1 Introduction

Graph theory is a branch of mathematics, and graphs are the main research object of graph theory. **A graph** is a figure composed of a number of given vertices and edges connecting two vertices. This kind of graph is usually used to describe a certain specific relationship between certain things. Vertices are used to represent things, and edges connecting two vertices are used to indicate that there is such a relationship between two things.

Without specified mentioned, assume `arr` is declared as a signed integer array. The following code is added to the beginning of all C++ programs.

```
#include <bits/stdc++.h>
using namespace std;
```

2 Vector

Definition

`std::vector`

It is a memory-contiguous , **variable-length** array (also known as list) data structure provided by STL . It can provide linear complexity insertion and deletion, as well as constant complexity random access.

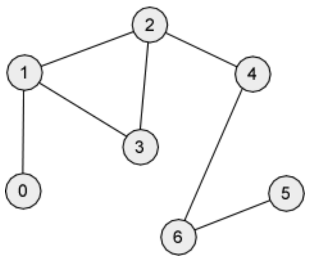
C++ Code Implementation

```
vector<int> V;

int main() {
    // Push elements
    V.push_back(1); V.push_back(2); V.push_back(3); V.push_back(4);

    // Pop elements
    V.pop_back();

    // Insert elements
    V.insert(V.begin()+2,8);
    // Elements inside V : {1,2,8,3}

    // Erase elements
    V.erase(V.begin()+1);
    // Elements inside V : {1,8,3}

    // Check if vector is empty
    if (V.empty()) cout << "vector is empty" << endl;

    // Check vector's size
    cout << V.size();
    // Clear the whole vector
    V.clear();
    return 0;
}
```

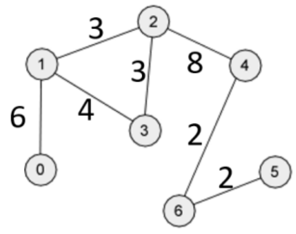
3 Graph

Definition

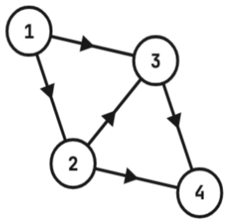
A graph is a figure composed of a number of given vertices and edges connecting two vertices. This kind of graph is usually used to describe a certain specific relationship between certain things. Vertices are used to represent things, and edges connecting two vertices are used to indicate that there is such a relationship between two things.



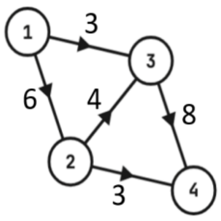
Unweighted Undirected Graph



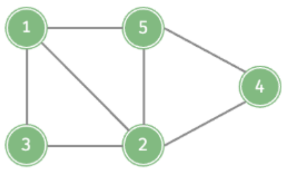
Weighted Undirected Graph



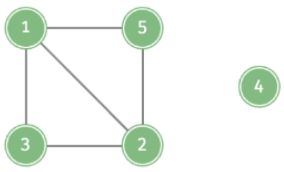
Unweighted Directed Graph



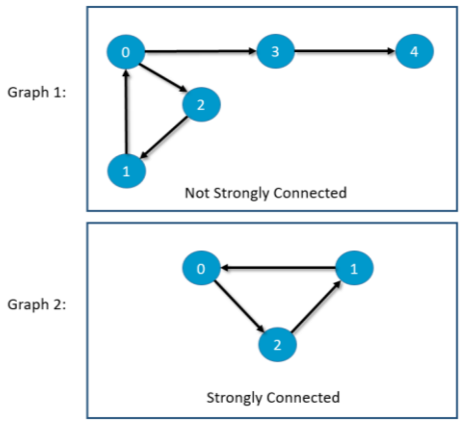
Weighted Directed Graph



Connected Graph



Disconnected Graph



Storage of Graphs

- Adjacency Matrix
  - Stores relationship between each pair of vertices
  - Implement by a 2D array
- Adjacency List
  - Stores edges
  - Implement by a 2D array-vector to lower space complexity

C++ Code Implementation

```
int adj_matrix[110][110];
vector<int> adj_list[110];
int N, M, u, v;

int main() {
    cin >> N >> M;    // N is number of vertices, M is number of edges
    for (int i = 1; i <= M; ++i)
    {
        cin >> u >> v;  // an edge from vertex u to vertex v
        adj_matrix[u][v] = 1;
        adj_list[u].push_back(v);
    }
    return 0;
}
```

Exercises

[CSP-J1 2022 Q9]

Consider a directed connected graph consisting of N vertices. When represented by the data structure of an adjacency matrix, there are at least ( ) non-zero elements in the matrix.

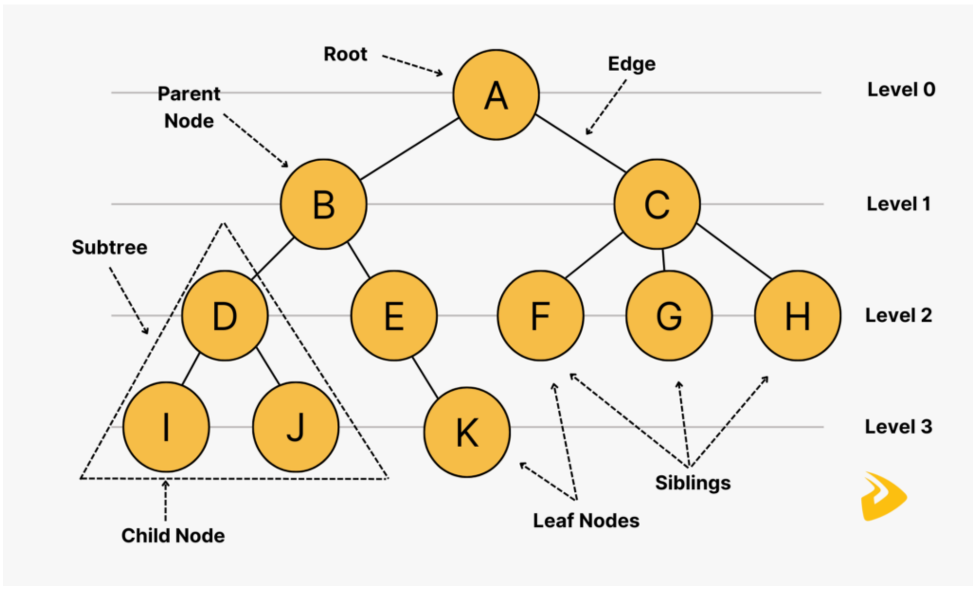
A. N - 1 B. N C. N + 1 D. N^2

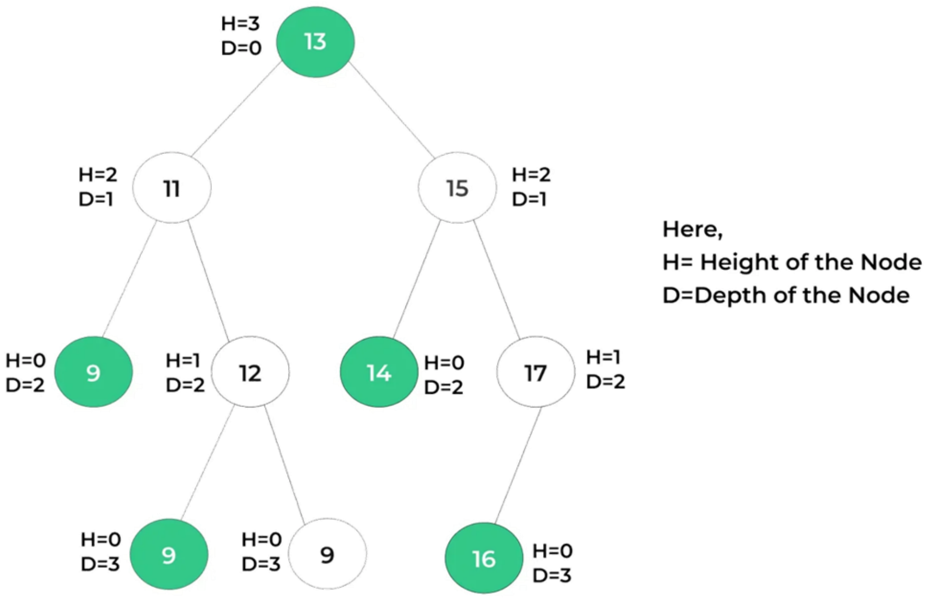
4 Tree

Definition

A tree is an **undirected connected graph** that doesn't have **cycles**. **If the tree has *n* vertices, the tree must have *n – 1* edges.**

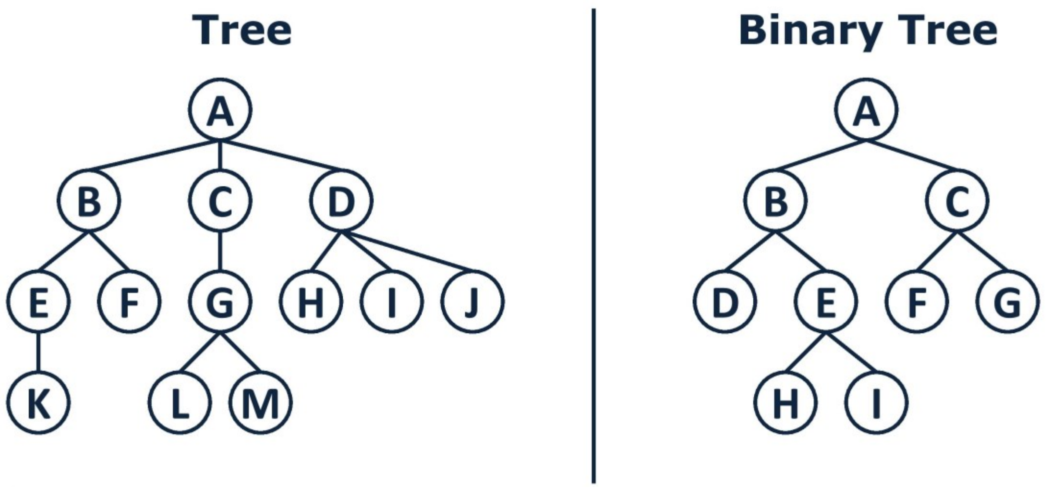
Structure of a Tree





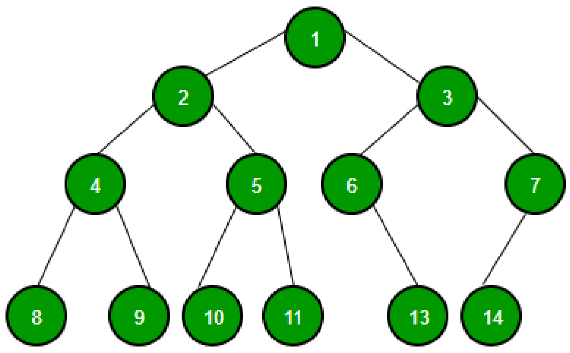
Binary Tree

A binary tree is a tree data structure in which each node has at most two children, referred to as the left child and the right child.



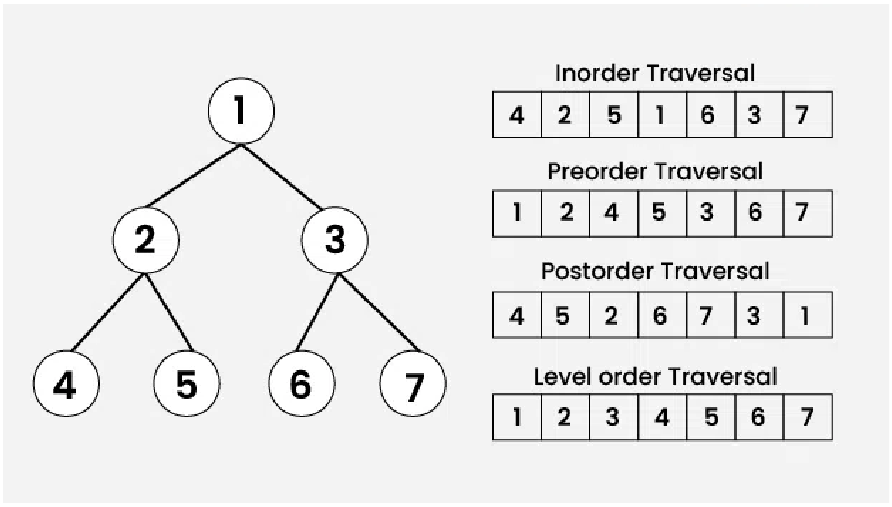
Storage of Binary Trees

- Adjacency List
- Linear Array
  - for each node N, N's left child $= 2N$, N's right child $= 2N + 1$.



Traverse of Binary Trees

- Inorder Traversal 中序遍歷（左-根-右）
- Preorder Traversal 先序遍歷（根-左-右）
- Postorder Traversal 後序遍歷（左-右-根）
- Level order Traversal 層序遍歷



Exercises

[CSP-J1 2023 Q11]

Given a binary tree, the result of its pre-order traversal is: ABDECFG, and the result of its in-order traversal is: DEBACFG. What is the correct post-order traversal result of this tree?

A.	EDBFGCA	B.	EDBGCFA
C.	DEBGFCA	D.	DBEGFCA

[CSP-J1 2022 Q8]

A complete binary tree with n nodes is stored and represented by an array. It is known that the root node is stored in the first position of the array. If the node stored at the 9th position of the array has a sibling node and two child nodes, the positions of its sibling node and right child node are ( ).

A. 8, 18 B. 10, 18 C. 8, 19 D. 10, 19