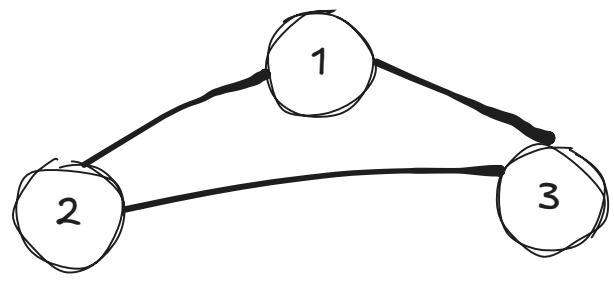
Implementation of Graph using List

1. unordered_list< int , list<int>> adj;

This will create the mapping, where we have value of each node and the particular node is mapped to values of the nodes which are adjacent to it.

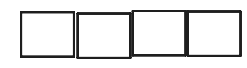


1 -> 2,3
2 -> 1,3
3 -> 1,2

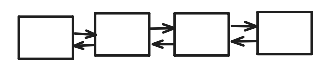
This is the unordered_map.

What is the difference in a list and vector?

- vector is contiguous in memory and is used when you need fast indexing, it is very bad in insertions and deletions as it needs to get all the element to get shifted to insert the upcoming element.

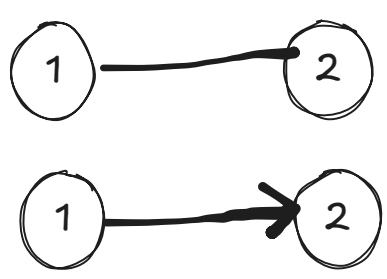
vector : 

- list is the chain of elements, where each element is connected via chains or pointers, It is bad for indexing and is O(n) as we need to track the chain to get to the element, It is mainly useful in the insertions and deletions as manipulation of pointers at the specific position is the only task to be done.

list : 

2 . void addEdge(int u, int v, bool direction)

To add the edge, we just need to pass the two endpoints u and v and then pass the direction which is 0 or 1, 1 stands for directed graph and 0 stands for undirected graph.



addEdge(1,2,0)

addEdge(1,2,1)

this is directed graph, 1 is connected to 2, but 2 is not connected to 1.
adj[u].push_back(v);

this is undirected, so for key=2 we should have 1 and for the key=1 we should have 2.
adj[u].push_back(v);
adj[v].push_back(u);

to incorporate this logic, we do

```
if (direction = 0) {
    adj[v].push_back(u);
}
```

What is adj[u].push_back[v] ?

adj is the map, adj[u] makes you to get the access of the list at key/node u and in that you do push_back(v) like this key / node is connected to v.

3. printing the graph

Now that, we have, the unordered map, we now need to access that map nicely and print the map.

```
for( auto i : adj ){
    cout << i.first << " ->" ;
    for( auto j : i.second) {
        cout << j << " ";
    }
    cout << endl;
}
```

i -> iterator of the map
i.first -> gives a particular key of the map
i.second - > gives the access of the list correspoding to that key
j is the iterator of that list