit of a graph obtained by **duplicating edges**.
- Finding the **shortest traversal** is **equivalent** to finding the minimum **total weight of edges whose duplication will make all vertex degrees even**

- We say **duplication** because we need not use an edge more than twice.
- There may be **many ways to choose the duplicated edges..**

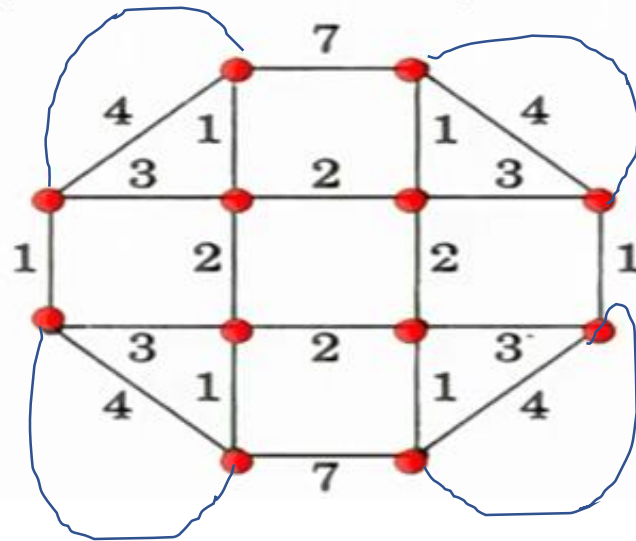
Examples

- In the example below, the eight outer vertices have odd degree. If we match them around the outside to make the degrees even, the extra cost is $4 + 4 + 4 + 4 = 16$ or $1 + 7 + 7 + 1 = 16$. We can do better by using all the vertical edges, which total only 10.



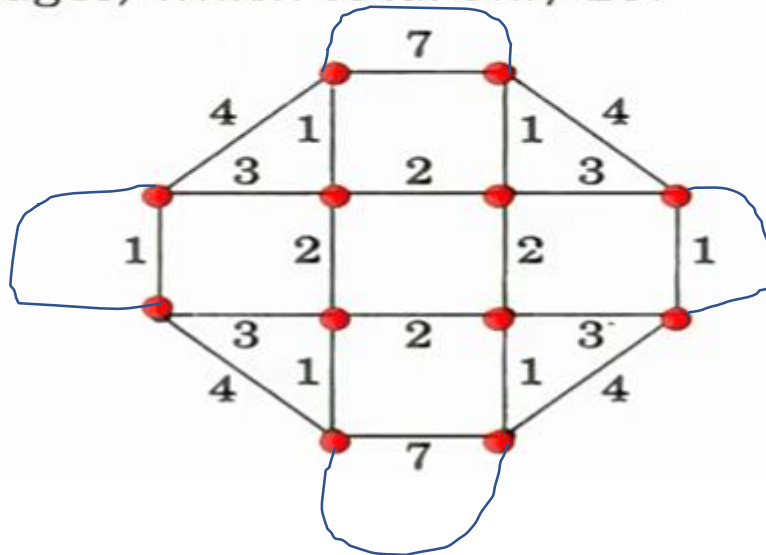
Examples

- In the example below, the eight outer vertices have odd degree. If we match them around the outside to make the degrees even, the extra cost is $4 + 4 + 4 + 4 = 16$ or $1 + 7 + 7 + 1 = 16$. We can do better by using all the vertical edges, which total only 10.



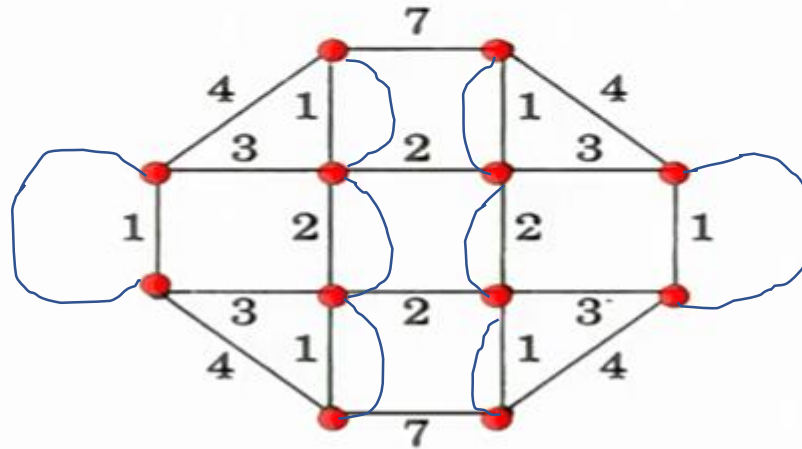
Examples

- In the example below, the eight outer vertices have odd degree. If we match them around the outside to make the degrees even, the extra cost is $4 + 4 + 4 + 4 = 16$ or $1 + 7 + 7 + 1 = 16$. We can do better by using all the vertical edges, which total only 10.



Examples

- In the example below, the eight outer vertices have odd degree. If we match them around the outside to make the degrees even, the extra cost is $4 + 4 + 4 + 4 = 16$ or $1 + 7 + 7 + 1 = 16$. We can do better by using all the vertical edges, which total only 10.



Chinese Postman Problem

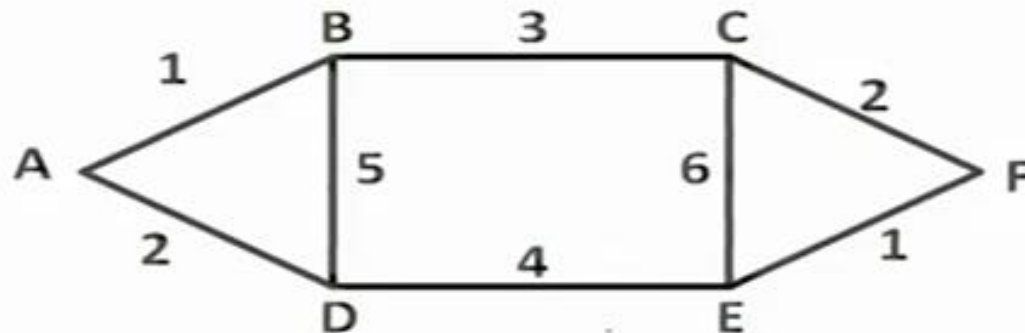
- **Edmonds and Johnson** described a way to solve the Chinese Postman Problem.
- If there are only two odd vertices, then
 - We can use Dijkstra's Algorithm to find the shortest path between them and solve the problem
- If there are $2k$ odd vertices, then
 - Use **Dijkstra's algorithm** to find the shortest paths connecting each pair of odd vertices
 - Use these lengths as weights on the edges of K_{2k}
 - Find the minimum total weight of k edges that pair up these $2k$ vertices.

Edmonds and Johnson Algorithm

- Step 1: List all odd vertices.
- Step 2: List all possible pairings of odd vertices.
- Step 3: For each pairing find the edges that connect the vertices with the minimum weight.
- Step 4: Find the pairings such that the sum of the weights is minimized.
- Step 5: On the original graph **add the edges that have been found in Step 4.**
- Step 6: The length of an optimal Chinese postman route is the sum of all the edges added to the total found in Step 4.
- Step 7: A route corresponding to this minimum weight can then be easily found --- Fleury's algorithm

Edmonds and Johnson Algorithm

- A mail carrier has to deliver mail to all streets in a subdivision. The mail carrier has to start and end at point A, where the post office is. The weight of each edge represents the length of each street. What is the most efficient route for the mail carrier?



Edmonds and Johnson Algorithm

Step 1 The odd vertices are BCDE

Step 2 There are three ways of pairing these odd vertices

Step 3 The shortest way of joining pairs is using the paths.

Odd Nodes

BCDE

Pairs

BC-DE

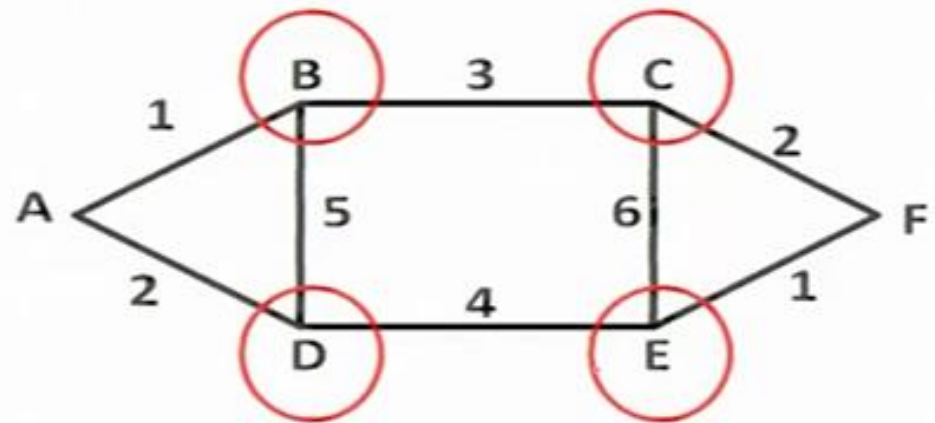
3 4 7

BD*-CE*

3 3 6

BE*-CD*

6 6 12



Edmonds and Johnson Algorithm

Step 1 The odd vertices are BCDE ✓

Step 2 There are three ways of pairing these odd vertices

Step 3 The shortest way of joining pairs is using the paths.

Odd Nodes

BCDE

Pairs

BC-DE ①

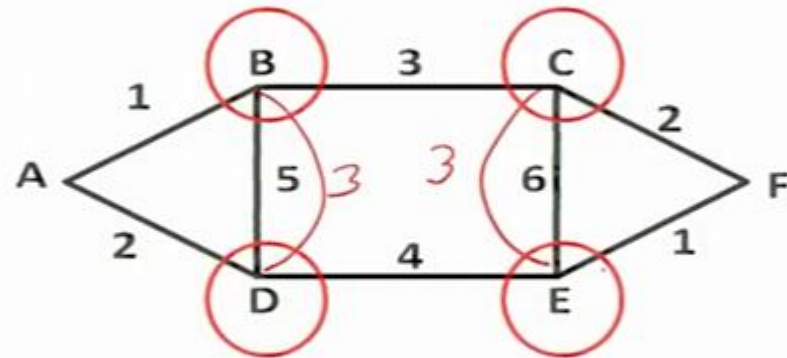
3 4 7

BD*-CE* ②

3 3 6

BE*-CD* ③

6 6 12



Edmonds and Johnson Algorithm

Step 1 The odd vertices are BCDE ✓

Step 2 There are three ways of pairing these odd vertices

Step 3 The shortest way of joining pairs is using the paths.

Odd Nodes

BCDE

Pairs

BC-DE ①

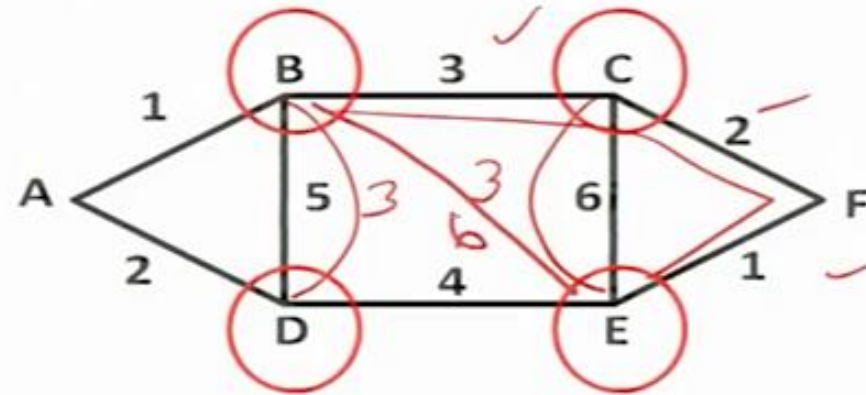
3 4 7 ✓

BD*-CE* ②

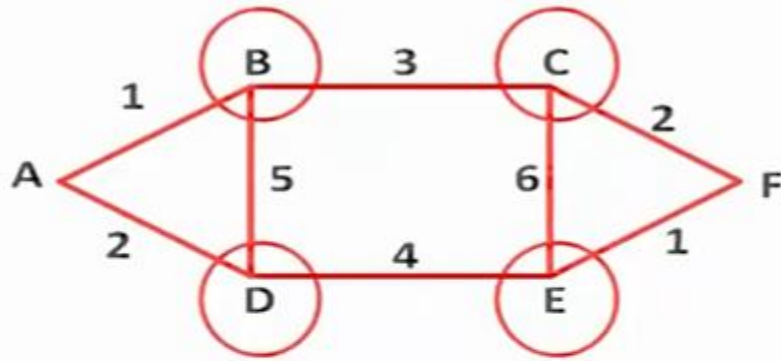
3 3 6 ✓

BE*-CD* ③

6 6 12 ✓



Edmonds and Johnson Algorithm



Find a route round all points
from A with least cost

Smallest + Total

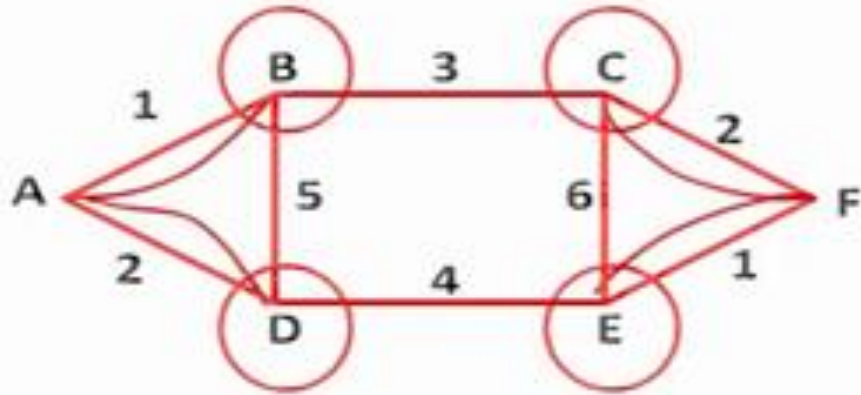
$$6 + 24 = 30$$

A B C F C E F E D B A D A

Step 4 Draw these edges onto
the graph

Step 5 The length of the
optimal Chinese postman
route is the sum of all the
edges in the original graph,
which is 24, plus the answer
found in step 4, which is 6. So
the length is 30.

Edmonds and Johnson Algorithm



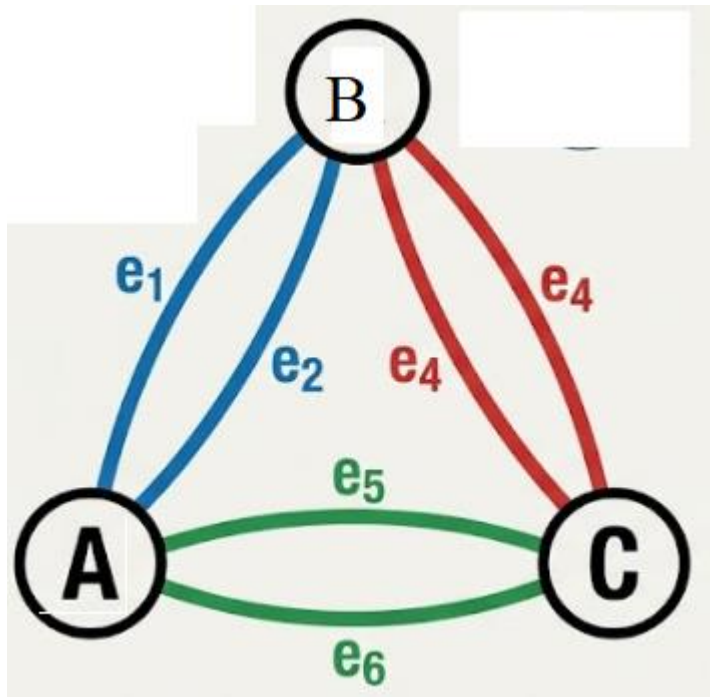
Find a route round all points
from A with least cost
Smallest + Total
 $6 + 24 = 30$

A B C F C E F E D B A D A

Step 4 Draw these edges onto
the graph ✓

Step 5 The length of the
optimal Chinese postman
route is the sum of all the
edges in the original graph,
which is 24, plus the answer
found in step 4, which is 6. So
the length is 30.

Fleury's Algorithm



Valid for parallel edges too --- follow exactly the same procedure as for the simple graph

Fleury's Algorithm

Step 1

Choose the starting vertex.

- Start at any vertex (say, A).

Step 2

Among all edges incident on the current vertex (A),

- Prefer an edge that **is not a bridge** – doesn't disconnect G to two components
- Use a bridge only when it is the only available edge.

Step 3

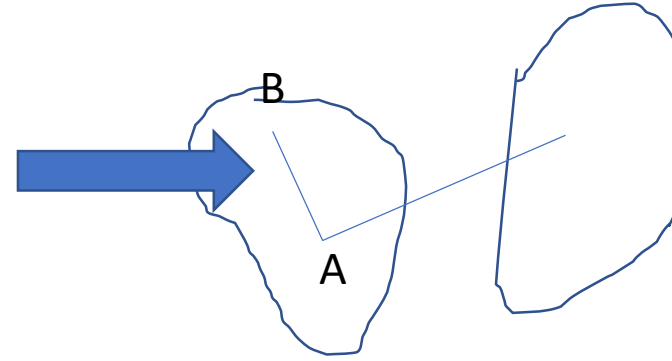
Traverse the selected edge (AB).

- Add it to the Euler tour.
- Delete the edge from the graph.

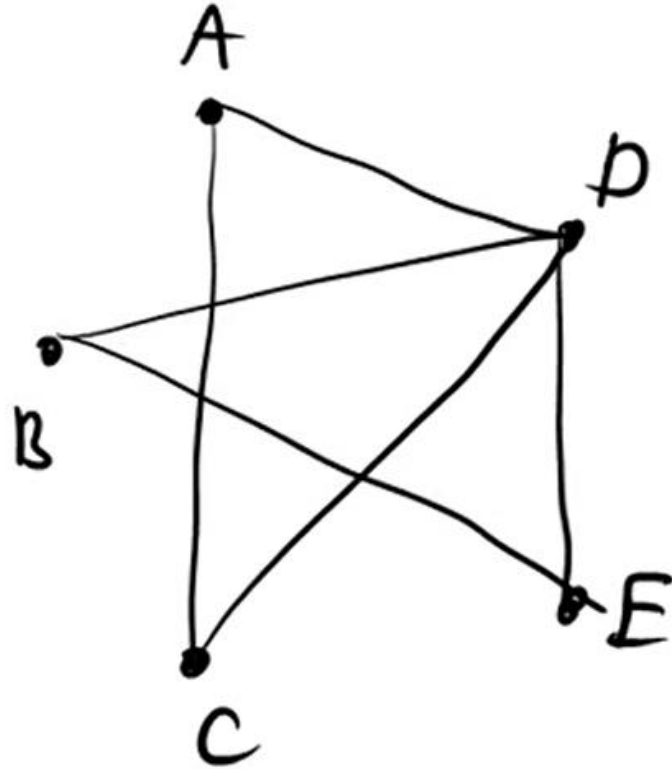
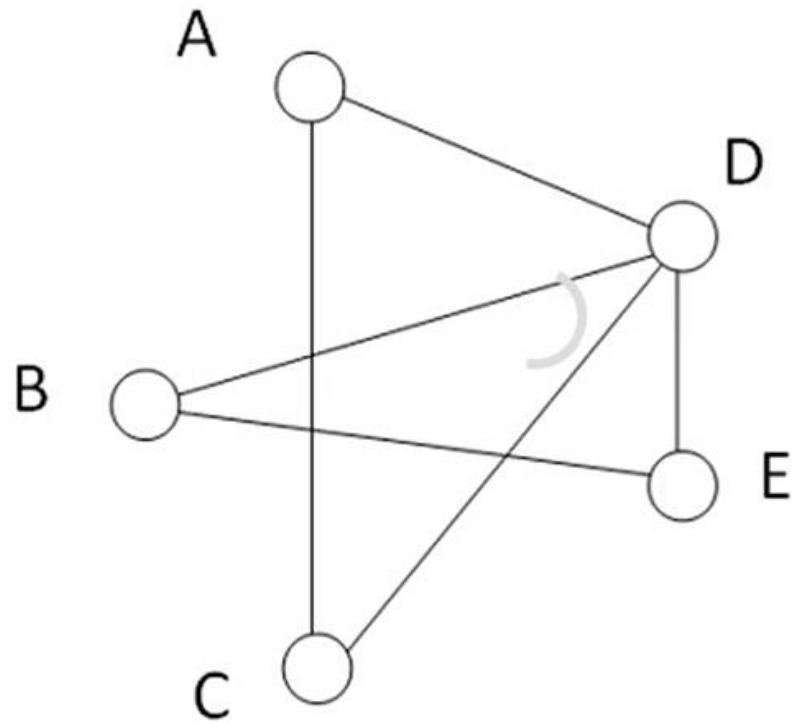
Step 4

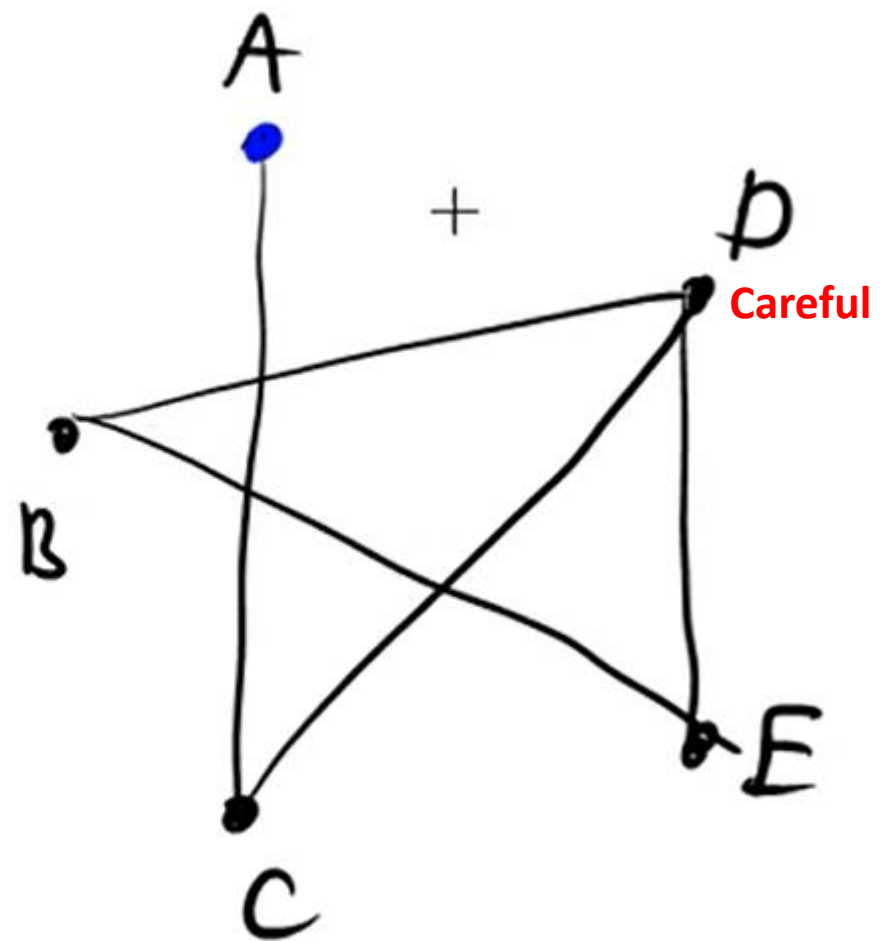
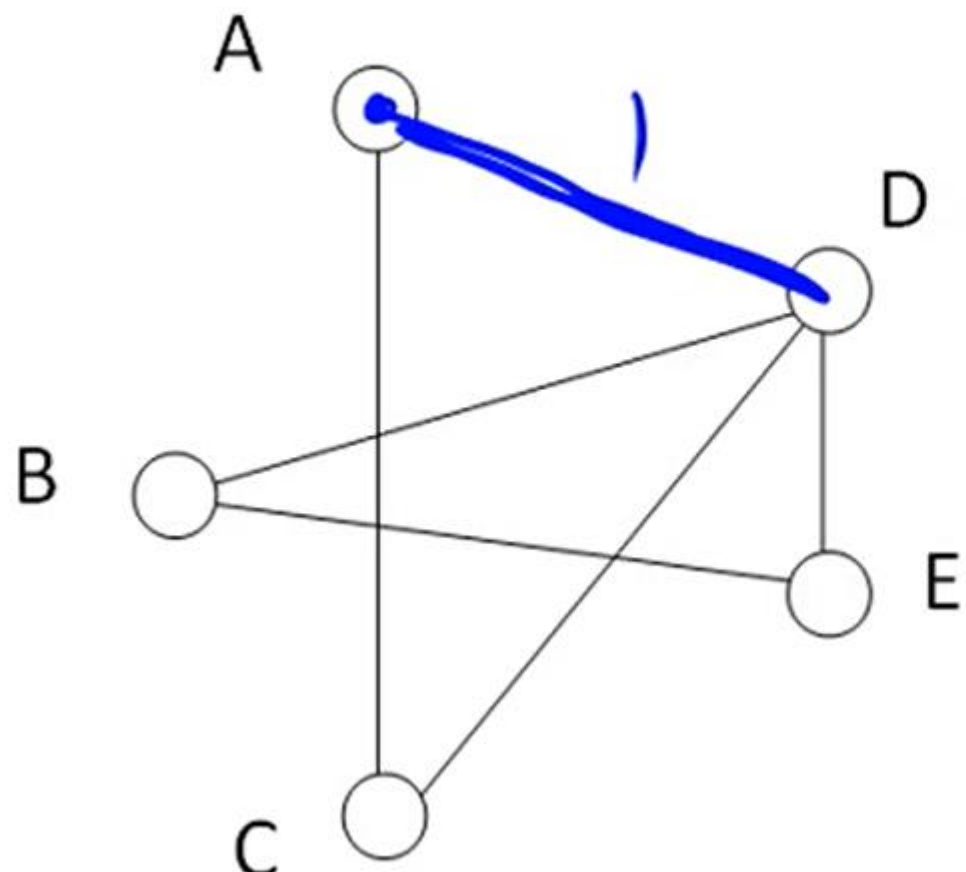
Move to the adjacent vertex (B).

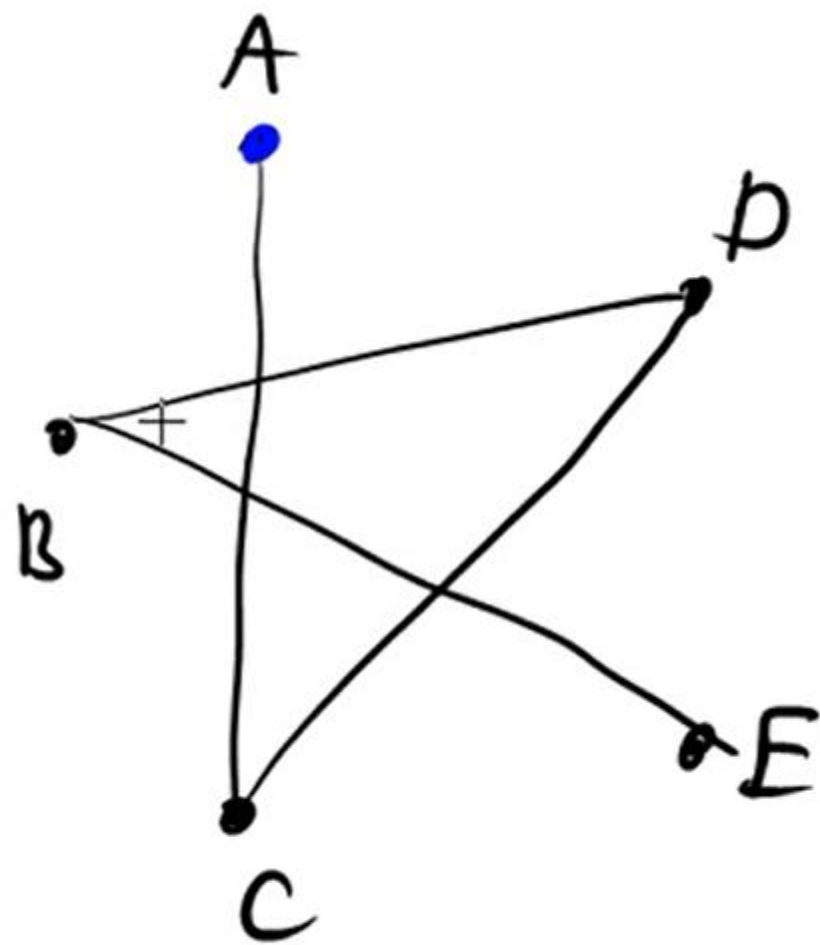
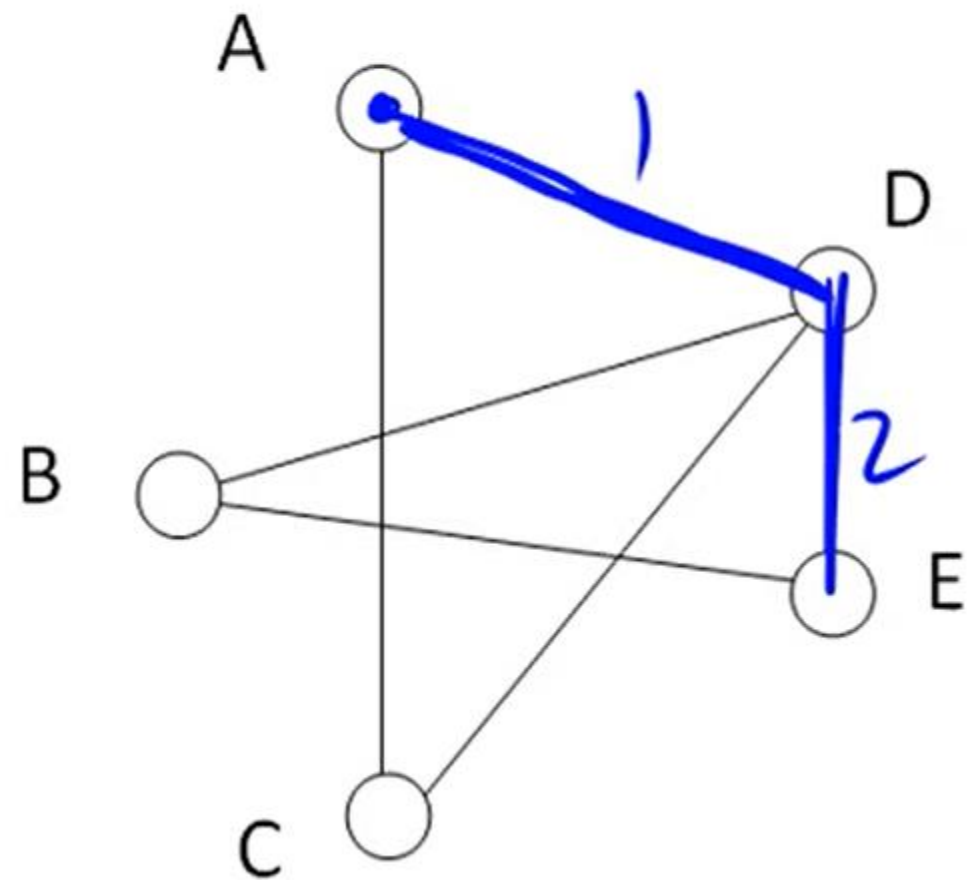
Repeat Steps 2–4 until all edges are used exactly once.

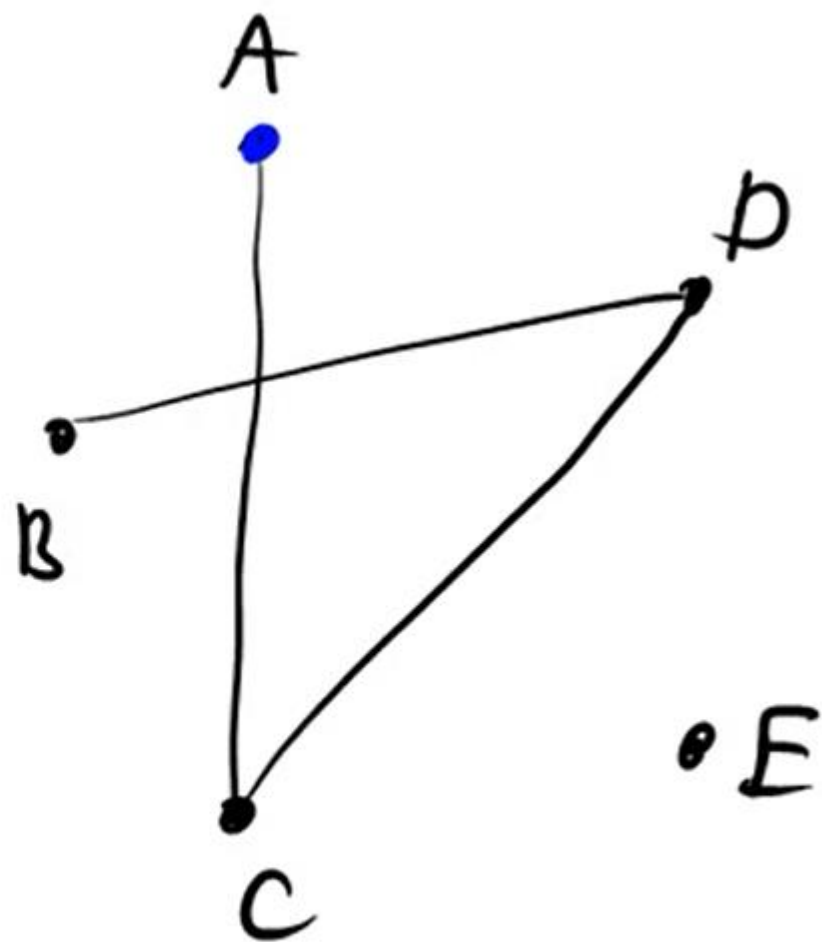
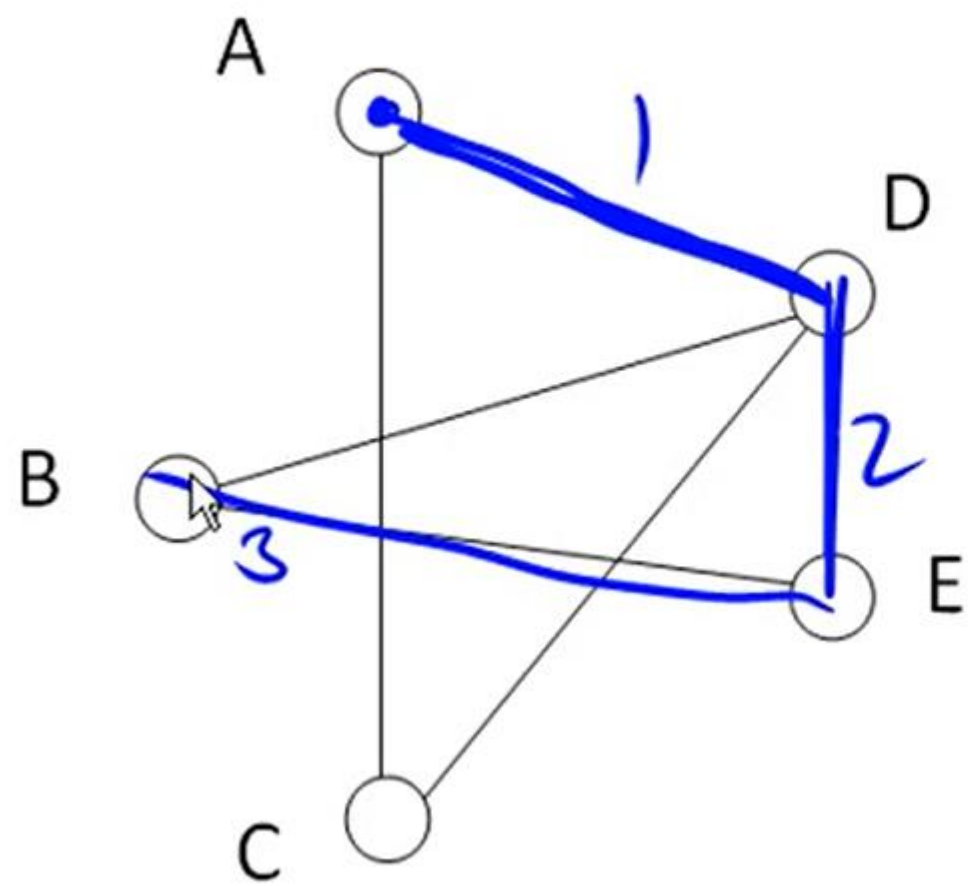


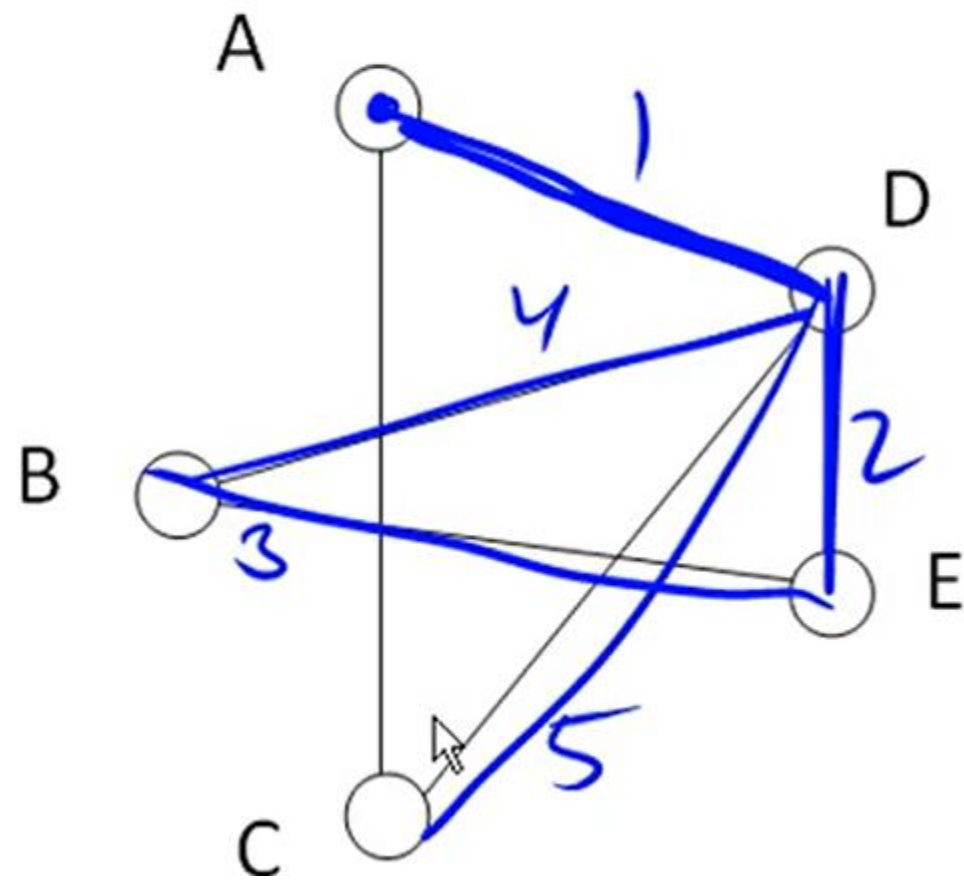
Fleury's Algorithm











B



b

E