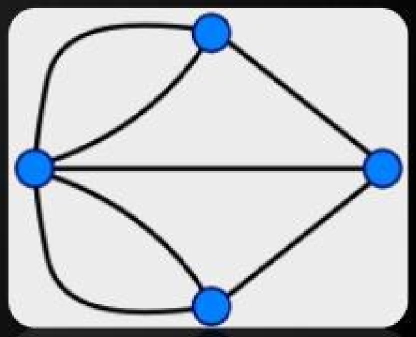
Telerik Academy

Euler Graphs

Overview of algorithms for finding Eulerian cycles and paths in connected graphs



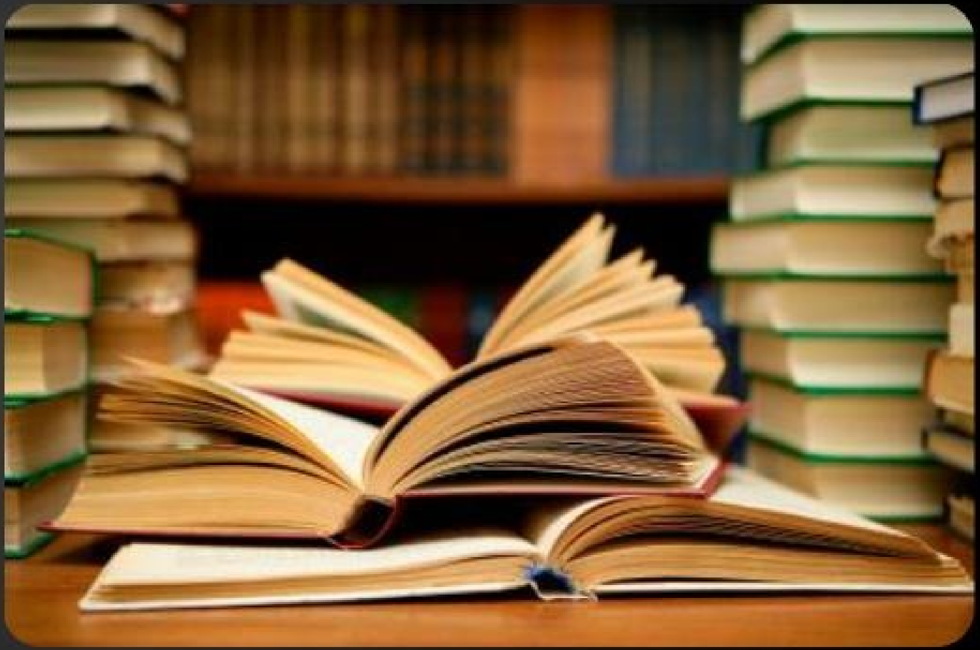
Deyan Yosifov

Telerik Corporation

http://deyan-yosifov.com

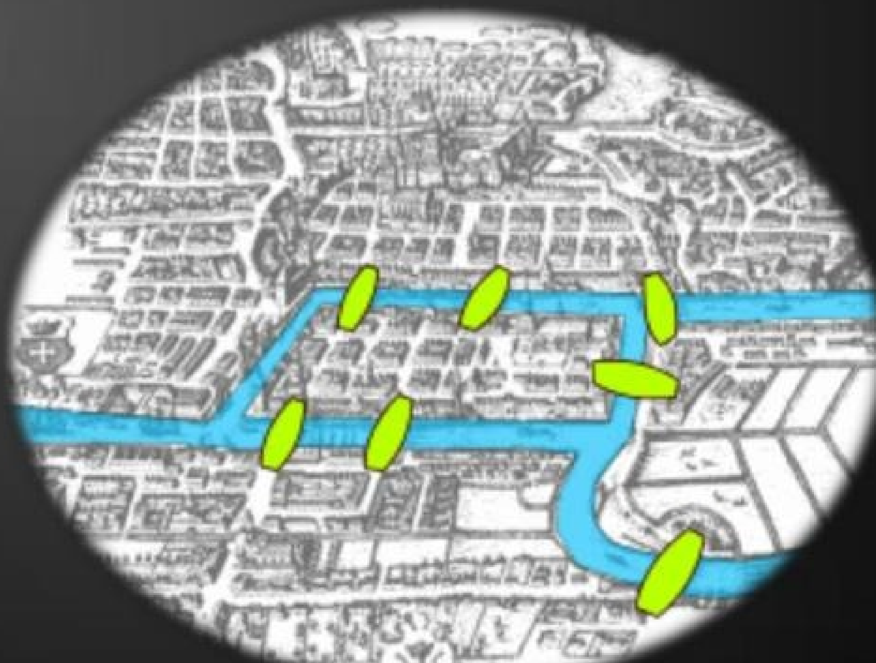
Table of Contents

1. Defining the problem
2. Euler's theorem
3. Algorithms:
   - Fleury's
   - Hierholzer's
4. Resources
5. Questions



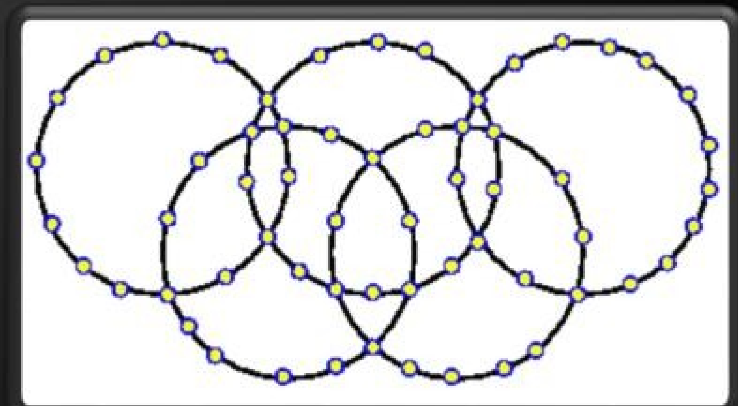
Telerik Academy

Defining the problem

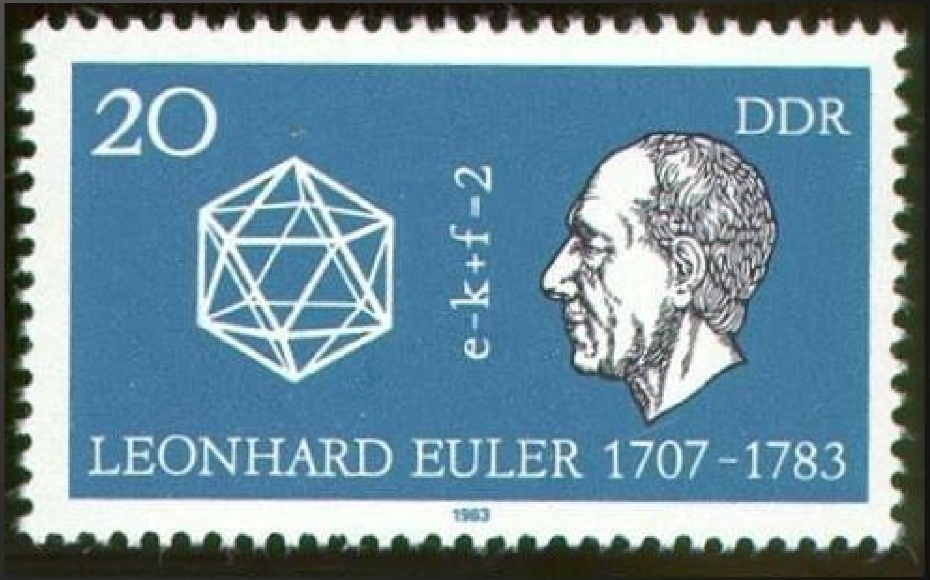


Definitions

- In graph theory Euler path is a path that visits each node from the graph exactly once.
- Euler cycle is a Euler path that starts and ends with the same node.
- Euler graph is a graph with graph which contains Euler cycle.

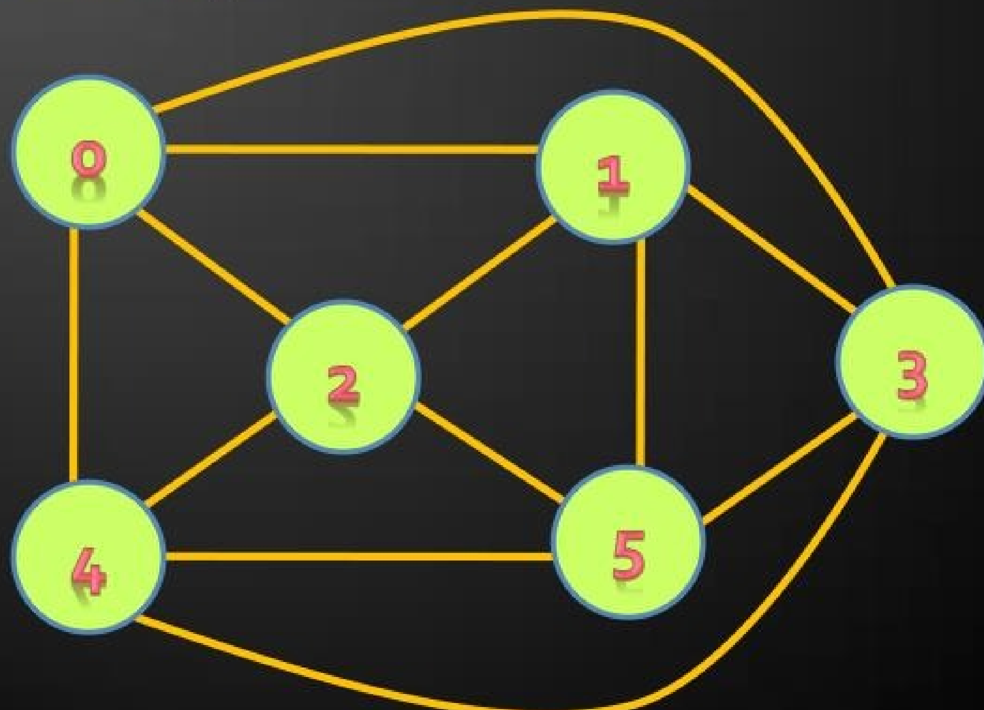


Euler's theorem



Euler's theorem

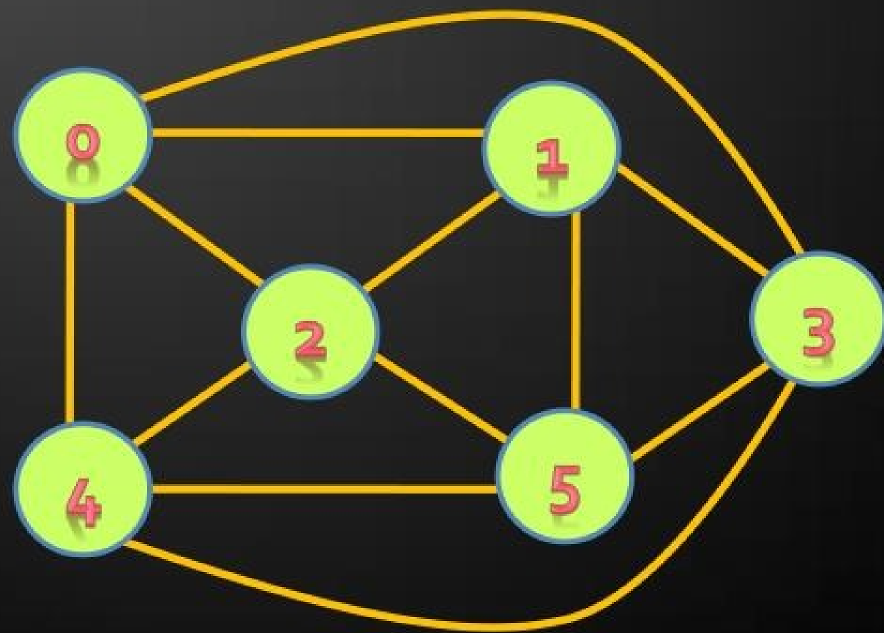
- Connected undirected graph is Euler graph if and only if every node in the graph is of even degree (has even number of edges starting from that node).



Euler's theorem – half proof

- Let's first assume we have Euler cycle in some connected graph.
  - For the example below:

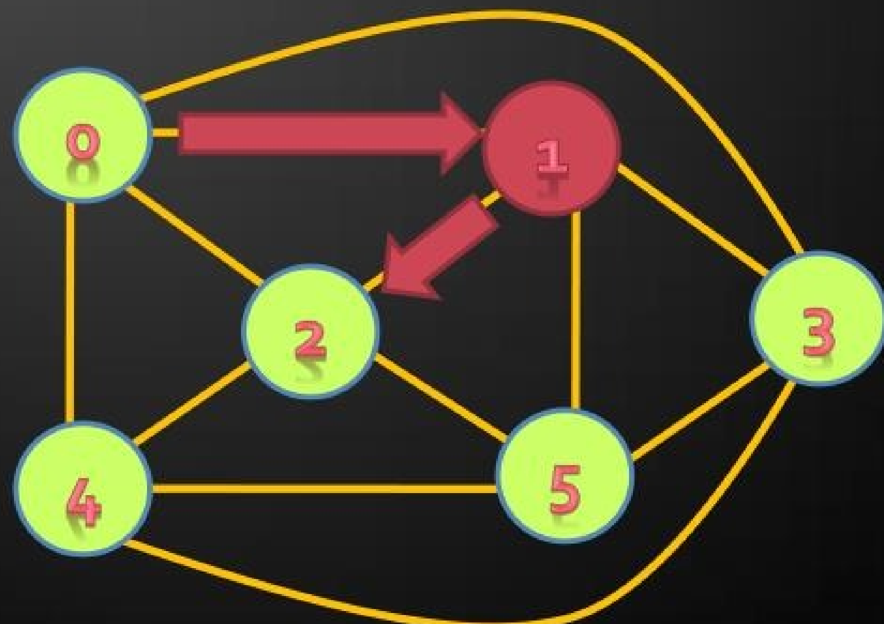
```
0 -> 1 -> 2 -> 0 -> 3 -> 1 -> 5- > 2 -> 4 -> 3 -> 5 -> 4 -> 0
```



Euler's theorem – half proof

- Once we "enter" a node, right afterwards we "exit" from it !
  - => on each visit we add +2 to node's degree

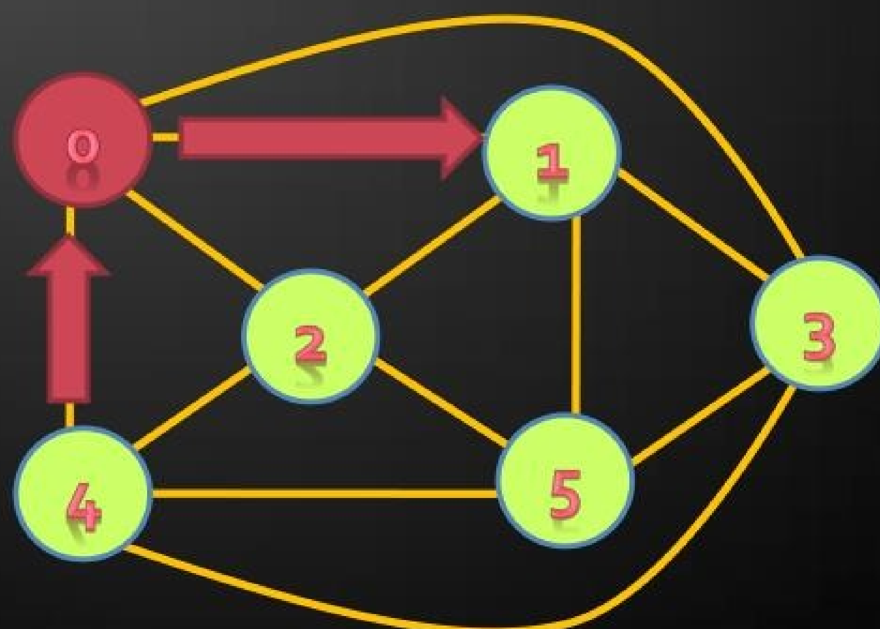
```
0 -> 1 -> 2 -> 0 -> 3 -> 1 -> 5- > 2 -> 4 -> 3 -> 5 -> 4 -> 0
```



Euler's theorem – half proof

- For the first node at the beginning we add +1 to its degree and in the end again we add +1
  - => All nodes have even degree!

```
0 -> 1 -> 2 -> 0 -> 3 -> 1 -> 5- > 2 -> 4 -> 3 -> 5 -> 4 -> 0
```



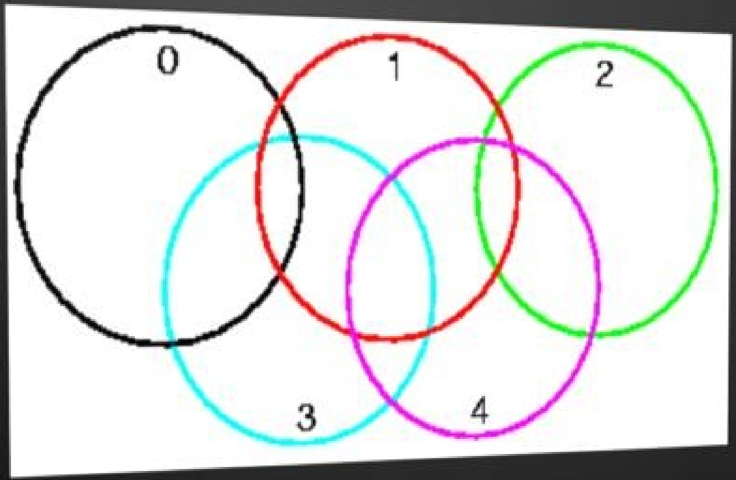
Telerik Academy

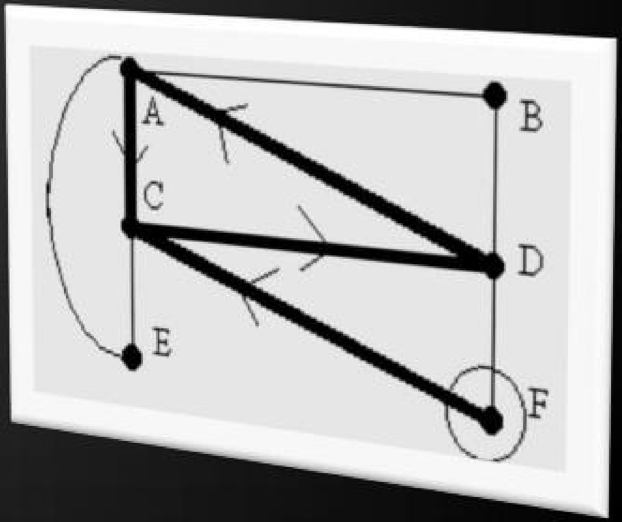
Euler's theorem – properties

- Undirected connected graph has Euler's path:
  - <=> there are exactly 2 nodes with odd degree
- Directed connected graph has Euler' cycle if and only if d+ = d- for every node.
  - d+ and d- are the number of edges in and out of the node
- Directed connected graph has Euler's path if:
  - there is exactly one node with d+ = 1 + d-
  - and there is exactly one node with d- = 1 + d+

Telerik Academy

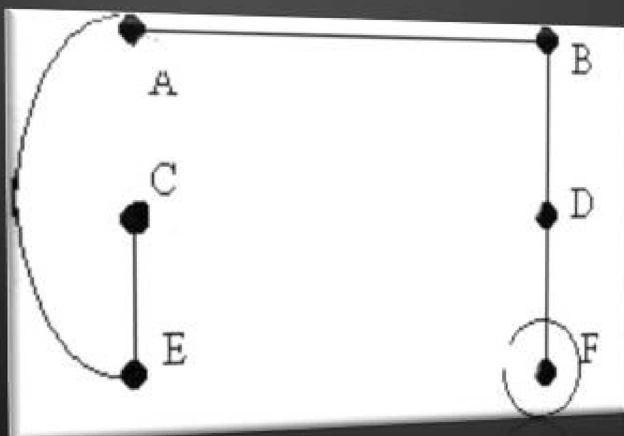
Algorithms

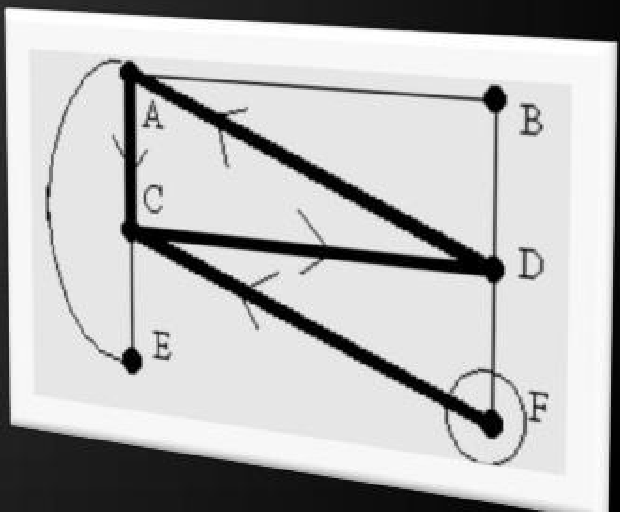




Telerik Academy

Fleury's algorithm





Fleury's algorithm

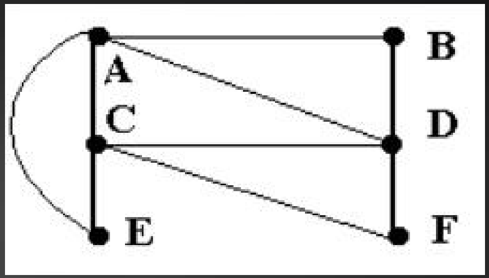
- Input: A connected graph *G = (V, E) with no vertices of odd degree*
- Output: A sequence *P of vertices and their connecting edges indicating the Euler circuit.*

```
1 Choose any vertex u0 in G.
2 P = u0
3 if Pi = u0e1u1e2…eiui choose edge ei+1 so that
      1. ei+1 is adjacent to ei
      2. Removal of ei+1 does not disconnect G – {e1,…,
      ei} (i.e. ei+1 is not a bridge).
4 if P includes all edges in E return P
5 goto 3
```

Fleury's algorithm example (1/3)

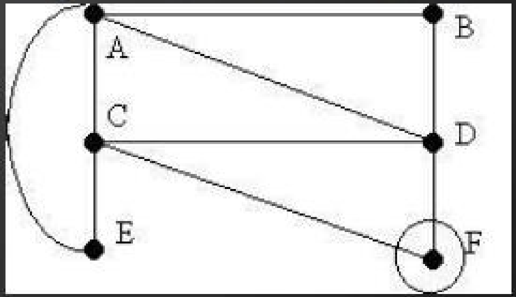
We start with the following graph

Start



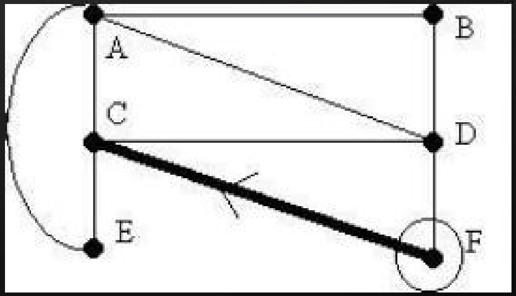
1. Choose any vertex (for example F)

Step 1



2. Travel from F to C (arbitrary choice).

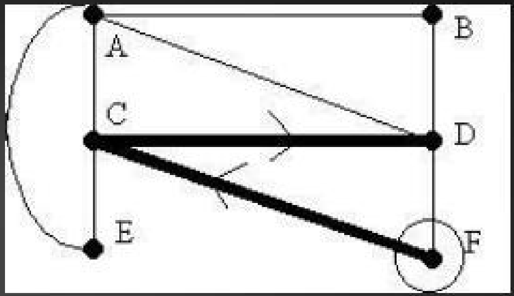
Step 2



Fleury's algorithm example (2/3)

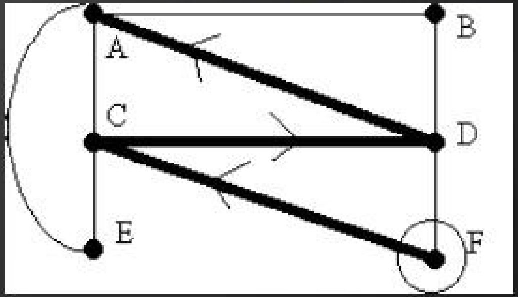
3. Travel from C to D (arbitrary choice).

Step 3



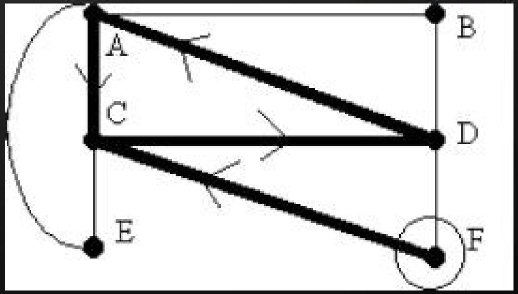
4. Travel from D to A (arbitrary choice).

Step 4



5. Travel from A to C (Can't go to B!).

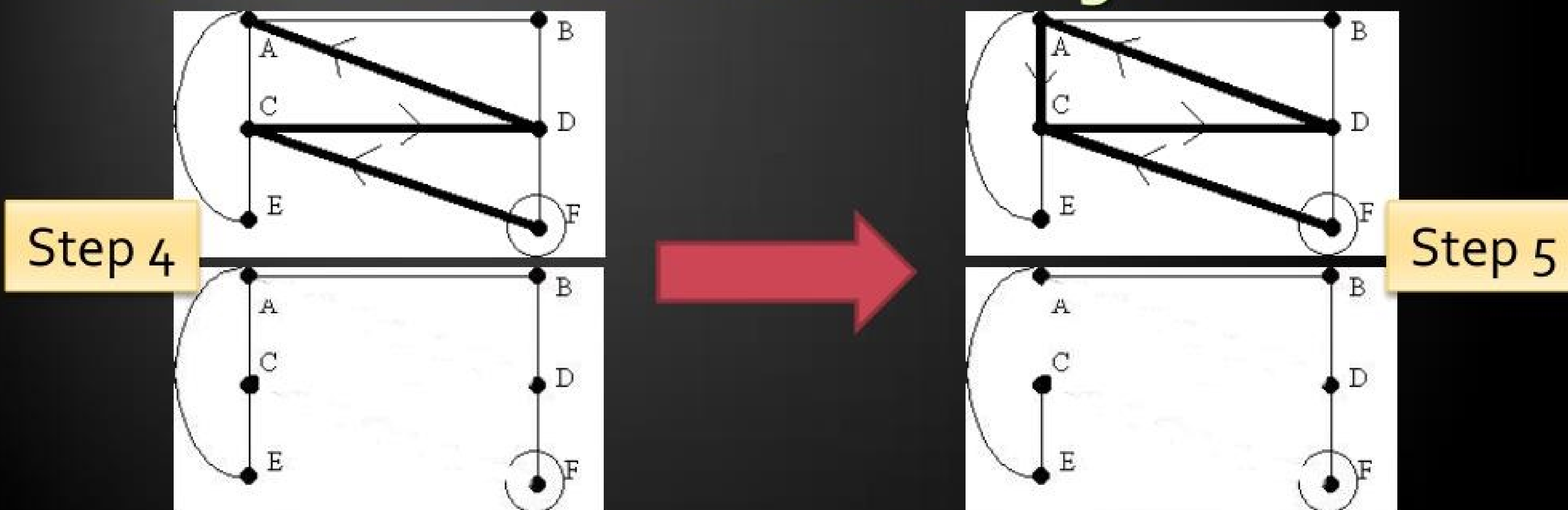
Step 5



Fleury's algorithm example (3/3)

Erasing the passed edges makes it easier to see:

- It's impossible to erase edge A->B in step 5 because this will disconnect the graph and we wouldn't be able to visit each edge!



The rest of the path is obvious. So the cycle is:

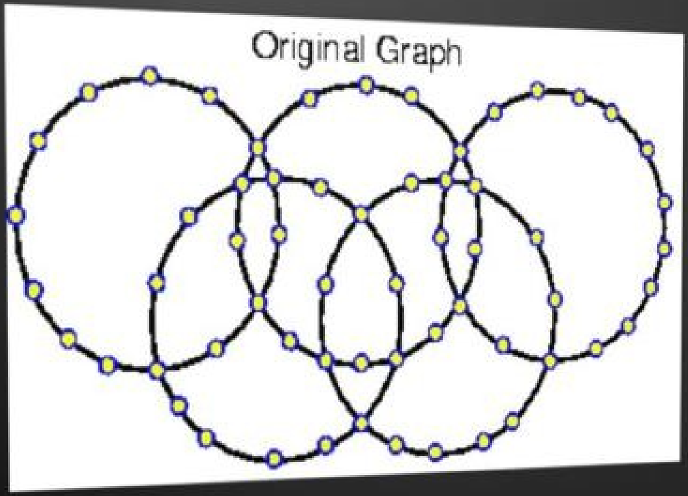
F -> C -> D -> A -> C -> E -> A -> B -> D -> F

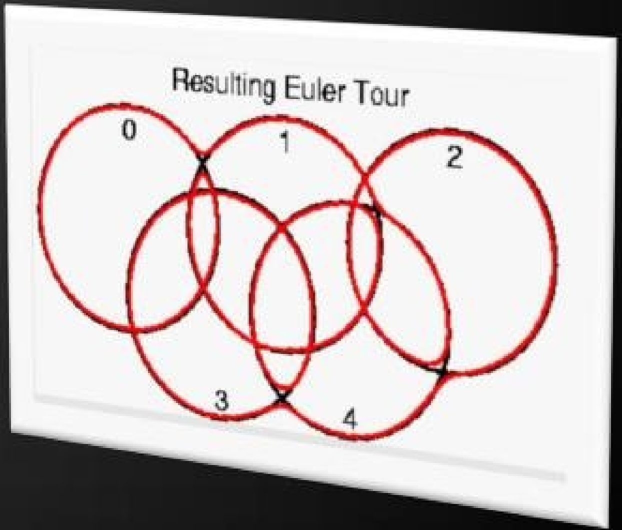
Fleury's algorithm summary

- Fleury's algorithm is an elegant but inefficient algorithm.
  - While the graph traversal in Fleury's algorithm is linear in the number of edges, i.e. O(|E|), we also need to factor in the complexity of detecting bridges.
  - Linear time bridge-finding algorithm after the removal of every edge will make a time complexity of O(|E|2).
  - A dynamic bridge-finding algorithm allows this to be improved but this is still significantly slow.

Telerik Academy

Hierholzer's algorithm





Telerik Academy

Hierholzer's algorithm

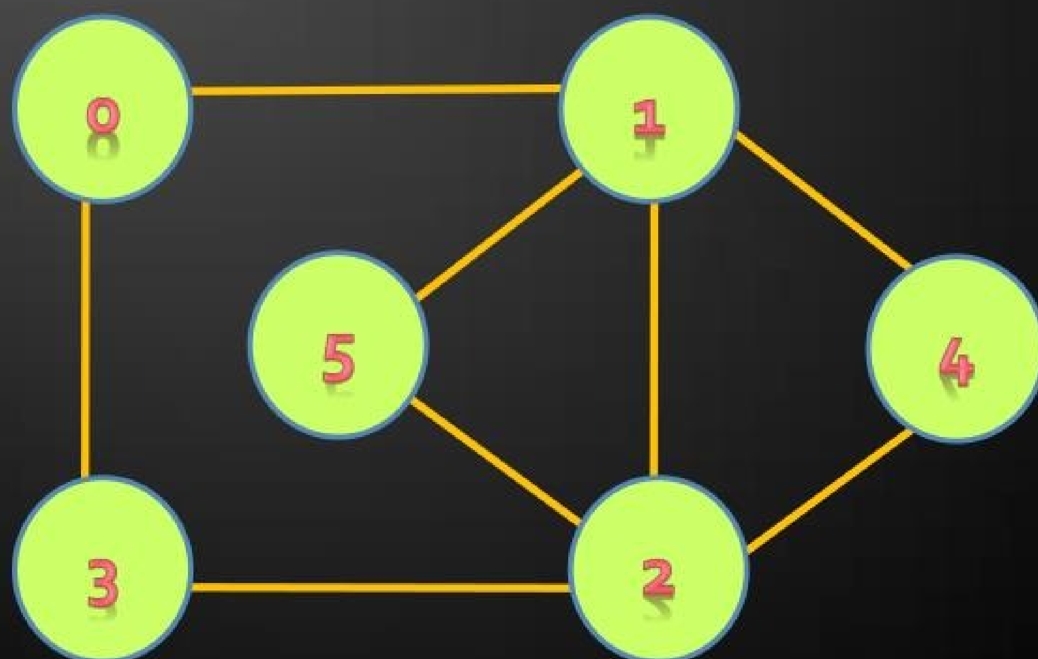
- Input: A connected graph $G = (V, E)$ *with two or zero vertices of odd degree (depending on whether we are searching for Euler's path or Euler's cycle).*
- Output: The graph with its edges labeled according to their order of appearance in the path found.

```
1 Find a simple cycle C in G.
2 Delete the edges belonging in C.
3 Apply algorithm to the remaining graph.
4 Amalgamate Euler cycles found to obtain the complete
Euler cycle.
```

Telerik Academy

Hierholzer's algorithm example

- We will use two stacks in this example: tempPath and finalPath in order to be able to combine the simple cycles we've found into one big cycle (the searched Euler's cycle).

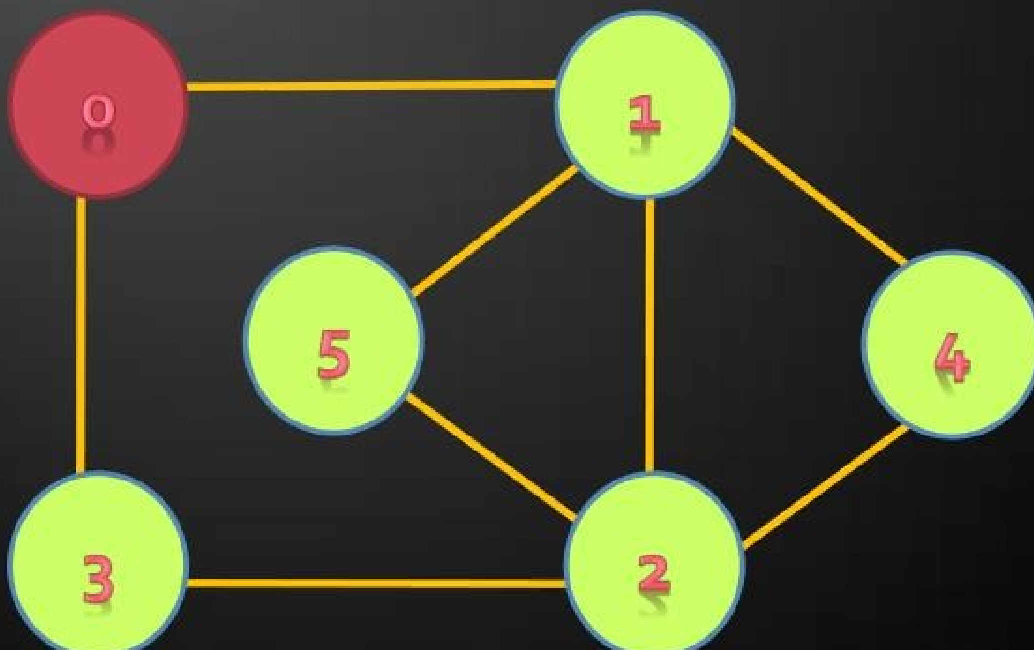


Hierholzer‘s algorithm example

1. Lets start with edge 0 (arbitrary choice)

Adding 0 to the tempPath stack.

```
tempPath:0
finalPath:<empty>
```



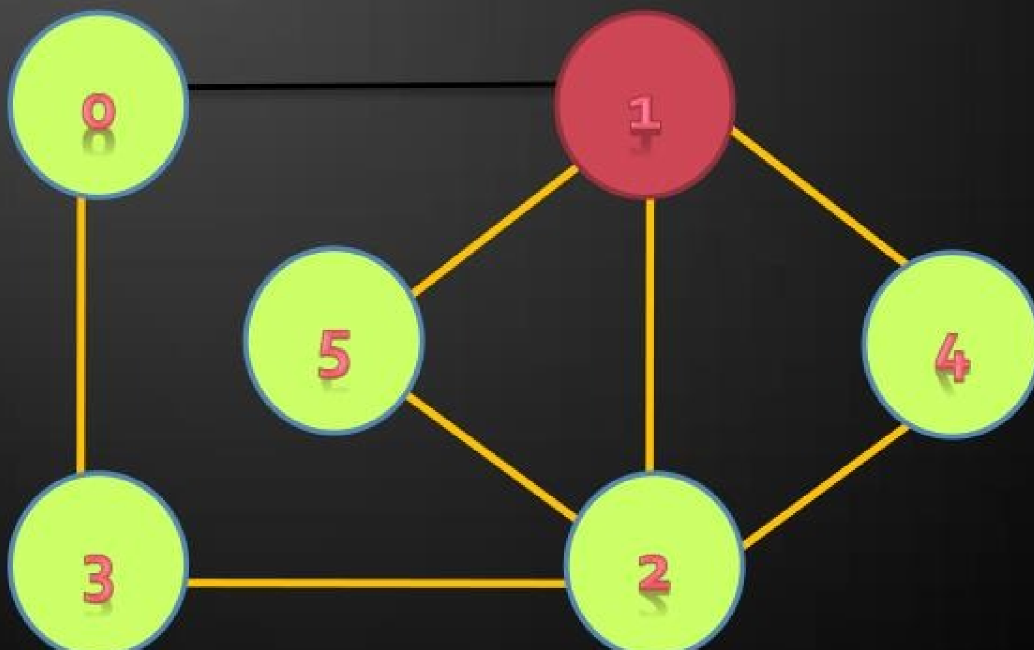
Hierholzer's algorithm example

Telerik Academy

2. Lets choose edge 0->1 (arbitrary choice).

Erasing the edge and adding 1 to tempPath stack

```
tempPath:0 1
finalPath:<empty>
```



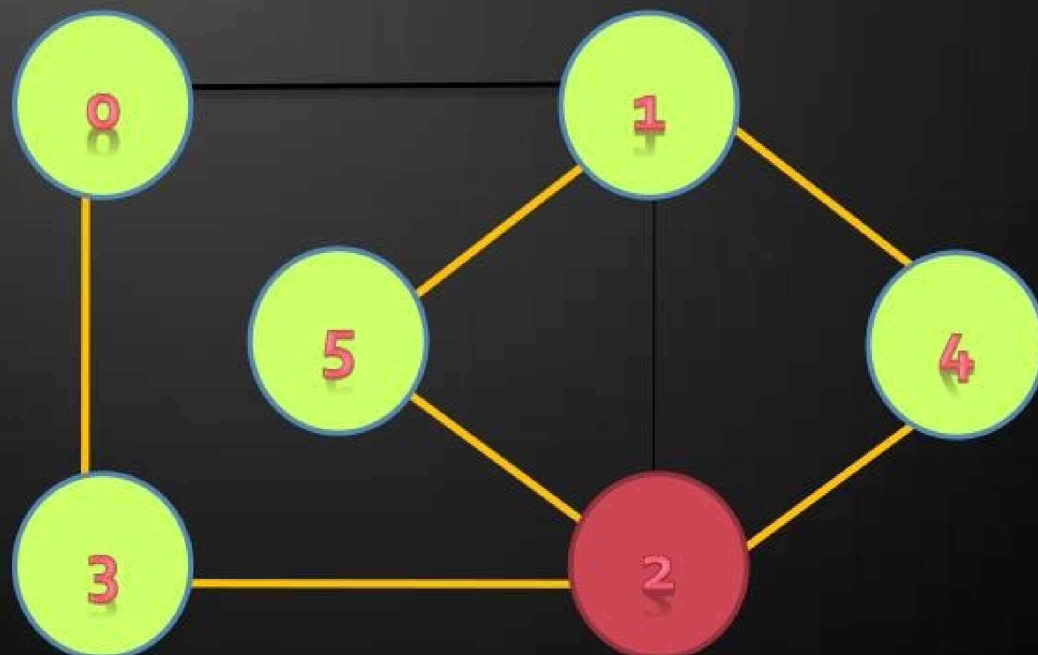
Telerik Academy

Hierholzer's algorithm example

3. Lets choose edge 1->2 (arbitrary choice).

Erasing the edge and adding 2 to tempPath stack

```
tempPath:0 1 2
finalPath:<empty>
```

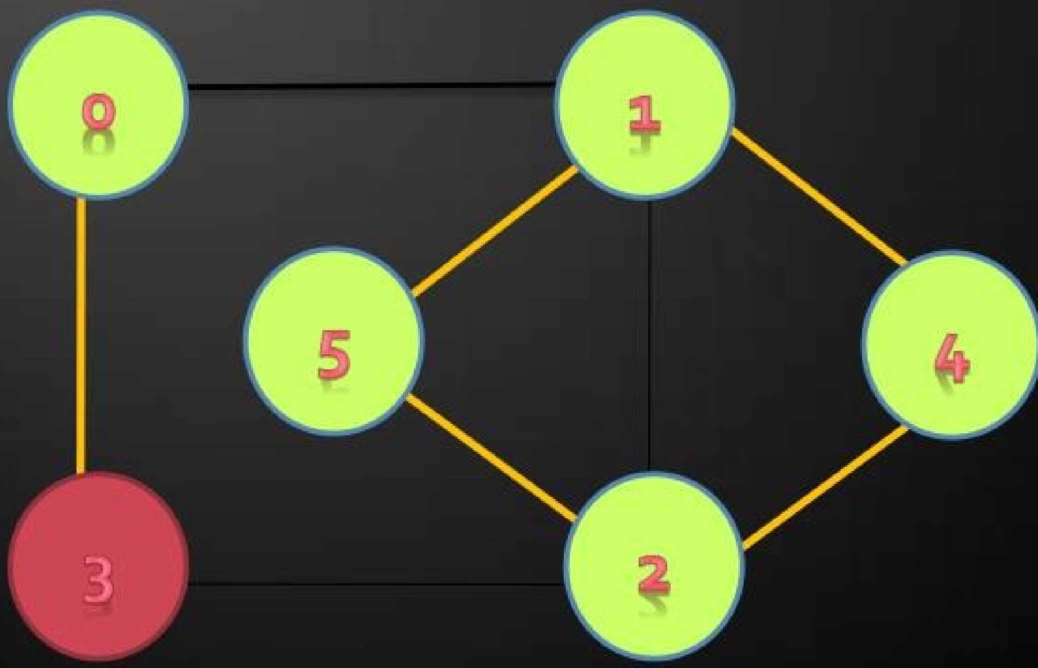


Hierholzer's algorithm example

4. Lets choose edge 2->3 (arbitrary choice).

Erasing the edge and adding 3 to tempPath stack

```
tempPath:0 1 2 3
finalPath:<empty>
```



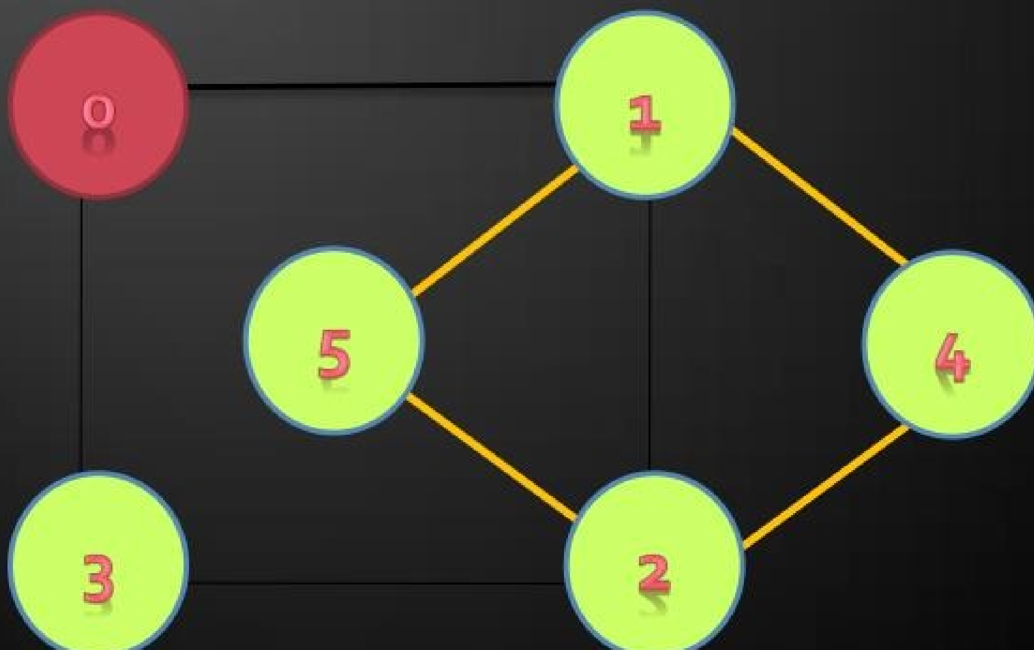
Telerik Academy

Hierholzer's algorithm example

5.Lets choose edge 3->0 (only possible choice).

Erasing the edge and adding 0 to tempPath stack

```
tempPath:0 1 2 3 0
finalPath:<empty>
```

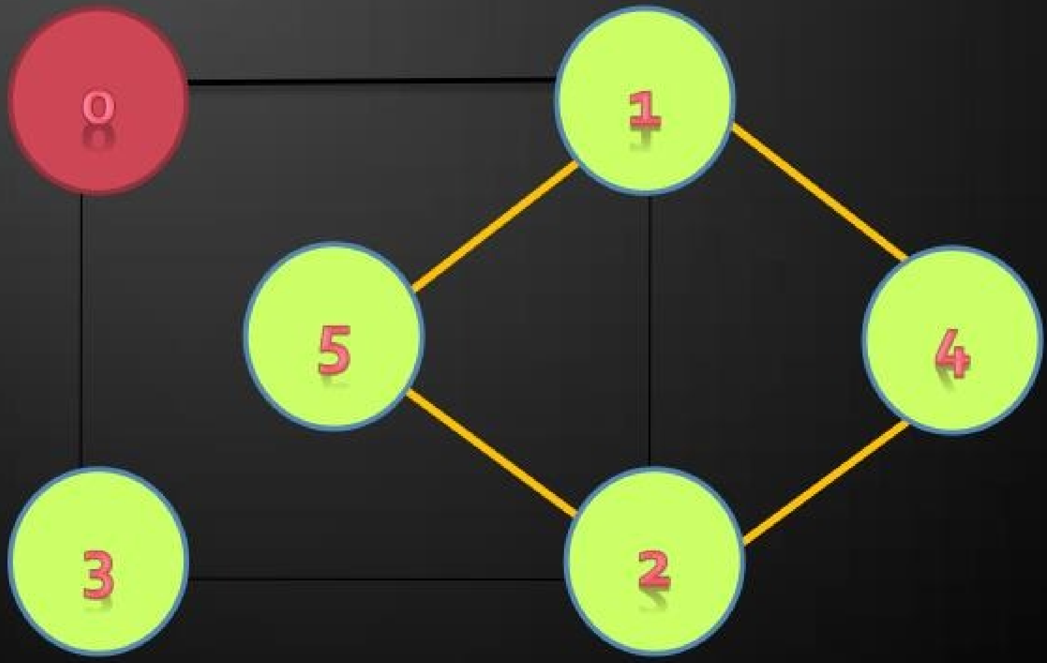


Telerik Academy

Hierholzer's algorithm example

- We've created a simple cycle and there is nowhere to go, but there are still unvisited edges! Go back to vertex with unvisited edge, moving elements from tempPath to finalPath.

```
tempPath:0 1 2 3 0
finalPath:<empty>
```



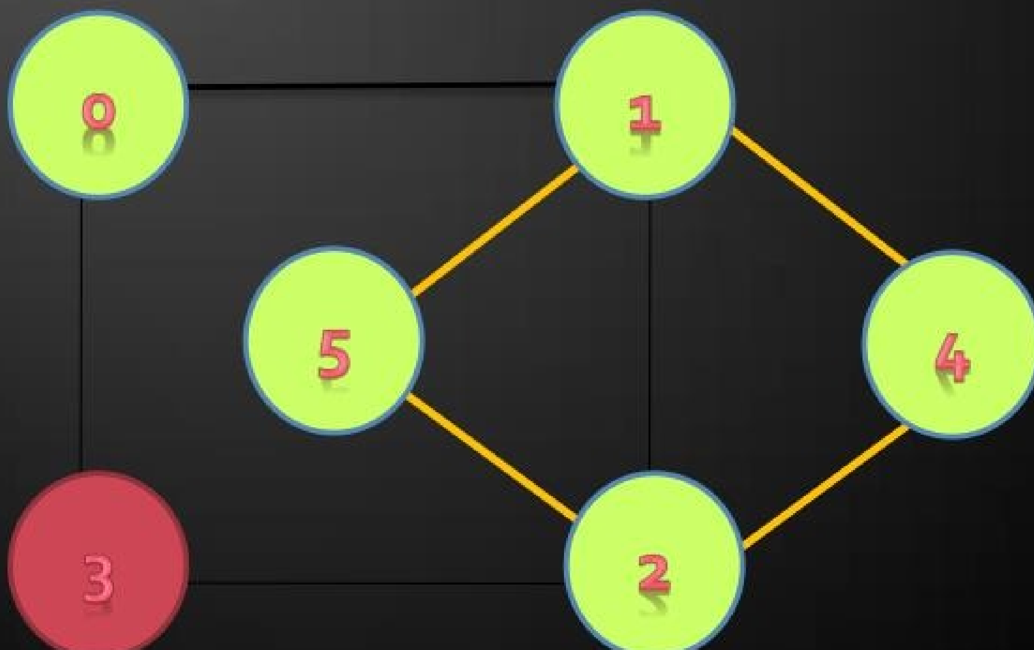
Telerik Academy

Hierholzer's algorithm example

6. Move back from 0 to 3.

Move 0 from tempPath to finalPath.

```
tempPath:0 1 2 3
finalPath:0
```



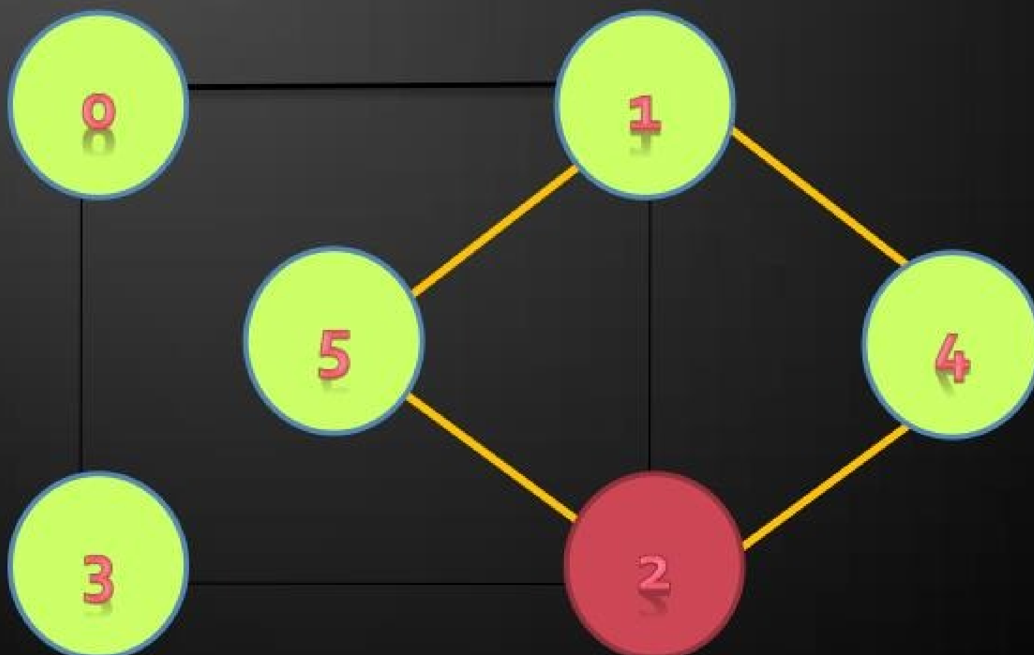
Telerik Academy

Hierholzer's algorithm example

7. Move back from 3 to 2.

Move 3 from tempPath to finalPath.

```
tempPath:0 1 2
finalPath:0 3
```

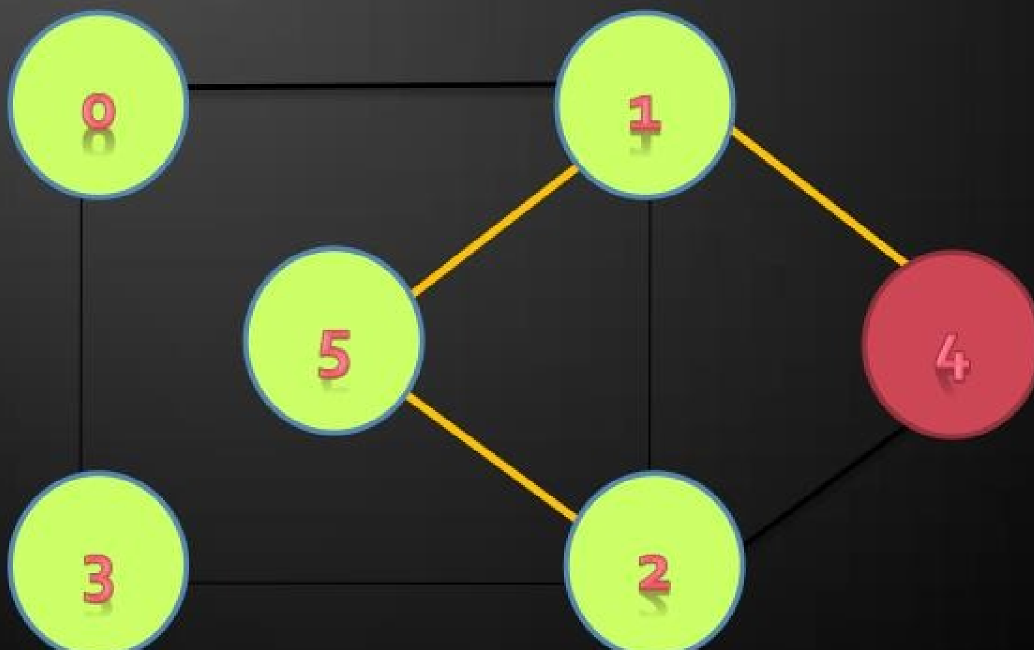


Hierholzer's algorithm example

8. Lets choose edge 2->4 (arbitrary choice).

Erasing the edge and adding 4 to tempPath stack

```
tempPath:0 1 2 4
finalPath:0 3
```



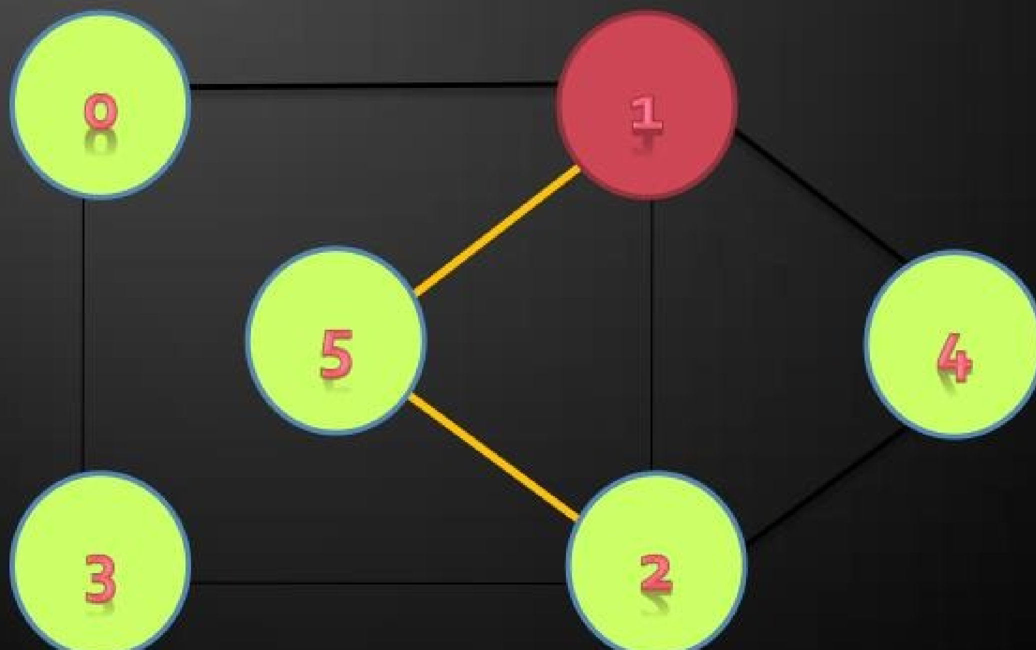
Telerik Academy

Hierholzer's algorithm example

9. Lets choose edge 4->1 (only possible choice).

Erasing the edge and adding 1 to tempPath stack

```
tempPath:0 1 2 4 1
finalPath:0 3
```

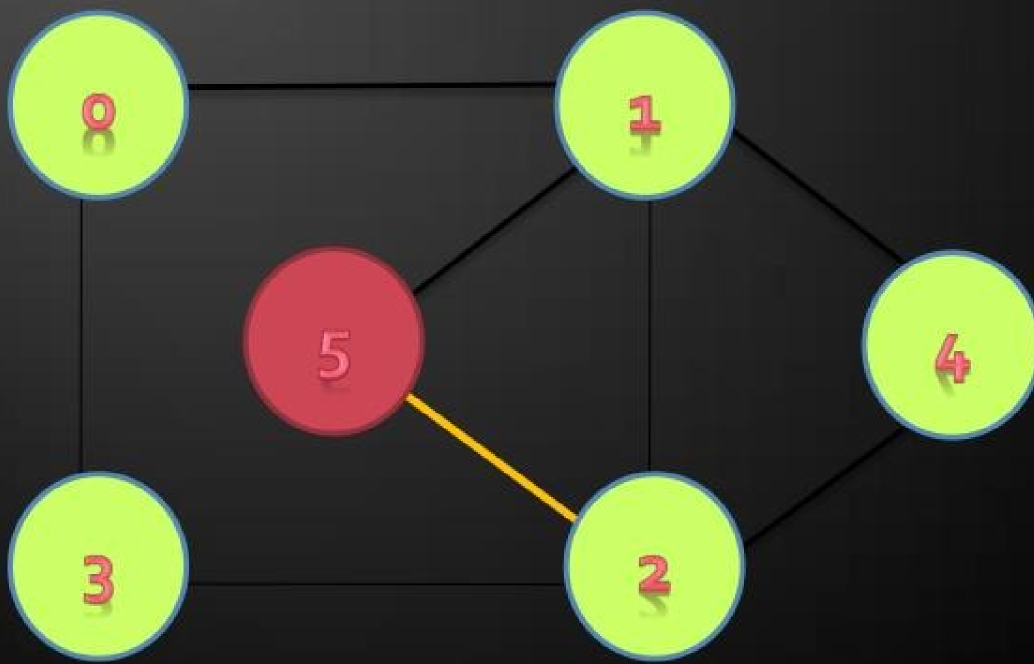


Hierholzer's algorithm example

10. Lets choose edge 1->5 (only possible choice).

Erasing the edge and adding 5 to tempPath stack

```
tempPath:0 1 2 4 1 5
finalPath:0 3
```

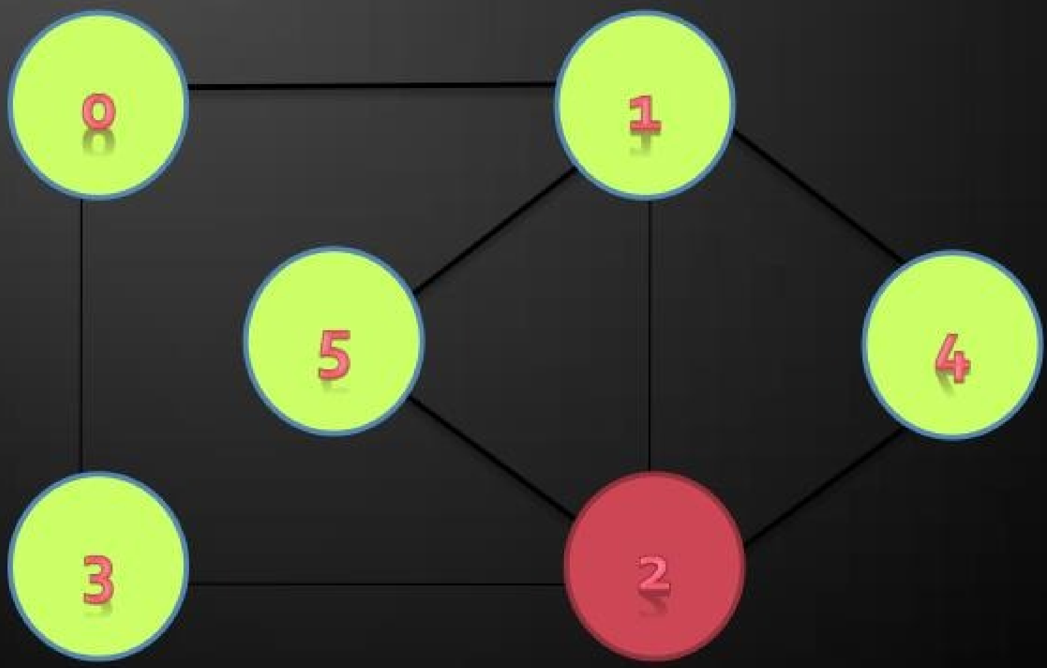


Hierholzer's algorithm example

11. Lets choose edge 5->2 (only possible choice).

Erasing the edge and adding 2 to tempPath stack

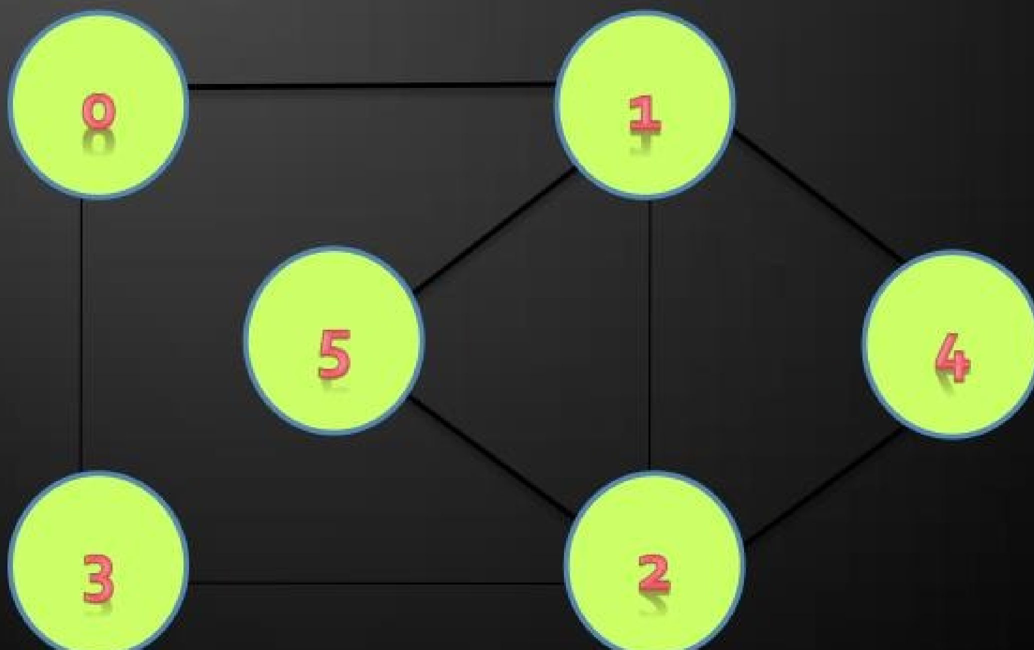
```
tempPath:0 1 2 4 1 5 2
finalPath:0 3
```



Hierholzer's algorithm example

- So we have passed through all the edges! Now all we have to do is move elements from tempPath stack to finalPath stack!

```
tempPath:0 1 2 4 1 5 2
finalPath:0 3
```



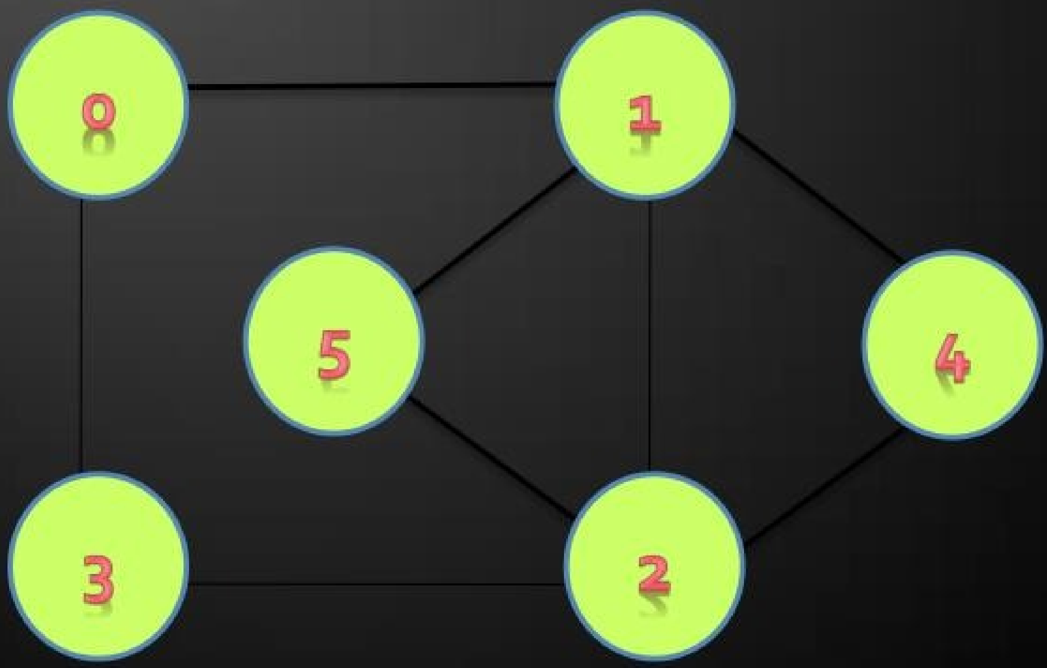
Telerik Academy

Hierholzer's algorithm example

- The result sequence in finalPath stack is an Euler's cycle!

```
tempPath:0 1 2 4 1 5 2
finalPath:0 3
```

```
tempPath:&lt;empty&gt;
finalPath:0 3 2 5 1 4 2 1 0
```



Hierholzer's algorithm summary

- Hierholzer's algorithm is an elegant and efficient algorithm.
  - The algorithm takes about linear time.
  - Can be easily changed to find all Euler's cycles if necessary
    - when we meet the word "arbitrary choice" in previous example, we may use recursion on all possible choices
  - Can be applied for searching Euler's path as well
    - must start from a vertex with odd degree.

Resources

- http://en.wikipedia.org/wiki/Eulerian_path
- Nakov's book: Programming = ++Algorithms
- http://www8.cs.umu.se/~jopsi/dinf504/chap14.shtml
- http://www.csd.uoc.gr/~hy583/papers/ch14.pdf

Summary

- Euler's path is a path passing through each graph's edge exactly once.
- An undirected graph has an Euler's cycle if and only if every vertex has even degree, and all of its vertices with nonzero degree belong to a single connected component.
- For finding an Euler's cycle or path the fastest way is by using Hierholzer's constructive algorithm.

Questions?

Free Trainings @ Telerik Academy

- "C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com

- Telerik Software Academy
  - academy.telerik.com
- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com