Lecture # 16

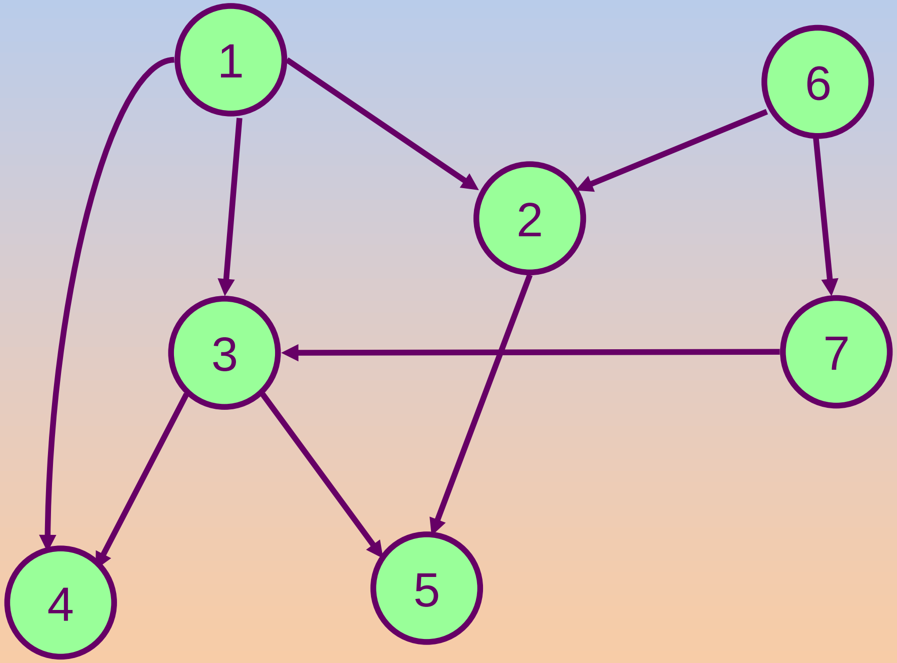
- Topological Sort Algorithm for Directed Acyclic Graph

Outline of Presentation

- What is Directed Acyclic Graph (DAG)?
- Topological Sort
- Existence of Topological Sort
- Topological Sort – Applications
- Topological Sort – Algorithm
- Example 1 – Getting Dressed for Office
- Example 2 – Course prerequisite Plan at University
- Time Complexity

What is Directed Acyclic Graph (DAG)?

- A Graph:
  - Which is Directed
  - and contain *no directed cycles*



Topological Sort

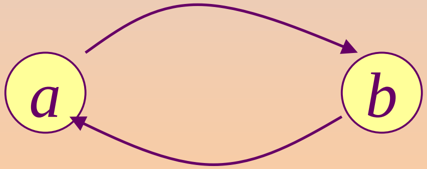
- *Topological sort* of a DAG:
  - Linear ordering of all vertices in graph G such that vertex *u* comes before vertex *v* if there is an edge $(u, v) \in G$.
- It is important to note that if the graph is not acyclic, then no linear ordering is possible. That is, we must not have circularities in the directed graph. For example, in order to get a job you need to have work experience, but in order to get work experience you need to have a job.

Existence of Topological Sort

Lemma
A directed graph *G* is acyclic iff a DFS of G yields no back edges.

Lemma
G can be topologically sorted iff it has no cycle, that is, iff it is a ***dag*** (directed acyclic graph).

Proof ⇒ If *G* has a cycle, then it cannot be topologically sorted.



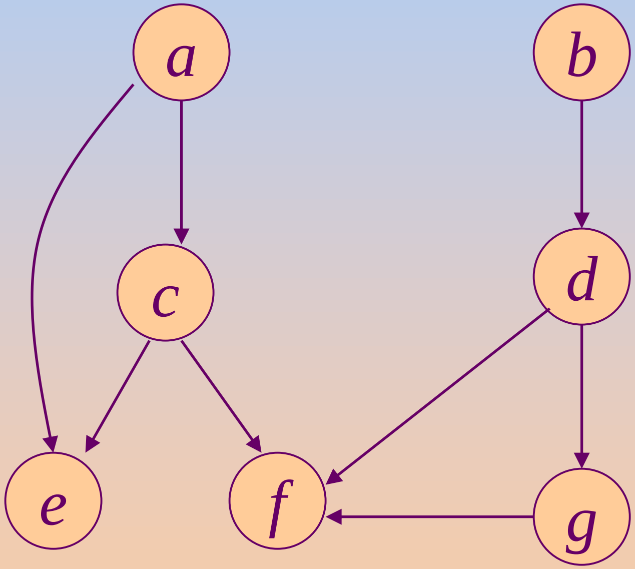
⇐ If *G* has no cycle, then it can be topologically sorted.

Topological Sort

- Each node represents an activity; e.g., taking a class.
- $(u, v) \in E(G)$ implies activity u must be scheduled before activity v.
- Topological sort schedules all activities.
- More than one schedule may exist.

Topological Sort of Digraphs

Different orderings in Topological Sort Algorithm are as follows



Some topological sorts:

1. *a, c, e, b, d, g, f*
2. *a, b, c, d, g, f, e*
3. *b, d, g, a, c, f, e*

Topological Sort - Applications

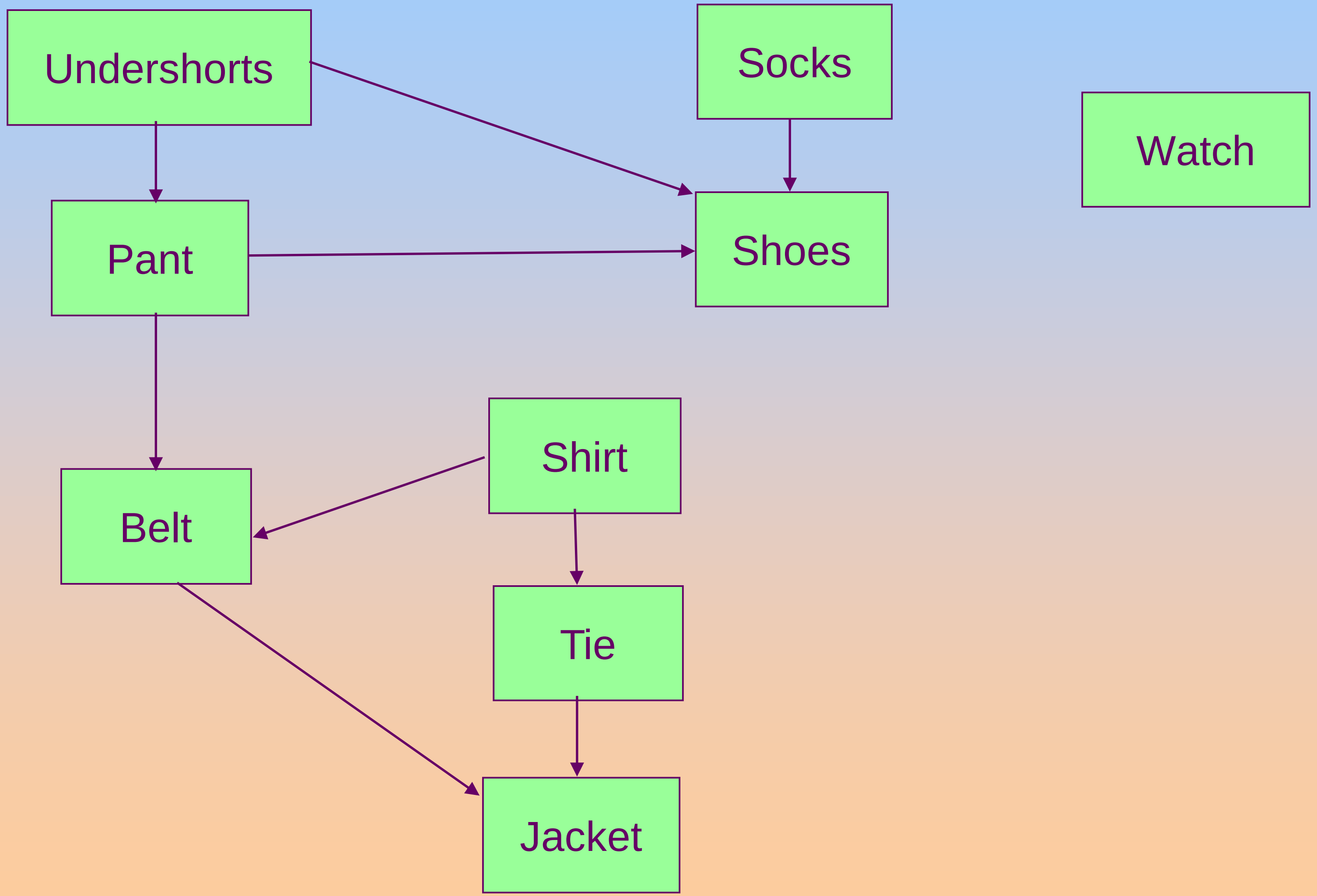
- Scheduling a dependent graph.
- Find a feasible course plan for university studies with Course prerequisites.
- Job scheduling (car manufacturing etc.)
- Etc…

Topological Sort - Algorithm

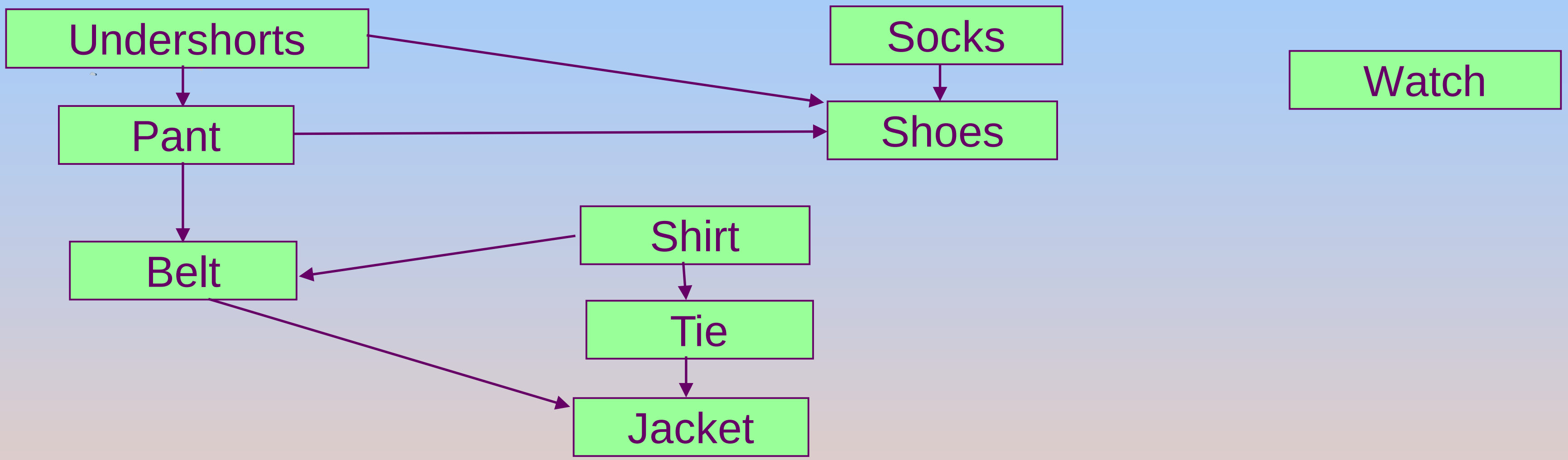
- Performed on a DAG.
- Linear ordering of the vertices of G such that if $(u, v) \in E$, then u appears somewhere before v.

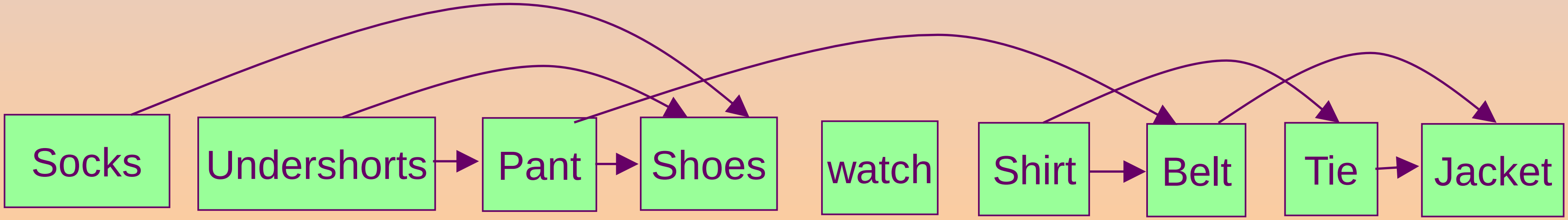
```
Topological-Sort ($G$)
1.  call DFS($G$) to compute finishing times $f[v]$ for all $v \in V$
2.  as each vertex is finished, insert it onto the front of a linked list
3.  return the linked list of vertices
```

Example 1 – Getting Dressed for Office

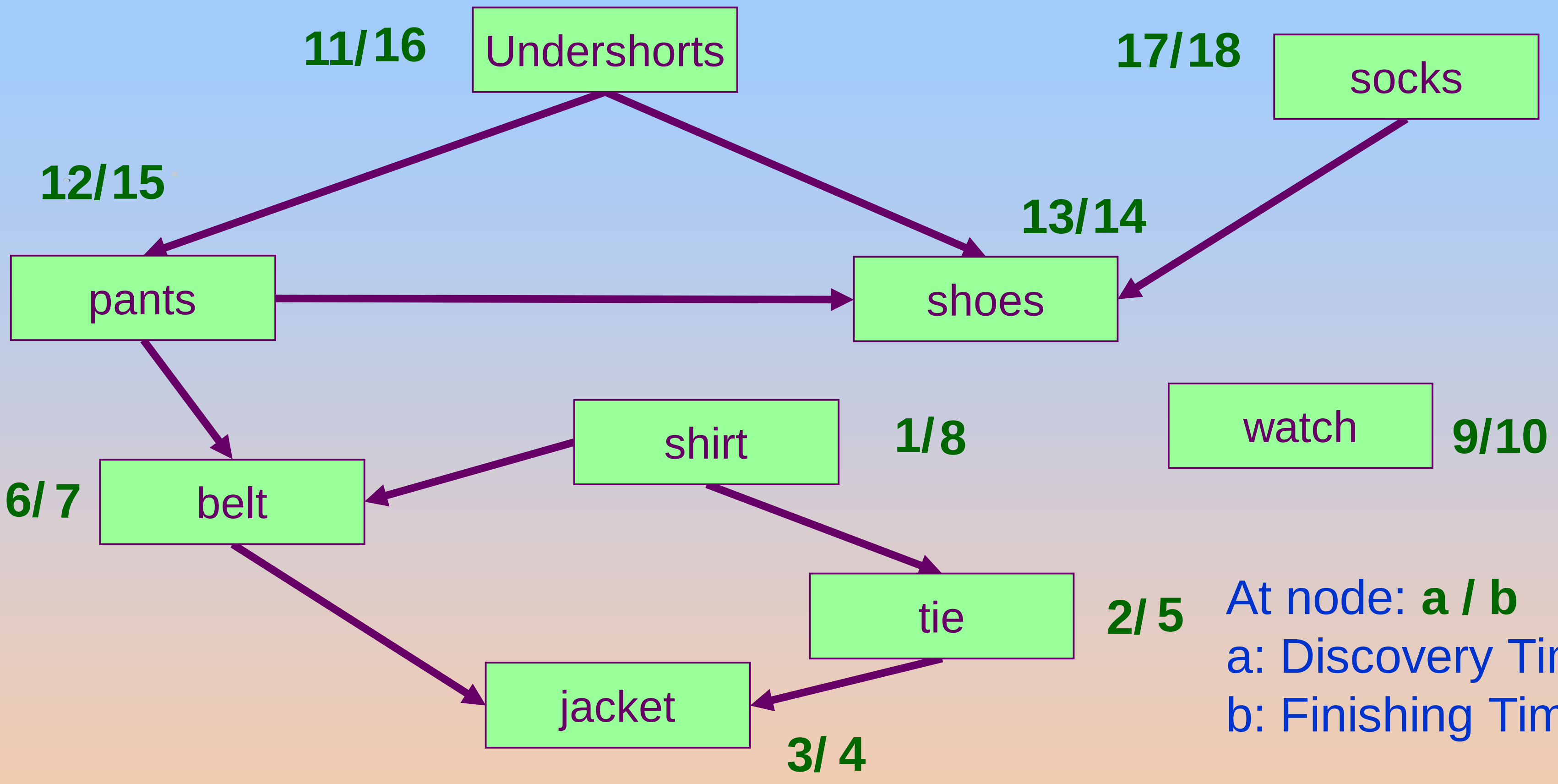


Example (Cont.)





Example – Getting Dressed for Office



11/16 Undershorts

17/18 socks

12/15 pants

13/14 shoes

shirt 1/8

watch 9/10

6/ 7 belt

tie 2/ 5

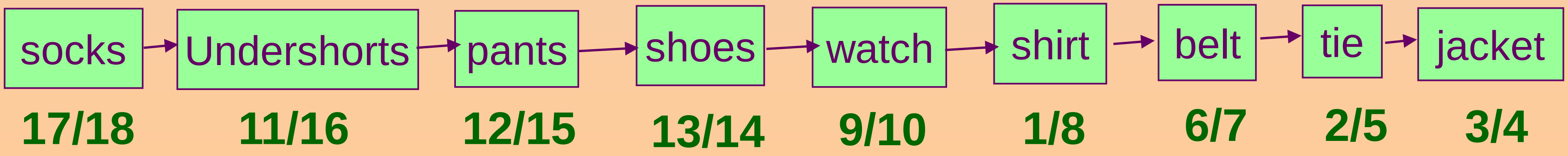
jacket 3/ 4

At node: **a / b**

a: Discovery Time

b: Finishing Time

Linked List :-



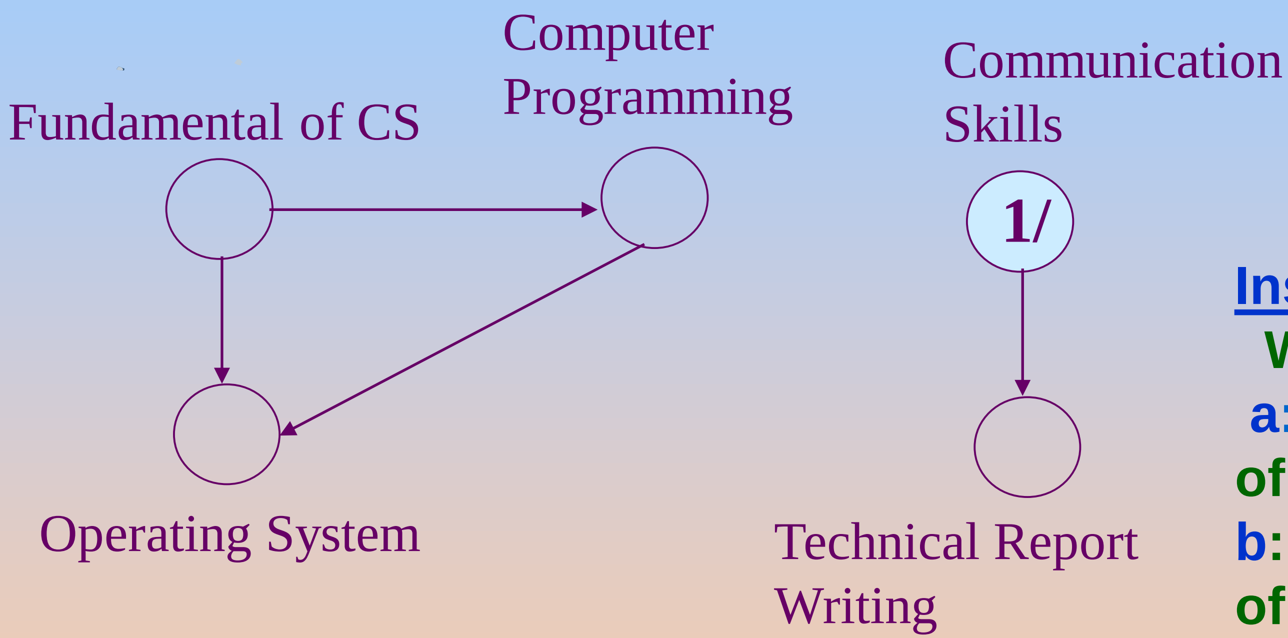
socks → Undershorts → pants → shoes → watch → shirt → belt → tie → jacket

17/18 11/16 12/15 13/14 9/10 1/8 6/7 2/5 3/4

Another Example

- The CS dept course prerequisites can be represented as a directed acyclic graph (DAG).
  - ☞ It must be directed because one course is the prerequisite for another (and not vice versa).
  - ☞ There can't be any cycles because then it would be impossible to meet all the prerequisites.

Example 2- Course prerequisite Plan at University



Fundamental of CS

Computer Programming

Communication Skills

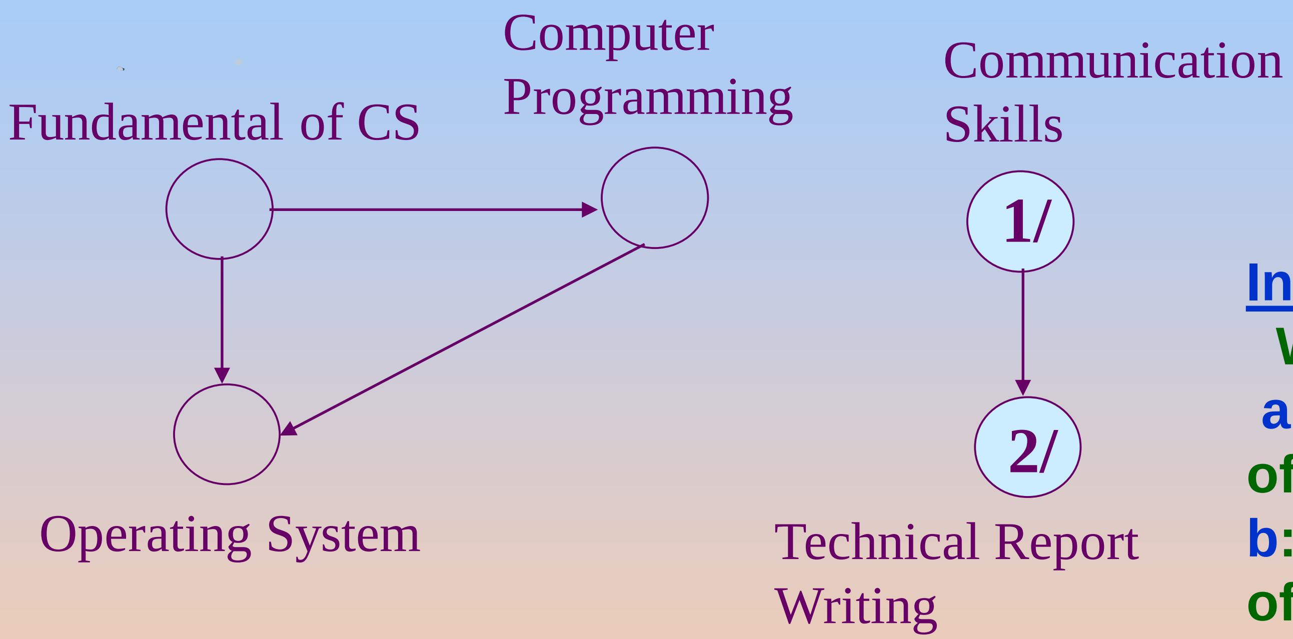
Operating System

Technical Report Writing

Inside Node: a/b
Where
a: Discovery Time of Node and
b: Finishing Time of Node

Linked List:

Example - 2



Fundamental of CS

Computer Programming

Communication Skills

1/

2/

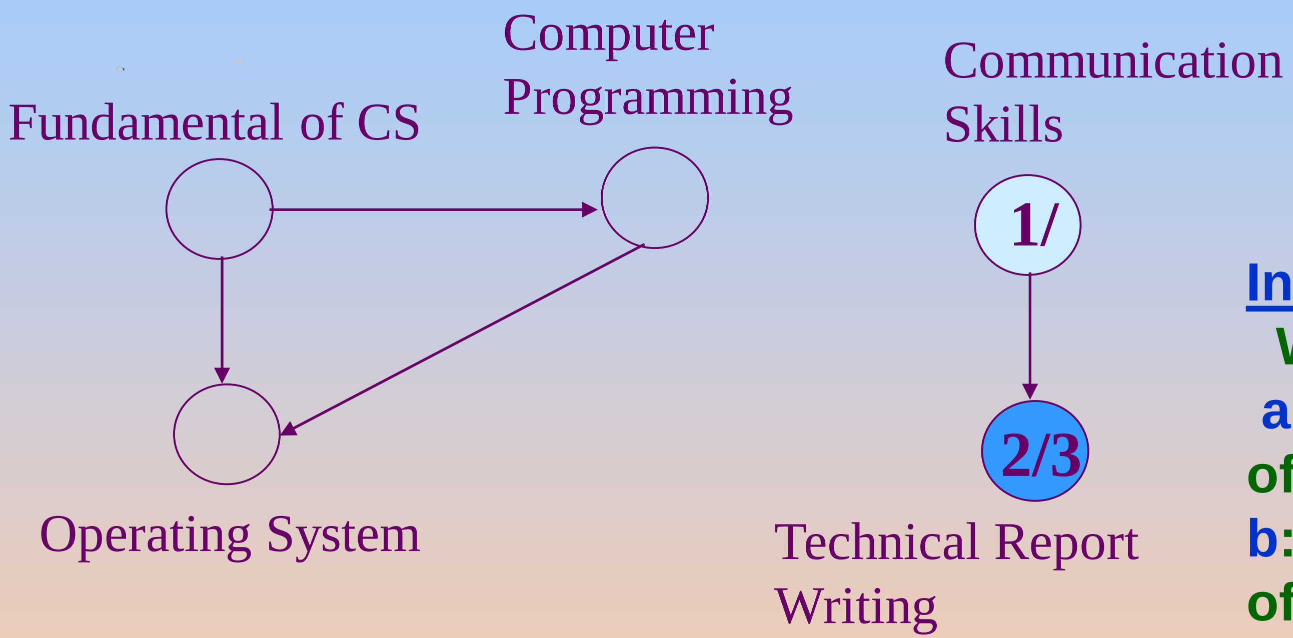
Operating System

Technical Report Writing

Inside Node: a/b
Where
a: Discovery Time of Node and
b: Finishing Time of Node

Linked List:

Example - 2



Fundamental of CS

Computer Programming

Communication Skills

1/

2/3

Operating System

Technical Report Writing

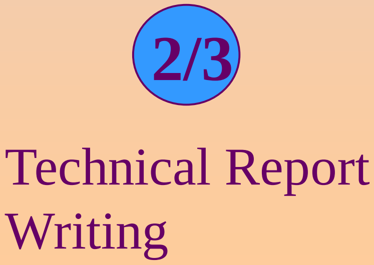
Inside Node: a/b

Where

a: Discovery Time of Node and

b: Finishing Time of Node

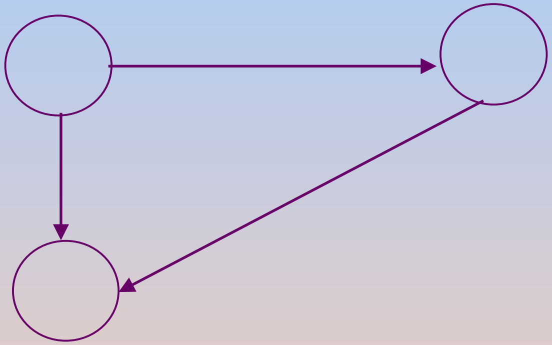
Linked List:



2/3

Technical Report Writing

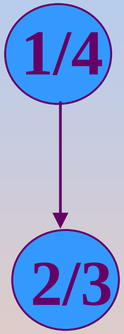
Example - 2



Computer Programming

Fundamental of CS

Operating System



Communication Skills

1/4

2/3

Technical Report Writing

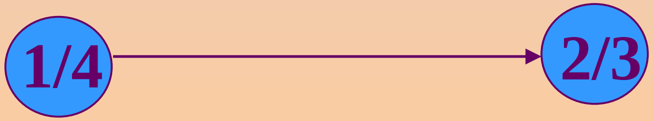
Inside Node: a/b

Where

a: Discovery Time of Node and

b: Finishing Time of Node

Linked List:



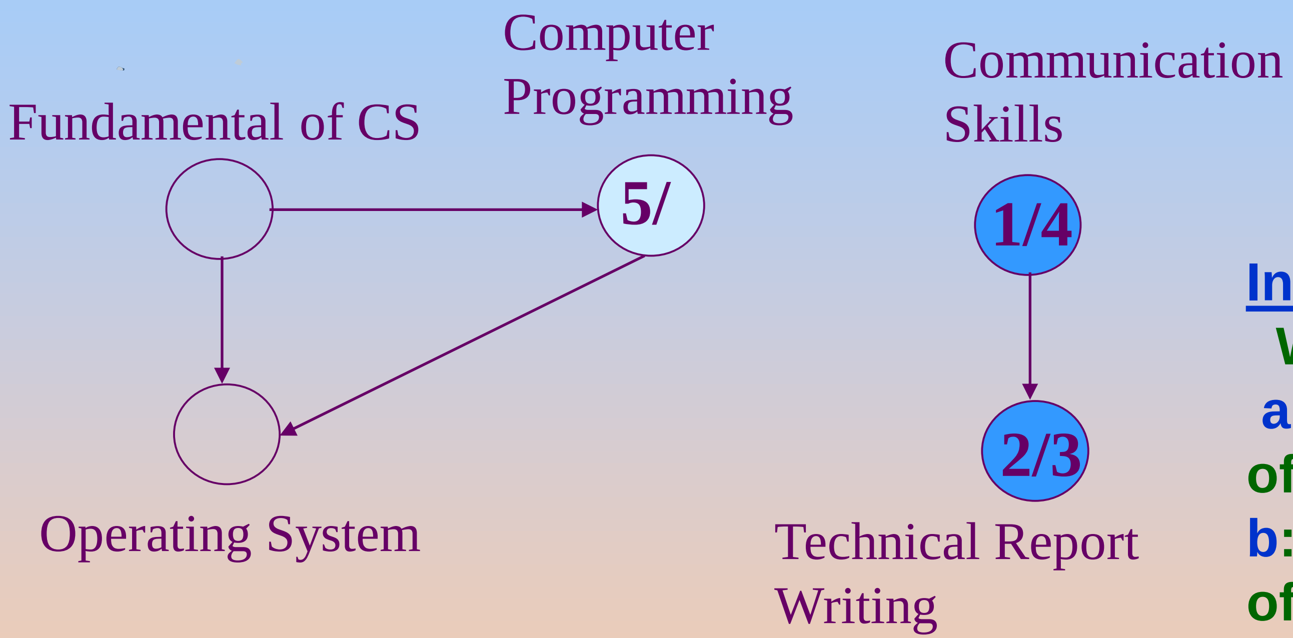
1/4

2/3

Communication Skills

Technical Report Writing

Example - 2



Fundamental of CS

Computer Programming

5/

Operating System

Communication Skills

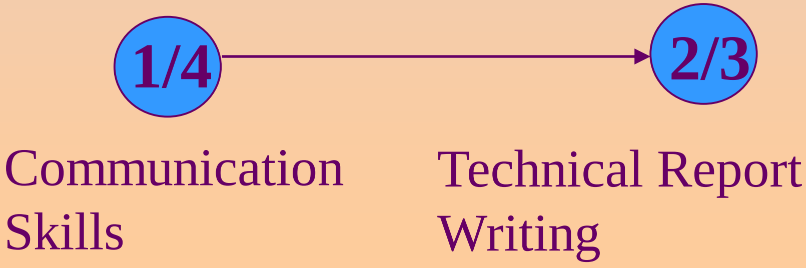
1/4

2/3

Technical Report Writing

Inside Node: a/b
Where
a: Discovery Time of Node and
b: Finishing Time of Node

Linked List:



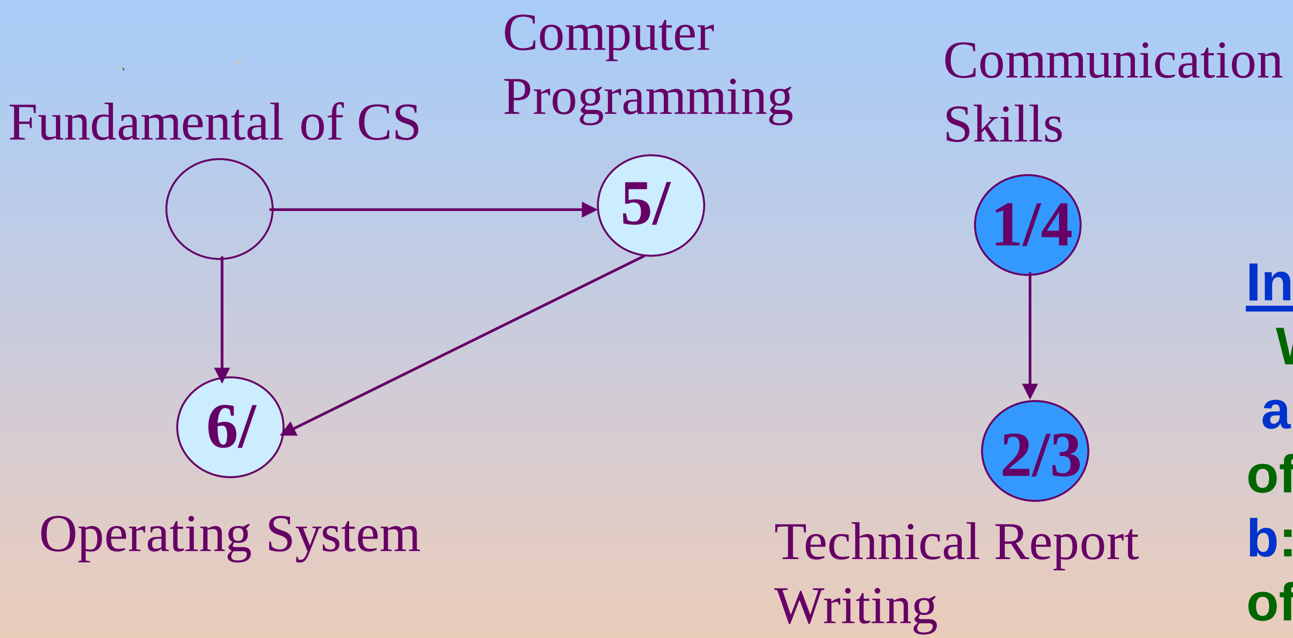
1/4

2/3

Communication Skills

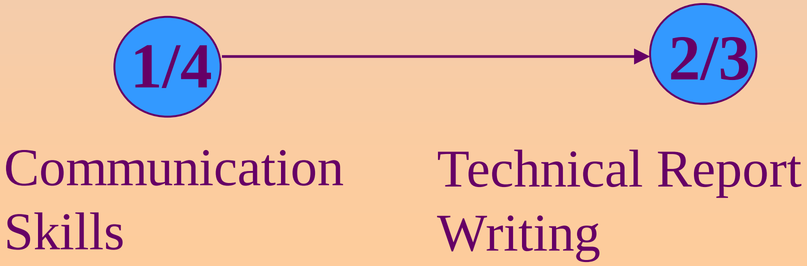
Technical Report Writing

Example - 2

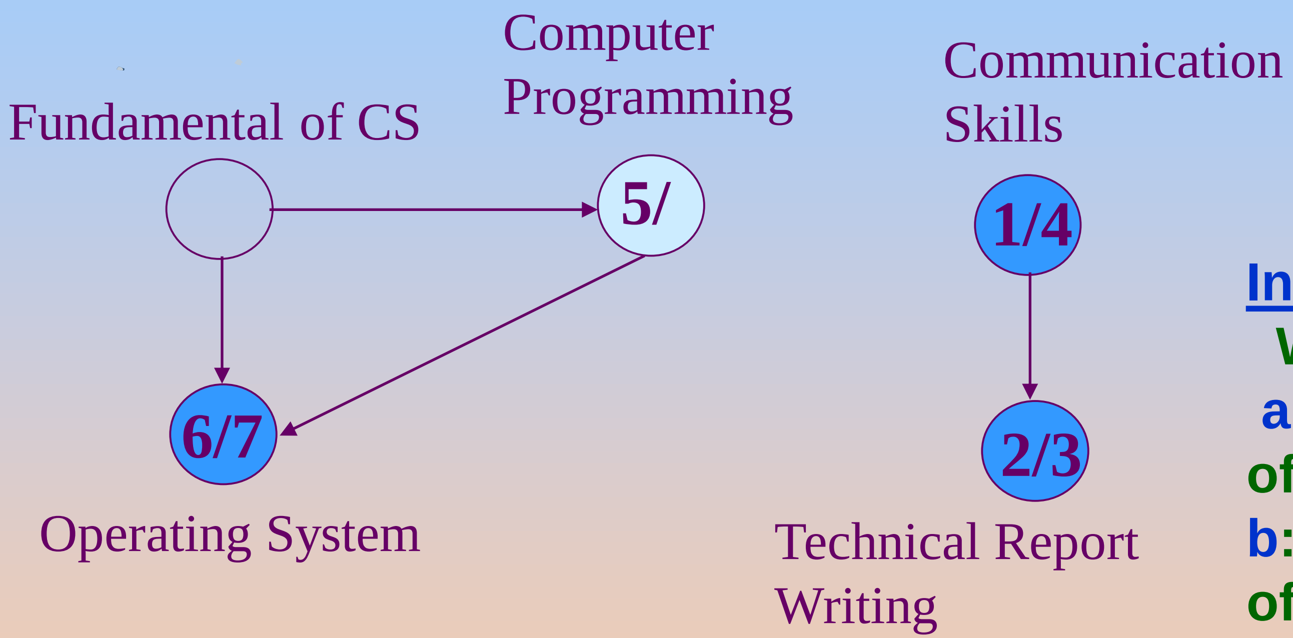


Inside Node: a/b
Where
a: Discovery Time of Node and
b: Finishing Time of Node

Linked List:



Example - 2



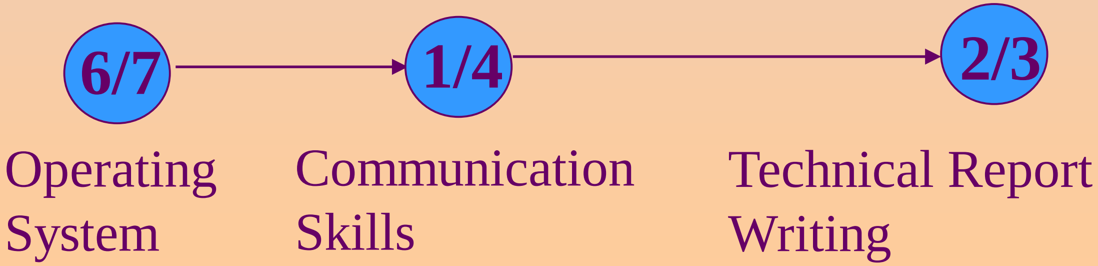
Inside Node: a/b

Where

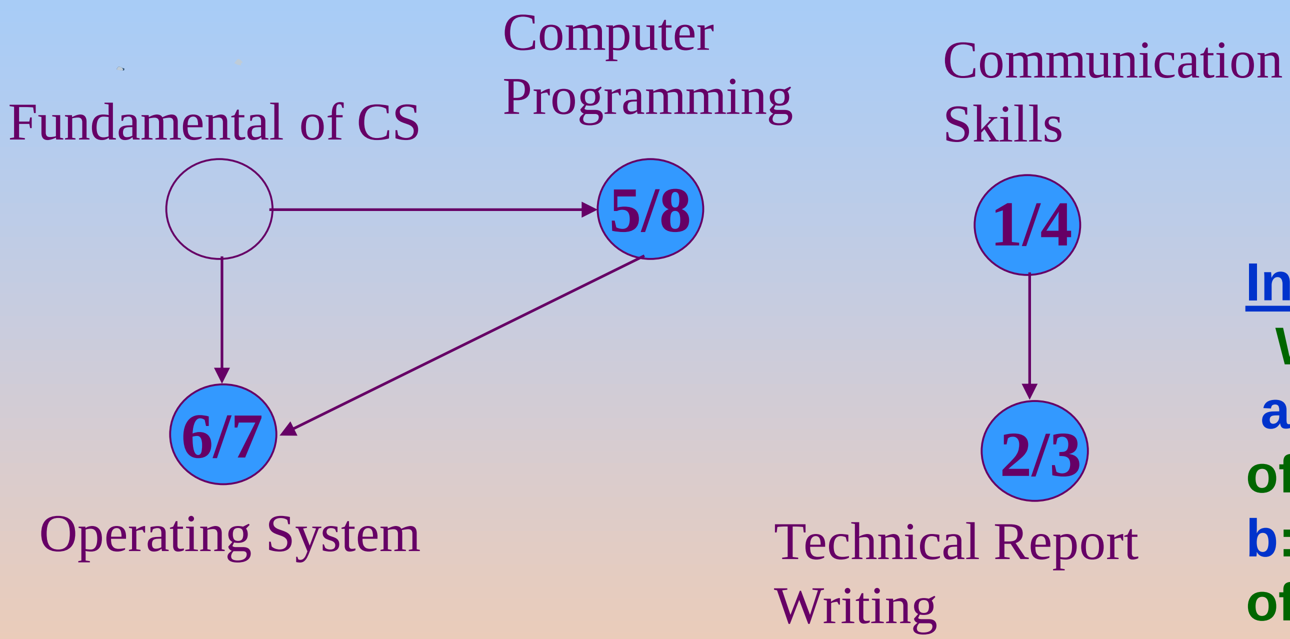
a: Discovery Time of Node and

b: Finishing Time of Node

Linked List:



Example - 2



Fundamental of CS

Computer Programming

5/8

6/7

Operating System

Communication Skills

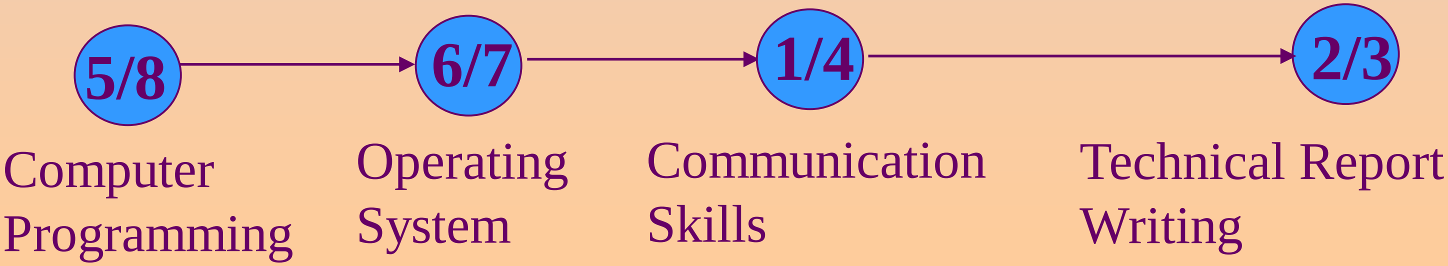
1/4

2/3

Technical Report Writing

Inside Node: a/b
Where
a: Discovery Time of Node and
b: Finishing Time of Node

Linked List:



5/8

Computer Programming

6/7

Operating System

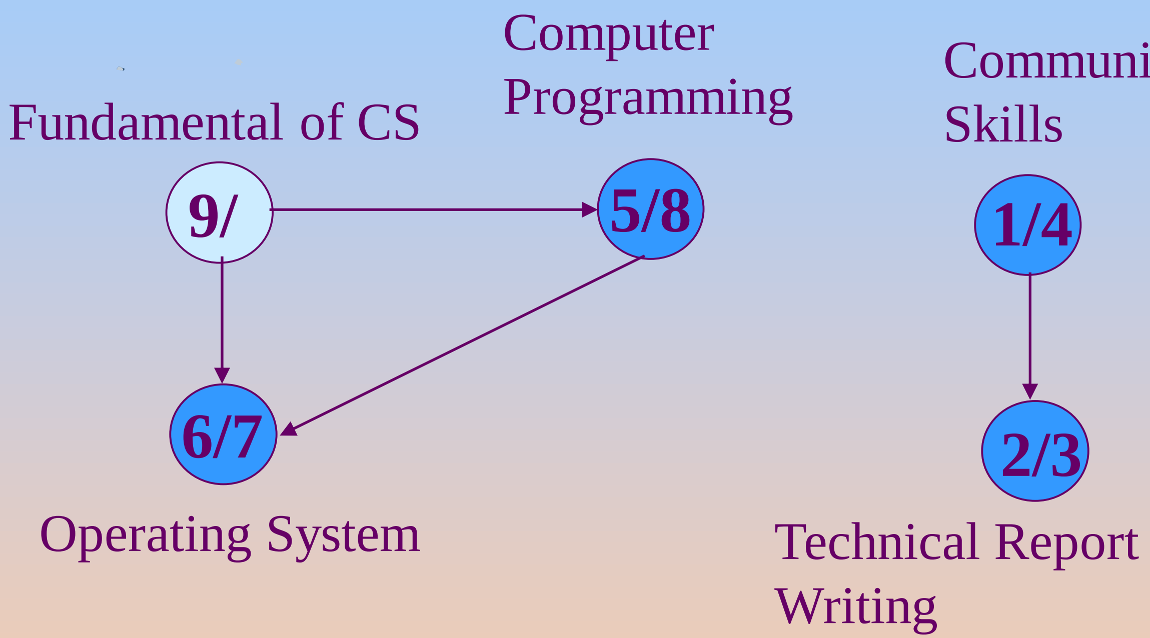
1/4

Communication Skills

2/3

Technical Report Writing

Example - 2



Computer Programming

Communication Skills

Fundamental of CS

9/

5/8

1/4

6/7

2/3

Operating System

Technical Report Writing

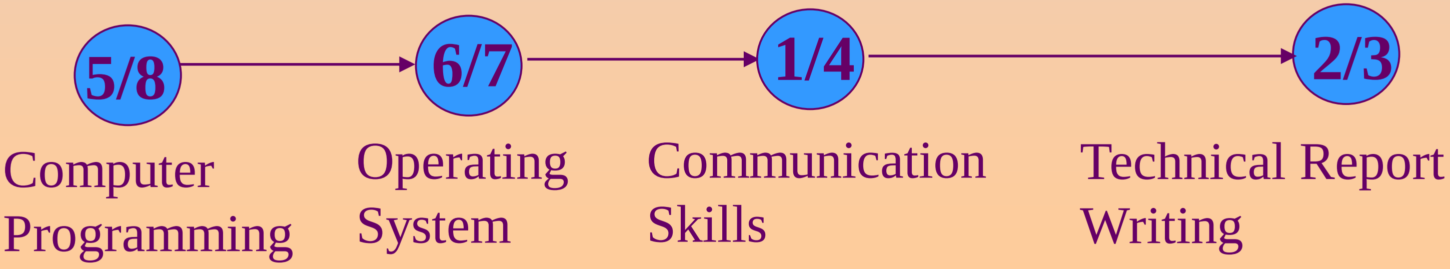
Inside Node: a/b

Where

a: Discovery Time of Node and

b: Finishing Time of Node

Linked List:



5/8

6/7

1/4

2/3

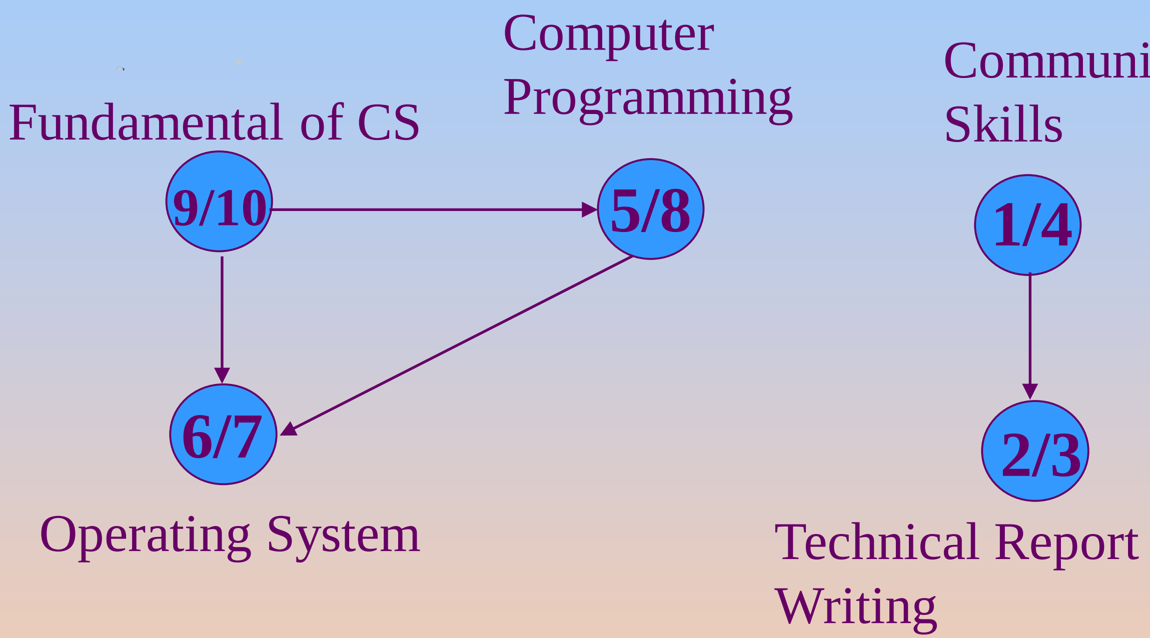
Computer Programming

Operating System

Communication Skills

Technical Report Writing

Example - 2



Fundamental of CS

9/10

Computer Programming

5/8

Operating System

6/7

Communication Skills

1/4

Technical Report Writing

2/3

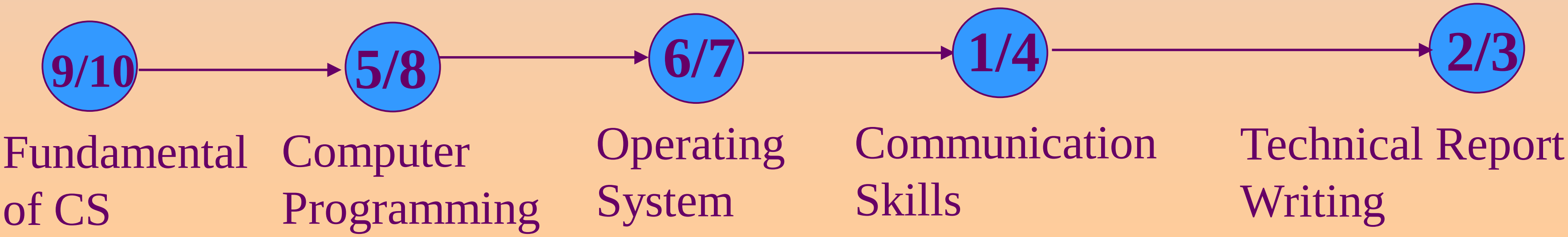
Inside Node: a/b

Where

a: Discovery Time of Node and

b: Finishing Time of Node

Linked List:



9/10

Fundamental of CS

5/8

Computer Programming

6/7

Operating System

1/4

Communication Skills

2/3

Technical Report Writing

Time Complexity

- It takes O(1) time to insert each of the |V| vertices onto the front of the linked list.

- Total running time of topological sort is θ(V+E) . Since DFS(G) search takes θ(V+E) time