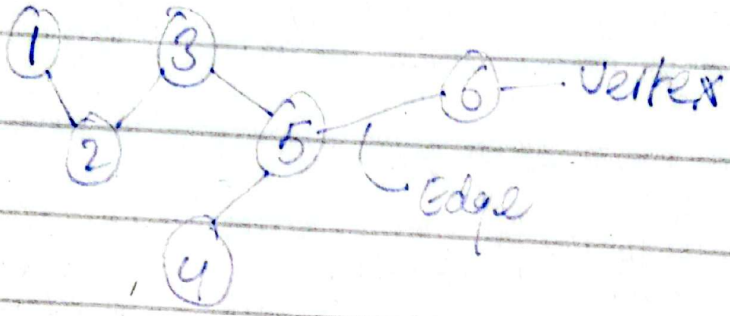
Graphs

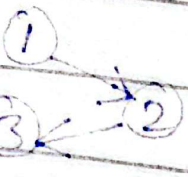
- There is no hierarchy
- Nodes are simply connected
  (connection is called edge)
- Node is called vertex/vertices.



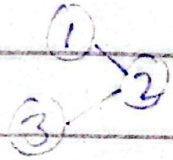
Types → Based on Edges

1) Directional

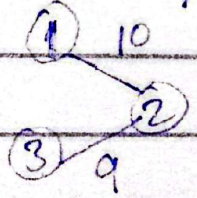
1 → 2 Unidirectional.



2) Undirectional (Bi-Directed)



Types weight (value Based)

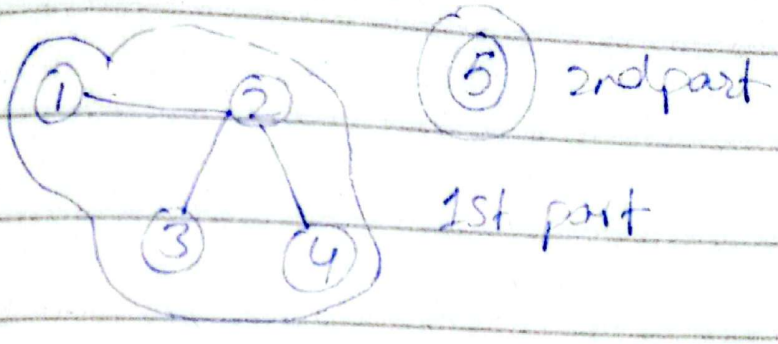


weight can be +ve/-ve

1) Undirected Unweighted
2) Undirected Weighted
3) Dir Weighted
4) Dir Unweighted

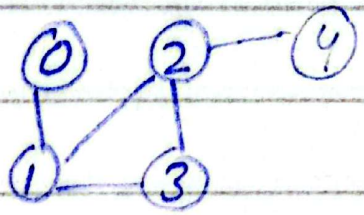
① Connected

② Disconnected



→ Adjancey List (Building) -

List → Doubly LL



0 → [1]

1 → [0,2,3]

2 → [1,3]

3 → [1,2]

List int

[1 | 0,2,3 | 1,3 | 1,2]

Use Dynamic Array

Building

```
class Graph(){
int V
list<int> *L
Graph(int v){
this->V = V
L = new list<int>[V]
addEdge(){int u, int v)
L[u] = push_back(v)
L[v] = push_back(u)
```

```
printAdjList(){
for(i=0; i<V; i++){
cout<<i<<":";
for(int neigh : L[i]){
cout<<neigh<<" "
}
cout<<endl;
}}
```