(1) Passport (4) Visa (5) Ticket
(2) Gifts (6) Insurance
(3) Forex

Passport → Visa → Ticket → Gifts
Insurance Forex

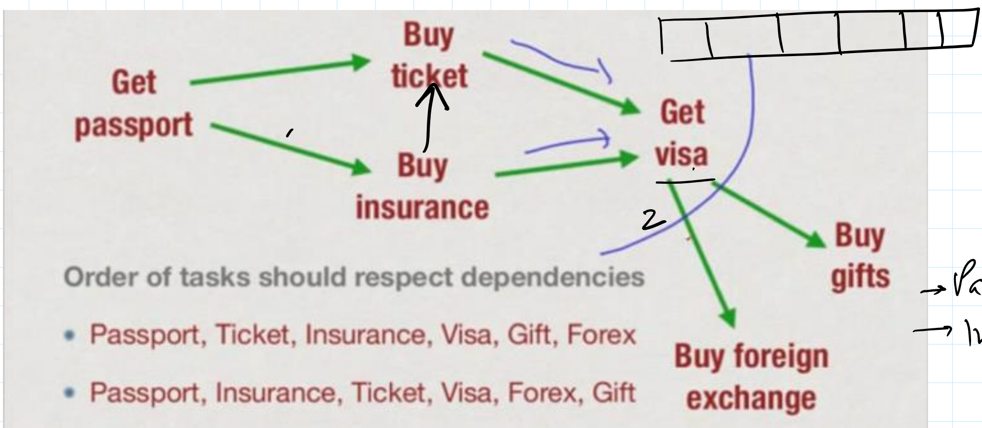
1) Passport → Visa → Ticket → Insurance → Gifts → Forex
2) Passport → Visa → Insur → Ticket → Gifts → Forex

(1) Directed Graph
(2) Acyclic

Tarjan Algo

```
functn toposort(src, visited) {
    visited = ✓
    for (every ngbr) {
        if (not visited)
            topological sort(ngbr);
    }
    cout<<print  or  ans.push_back
                     ↳ global var.
}
```

ans = rev seq => rev then sorted seq.



→ Pass
→ Ins

InDegree

Pass	Tic	Ins	Visa	Gifts	Exc
0	0	0	1	1	1

G E
V

"Passport" : "Ticket" : Insur : V G Ex.

① Compute indegree of each vtx

② Push all vtx with indegree zero into queue

③ while (!q.empty) {

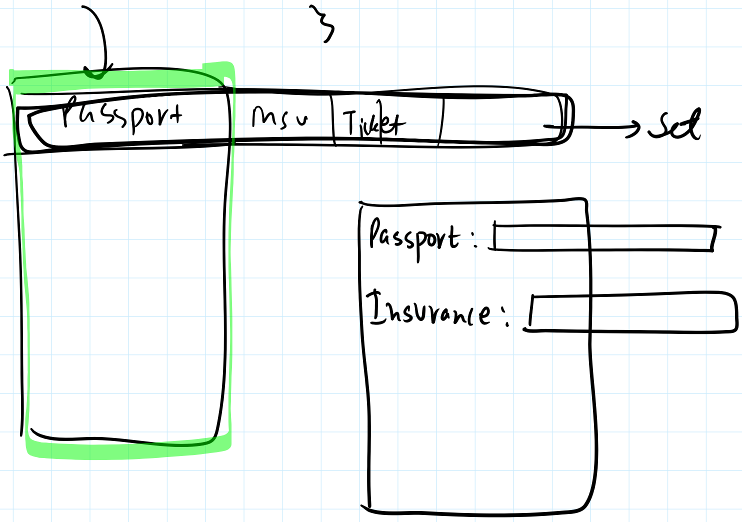
pop();

store;

↓ indegree of every nghr {

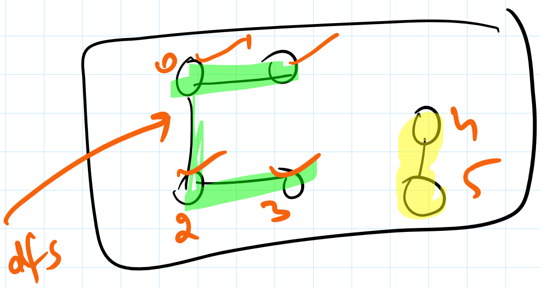
if indegree become 0 → push into queue.

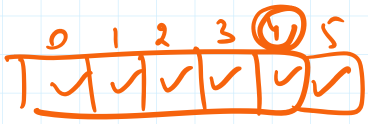
}



① g++ -std=c++11 <filename.cpp>

② a.exe ./a.out





2.

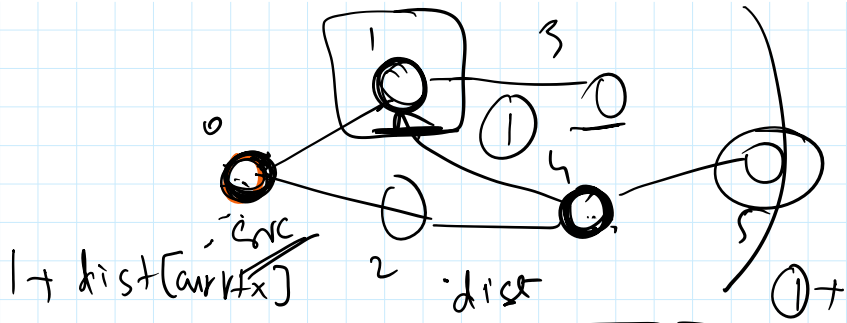
cc ( ) {

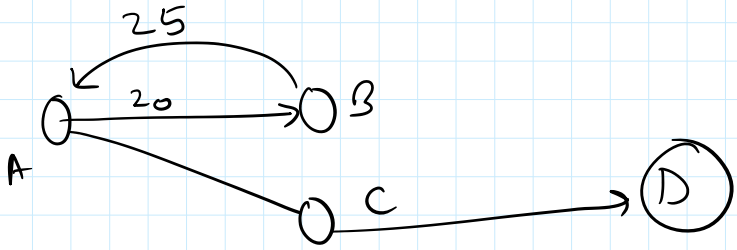
for (every non visited) put a dfs

++cnt.

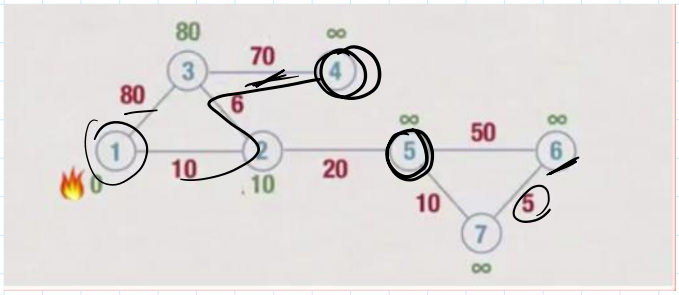
}

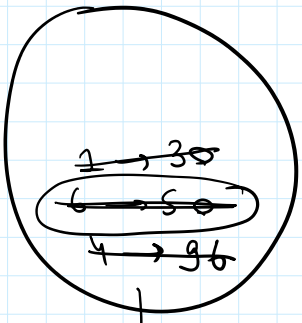
dfs

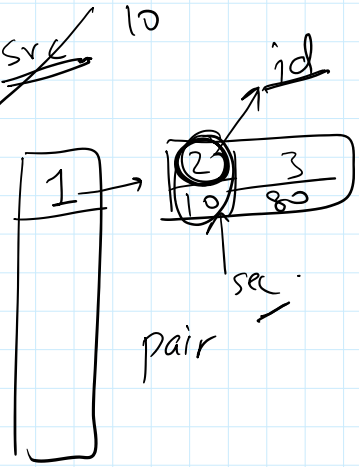


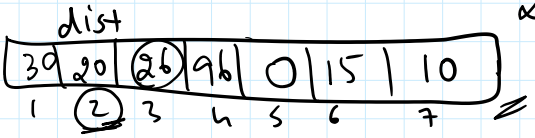


src is a macro whose val is 10










```
dijkstra(src, dest) {
    minheap; dist; d[src] = 0;   q.push(src);
  while(!q.empty()) {
      currVtx = pop;
      if (dist[currVtx] < currVtx.dist) continue;
dist[currVtx]=   for (every ngbr) {
          ngbrdist = dist[currVtx] + dist of NgbrFromCurrVtx
      if (ngbrdist < dist[ngbr]) {
          q.push();   → unvisited
  }
}
```

0 → q[1][0]

} }

return dist[dest] }