Tutorial 12

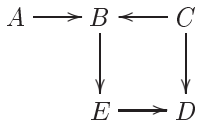
Exercise 1

- 1
- 4
- 9
- 25
- n^2

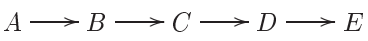
	Adjacency Matrix	Adjacency List
Nothing known	$O(n^2)$	$O(n+m)$
$m = O(n)$	$O(n^2)$	$O(n)$
$m = \Omega(n^2)$	$O(n^2)$	$O(n^2)$
$n = \Omega(m^2)$	$O(n^2)$	$O(n)$
$m = O(n \cdot \log n)$	$O(n^2)$	$O(n \cdot \log n)$

- Adjacency list is more space efficient, the 2nd, 4th and 5th row from the table above demonstrate this. If the graph is sparse (only a few edges) then adjacency list is much more space efficient compared to adjacency matrix. On the other hand if the graph contains almost all possible edges, the space complexity of both implementations is asymptotically the same.

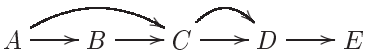
Exercise 2

- 

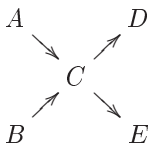
One sequence of topologically ordered nodes of the DAG G is e.g A, C, B, E, D.
- The following DAG has exactly one topologically ordered sequence of nodes.



This one has also exactly one topologically ordered sequence of nodes but contains more edges.



- Consider e.g. the following DAG.



  All sequences of topologically ordered nodes are:

  - A, B, C, D, E
  - A, B, C, E, D
  - B, A, C, D, E
  - B, A, C, E, D

- Maximal number of topologically ordered sequences is in a DAG with no edges, in our case it consists of 5 isolated nodes. The number of all topological orders is 120 in this case.

- $n!$

Exercise 3

$g.make$
$a := g.new\_vertex(A)$
$b := g.new\_vertex(B)$
$c := g.new\_vertex(C)$
$d := g.new\_vertex(D)$
$g.insert\_vertex(a)$
$g.insert\_vertex(b)$
$g.insert\_vertex(c)$
$g.insert\_vertex(d)$
$g.insert\_edge(g.new\_edge(1), a, b)$
$g.insert\_edge(g.new\_edge(3), b, b)$
$g.insert\_edge(g.new\_edge(7), c, d)$