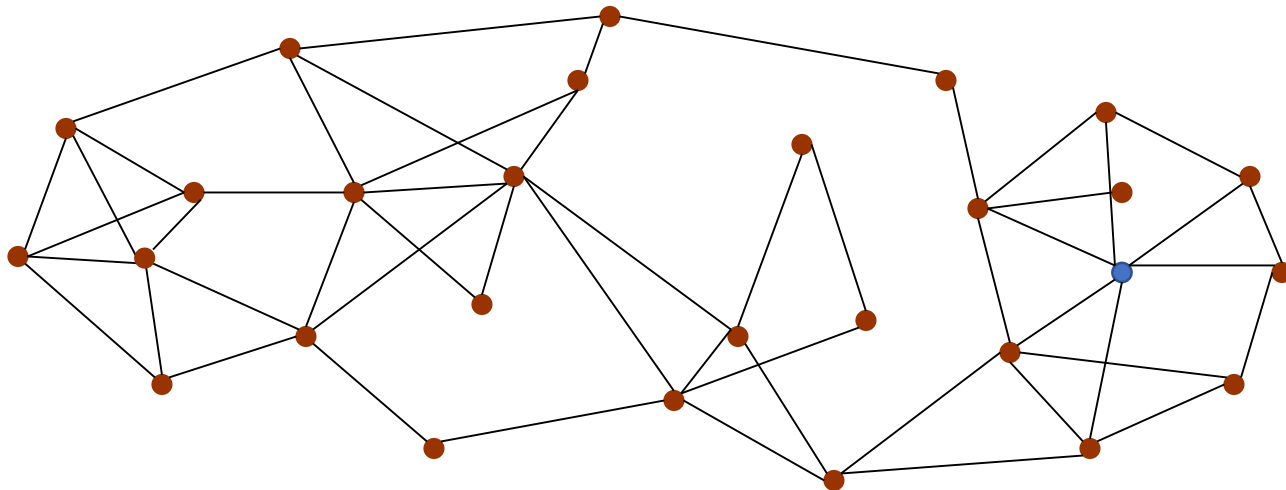


Data Structures

Programming Project #2

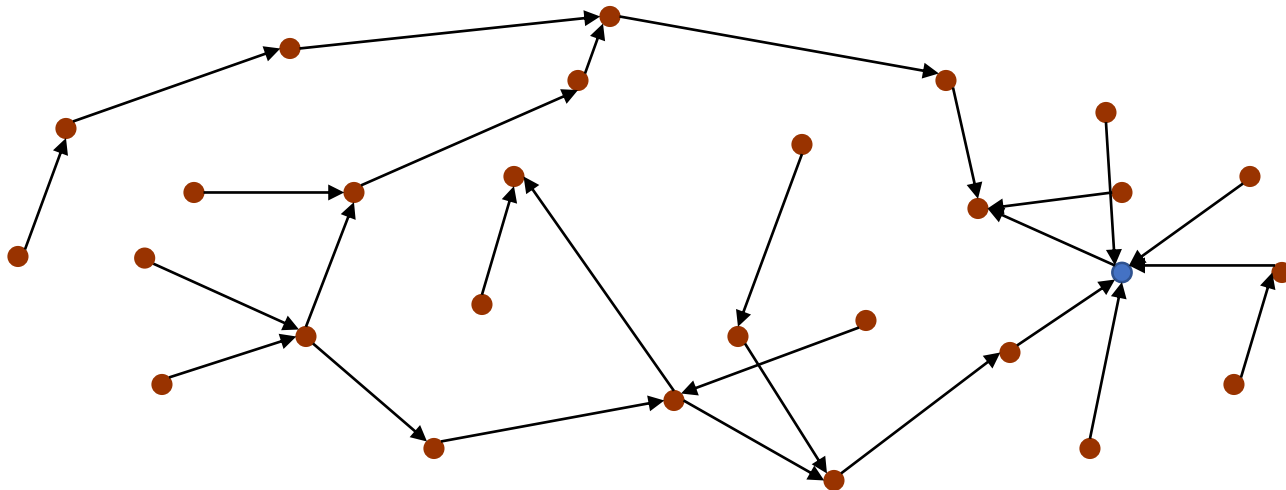
Background

- **Large-scale** sensor networks
- Every node senses the environment
- How to collect all the data to the **sink**?
Such as temperature



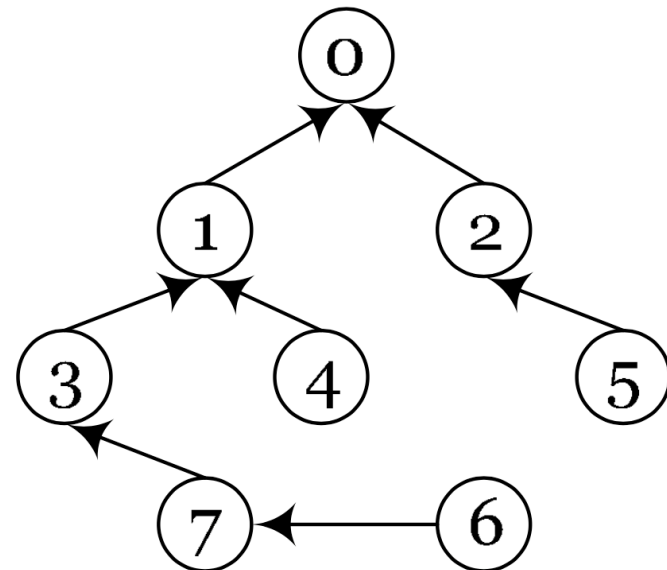
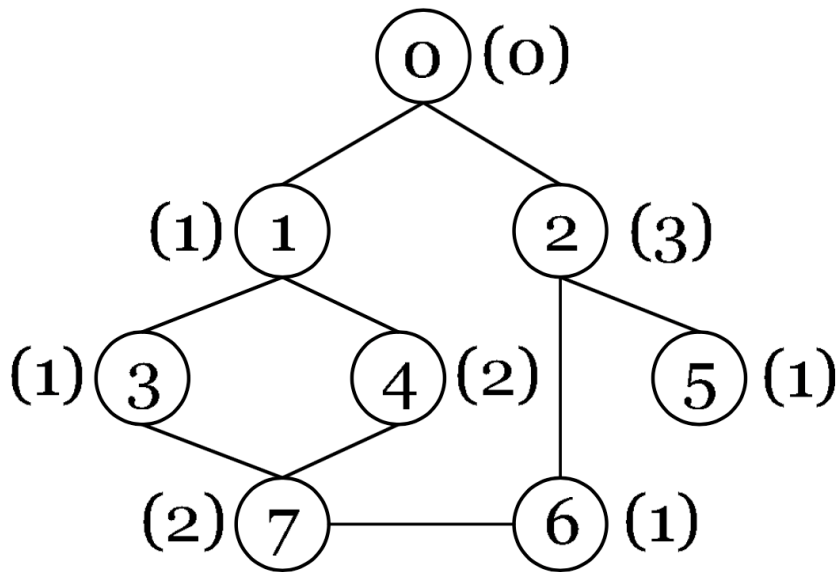
Why Aggregation Tree?

- Collect all the data to the **sink**?
- Limited storage and computation capability
- Routing on a **tree** → Simple to maintain
- Data **aggregation** → Save transmission energy



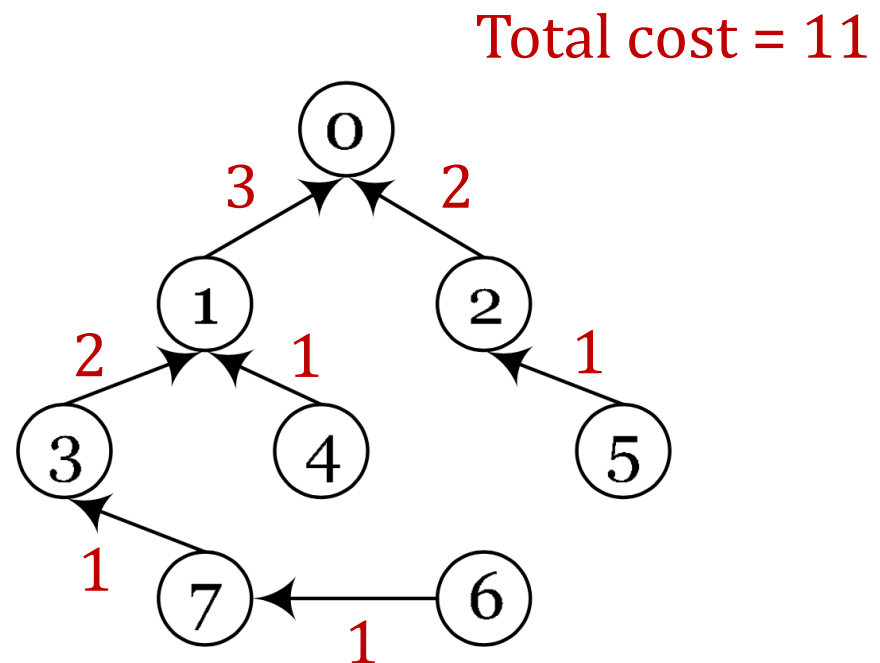
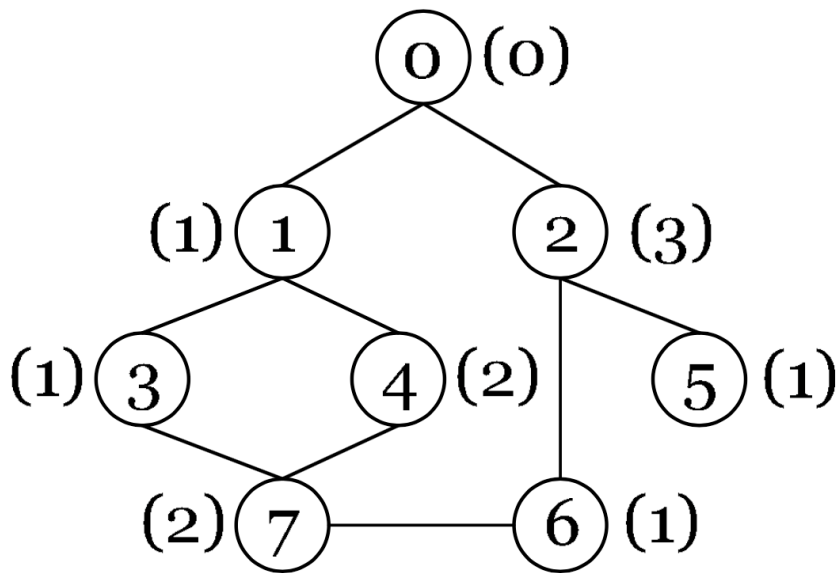
Example 1

- Routing on a **tree** → Simple to maintain
- Data **aggregation** → Save transmission energy
- Nodes' distinct data size (e.g., 1-5)
- Fixed packet size (e.g., **3**)



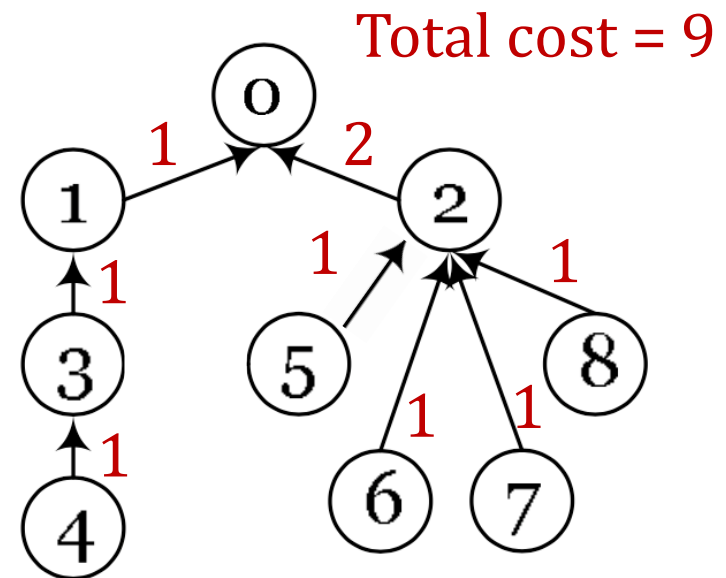
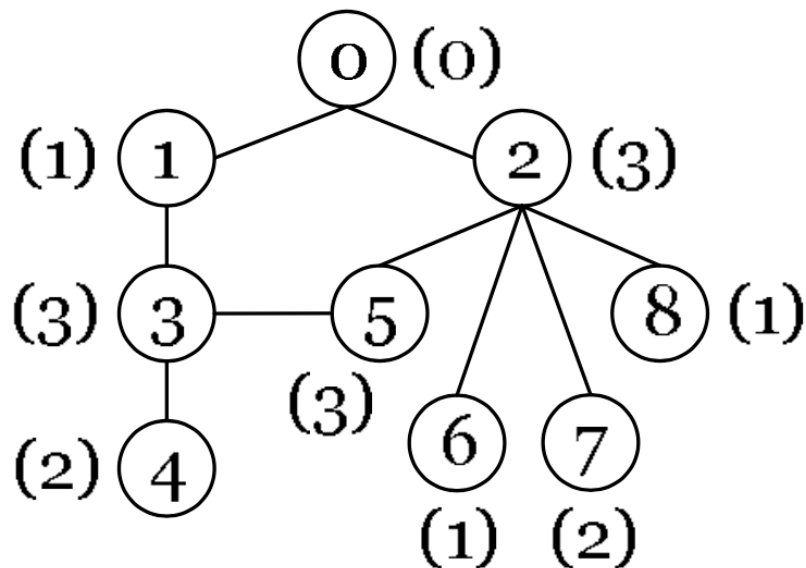
Example 1

- Routing on a **tree** → Simple to maintain
- Data **aggregation** → Save transmission energy
- Nodes' distinct data size (e.g., 1-5)
- Fixed packet size (e.g., **3**)



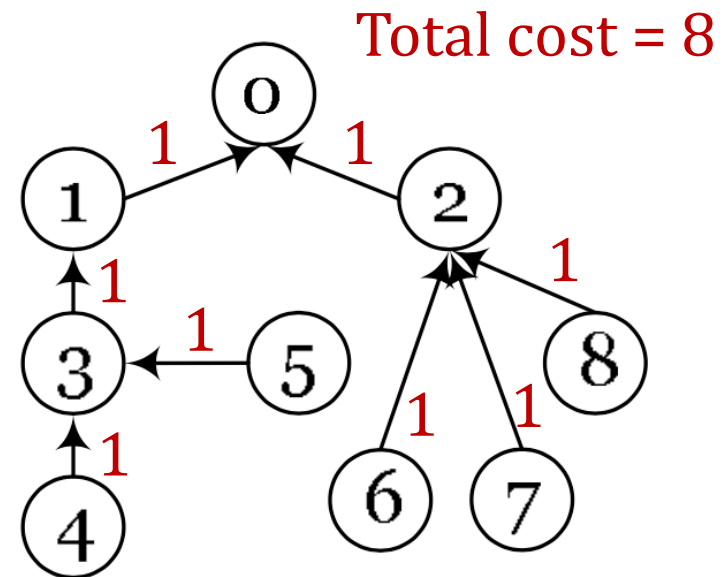
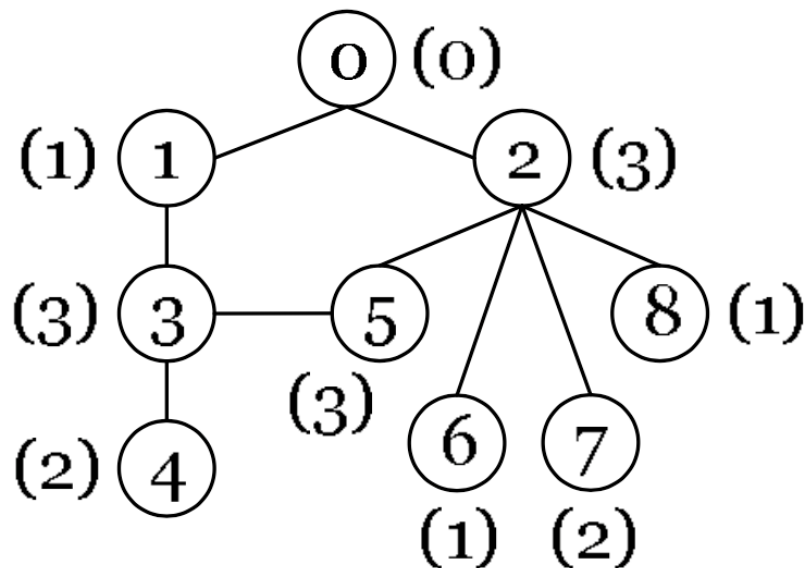
Example 2

- Routing on a **tree** → Simple to maintain
- Data **aggregation** → Save transmission energy
- Nodes' distinct data size (e.g., 1-5)
- Fixed packet size (e.g., **9**)



Example 2

- Routing on a **tree** → Simple to maintain
- Data **aggregation** → Save transmission energy
- Nodes' distinct data size (e.g., 1-5)
- Fixed packet size (e.g., **9**)



Bad News

- The problem is **NP-hard**
- We may not always find the optimal solution in polynomial time
- Alternatively, we aim at a **near-optimal solution**

Programming Project #2:

Construct an aggregation tree

- Input:
 - A node-weighted network $G = (V, E)$
 - Packet size
- Procedure:
 - Determine a set of links to construct a tree rooted at 0
 - Calculate the total cost
- Output:
 - Every node's parent in the constructed tree
 - Total cost
- The grade is inversely proportional to **the total cost**

Programming Project #2:

Construct an aggregation tree

- Input:
 - A node-weighted network $G = (V, E)$
 - Packet size
- Procedure:
 - Determine a set of links to construct a tree
 - Calculate the total cost
- Output:
 - Every node's parent in the constructed tree
 - Total cost
- The grade is inversely proportional to **the total cost**



Programming Project #2:

Construct an aggregation tree

- Input:
 - A node-weighted network $G = (V, E)$
 - Packet size
- Procedure:
 - Determine a set of links to construct a tree
 - Calculate the total cost
- Output:
 - Every node's parent in the constructed tree
 - Total cost
- The grade is inversely proportional to **the total cost**




歐都給？

Programming Project #2:

Construct an aggregation tree

- Input:
 - A node-weighted network $G = (V, E)$
 - Packet size
- Procedure:
 - Determine a set of links to construct a tree
 - Calculate the total cost
- Output:
 - Every node's parent in the constructed tree
 - Total cost
- The grade is inversely proportional to **the total cost**



又來這套
? ? ?

Programming Project #2: Construct an aggregation tree

- Input:

- A node-weight
- Packet size

- Procedure:

- Determine
- Calculate t

- Output:

- Every node
- Total cost

- The grade is



rooted at 0

the total cost

Programming Project #2: Construct an aggregation tree

- Input:

- A node
- Package

- Procedure

- Determine
- Calculate

- Output

- Every
- Total



oted at 0

- The grade is inversely proportional to the total cost

Programming P

Construct an ag

- Input:
 - A node-weighted
 - Packet size
- Procedure:
 - Determine a set
 - Calculate the tot
- Output:
 - Every node's par
 - Total cost
- The grade is inve



E)

at a tree rooted at 0

ted tree

nal to the total cost

Programming Project #2:

Construct an aggregation tree

- Input:
 - A node-weighted network $G = (V, E)$
 - Packet size
- Procedure:
 - Determine a set of links to construct a tree rooted at 0
 - Calculate the total cost
- Output:
 - Every node's parent in the constructed tree
 - Total cost
- We have a competition
- The grade is... (see next page)

The Competition

- The grade is inversely proportional to **the total cost**
- **Basic: 75 (deadline)**
 - A feasible aggregation tree
 - The correct total cost of the feasible one
- **Performance ranking** (decided after the deadline)
 - [0%, 50%) (bottom): +0
 - [50%, 75%): + 5
 - [75%, 90%): + 9
 - [90%, 95%): + 12
 - [95%, 100%] (top): + 15
- **Homework assistant** (superb deadline)
 - +10

The Competition

- The grade
- **Basic: 75**
 - A feasible
 - The cost
- **Performance**
 - [0%, 50%
 - [50%, 75%
 - [75%, 90%
 - [90%, 95%
 - [95%, 100%
- **Homework assistant** (superb deadline)
 - +10



the total cost

(deadline)

The Competition

- The grade
- **Basic: 75**
 - A feasible
 - The corr
- **Performance**
 - [0%, 50%
 - [50%, 75%
 - [75%, 90%
 - [90%, 95%
 - [95%, 100%
- **Homework assistant** (superb deadline)
 - +10



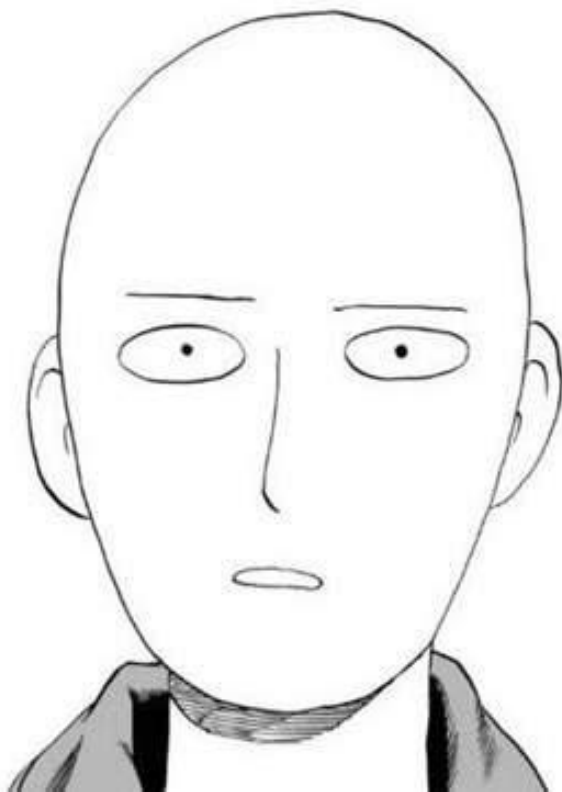
the total cost

We have
TIME LIMIT!



相信你們在做完作業以後

也變強了

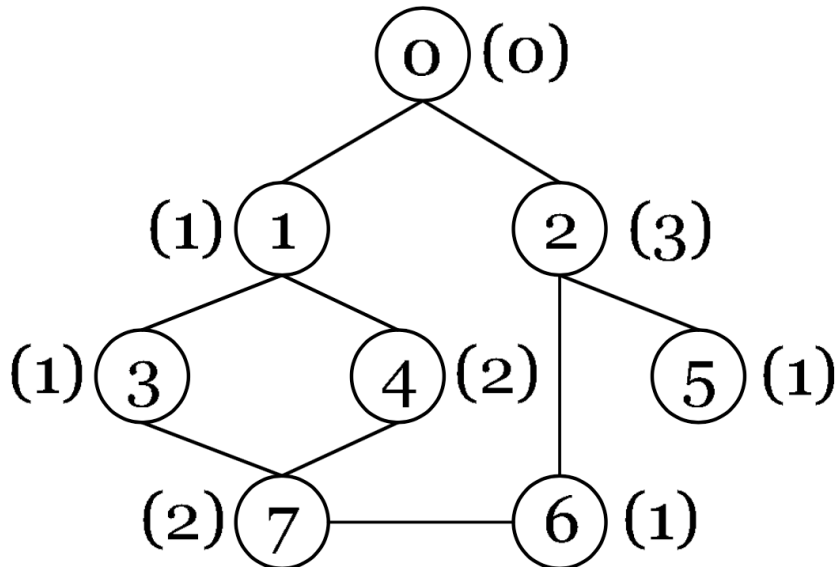


禿了

Input Sample: use scanf

Format:

#Nodes #Links Packet_Size
Node_ID Node_Weight
...
Link_ID Link_End1 Link_End2
...



Example:

8	9	3
0	0	
1	1	
2	3	
3	1	
4	2	
5	1	
6	1	
7	2	
0	0	1
1	0	2
2	1	3
3	1	4
4	2	5
5	2	6
6	3	4
7	4	7
8	6	7

Output Sample: use printf

Format:

#Nodes Total_Cost

Node_ID Parent_ID

...

Example:

8 11

0 0

1 0

2 0

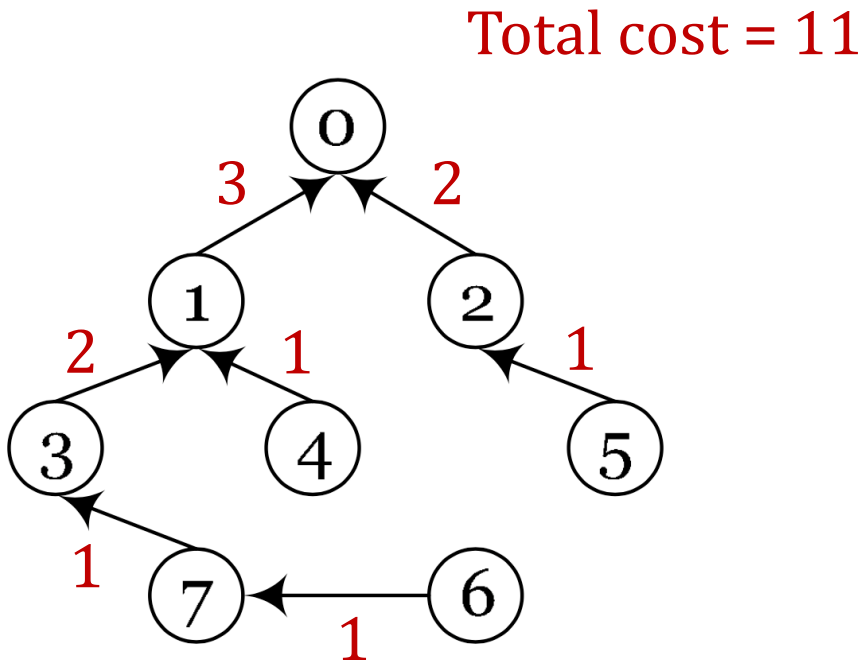
3 1

4 1

5 2

6 7

7 3



Note

- Superb deadline: 11/16 Tue
- Deadline: 11/23 Tue
- Pass the test of our [online judge](#) platform
- Submit your code to [E-course2](#)
- Demonstrate your code [remotely](#) with TA
- [C Source code \(i.e., only .c\)](#)
- Show a good programming style

Yesterday!!!

- Taiwanese Computer Science Day
- 1101101

