Sortowanie Topologiczne

Problem:

- Dany jest skierowany, acykliczny graf (DAG) G. Należy ustawić wierzchołki G w takiej kolejności, że jeśli istnieje krawędź $v \to u$ to wierzchołek v stoi przed wierzchołkiem u. Wierzchołki z G można posortować topologicznie na jeden lub więcej sposobów.

Jeśli G zawiera cykl, to nie da się go posortować topologicznie, ponieważ jeśli wierzchołki a i b należą do cyklu, to:

- a powinien stać przed b, bo istnieje ścieżka z a do b
- b powinien stać przed a bo istnieje ścieżka z b do a
  A zatem mamy sprzeczność.

Sortowanie Topologiczne przy użyciu BFS (Algorytm Kahn'a):
Definicja:

- $DEG_{in}(v)$ - liczba wchodzących krawędzi do wierzchołka v

Obserwacja:

- Pierwszy w kolejności topologicznej wierzchołek ma $DEG_{in} = 0$

A zatem jeśli v ma $DEG_{in} = 0$ to będzie on stał przed wszystkimi innymi wierzchołkami z G. Teraz jeśli usuniemy v z G to zmniejszymy DEG_{in} dla wszystkich wierzchołków na które wskazywał v i co najmniej jeden z nich będzie miał teraz $DEG_{in} = 0$. Więc powtarzamy całą procedurę tak długo, jak $G \neq \varnothing$.

Przebieg algorytmu:

1. Znajdź w G wierzchołek z $DEG_{in} = 0$ i dodaj go do kolejki
2. Wyjmij z kolejki wierzchołek v i wstaw go do wynikowej tablicy
3. Usuń v z grafu
4. Wróć do kroku 1)

Kod:

```
queue<int> Q;
for (int i=1; i<=n; i++)
        if (deg[i] == 0) Q.push(i);

vector<int> toposorted;
while (!Q.empty()) {
        int v = Q.front(); Q.pop();
```

```
        toposorted.PB(v);
        for (auto u : G[v]) {
                deg[u]--;
                if (deg[u] == 0)
                        Q.push(u);
        }
}
```

Sortowanie Topologiczne przy użyciu DFS:

Zauważmy że podczas przechodzenia grafu DFS-em każdy wierzchołek v można pokolorować na 3 kolory:

1. Biały: v nie był jeszcze odwiedzony
2. Szary: v znajduje się na stosie DFS
3. Czarny: v jest odwiedzony i został zdjęty ze stosu

Załóżmy, że jesteśmy DFS-em w wierzchołku v. Teraz jeśli v wskazuje na wierzchołek u to jest on Biały lub Czarny (Gdyby był Szary to w grafie jest cykl).

Zauważmy, że dany wierzchołek staje się Czarny dopiero wtedy, kiedy:

- nie wskazuje na żaden inny wierzchołek
- wszystkie wierzchołki na które wskazuje są Czarne

A zatem wystarczy przejść się po grafie DFS-em i dodawać każdy wierzchołek do wynikowej listy w momencie kiedy staje się Czarny. Na koniec należy odwrócić wynikową listę.

Kod:

```
void dfs(int v) {
        vis[v] = true;
        for (auto u : G[v])
        if (!vis[u]) dfs(u);
        toposorted.PB(v);
}
```

W main-ie:

```
for (int v=1; v<=n; v++)
        if (!vis[v]) dfs(v);
reverse(toposorted.begin(), toposorted.end());
```