## Objective

To enumerate all the permutations of n distinct objects, numbered 1…n, which satisfies a given set of precedence constraints. A precedence constraints *(a, b)* means that in the permutations *a* should appear before *b*.

## Implementation

Algorithm:

We have implemented the algorithm as described in [1] by **Donald E. Knuth and Jayme L. Szwarcfiter** which generate all the topological sorting arrangements.

A brief description of the algorithm is given below.

**Input:**  are n distinct objects and set of constraints (which means precede in all the permutations.

**Output:** All the possible permutations of that satisfy given constraints.

**Global Variables:**

* Arrays of N Nodes. Each node corresponding to each object. A node contains count of nodes it succeeds and a pointer to the list of nodes it precedes
* A deque (D) which stores nodes whose count is 0;
* An output array which contains the current permutation.

**Procedure:**

Generate (K) {

// this procedure will output all the permutations which begin with a sequence that has already been output.

If (D is not empty) {

base <- rightmost element of D;

do {

q <- rightmost element of D;

Delete it from D;

Erase all relations of the type ;

output [k+1] <- q;

if (K=N-1) {

print output array;

new line;

}

Generate (K+1)

Retrieve all the relations of the form ;

Insert q at the left of D;

} while (rightmost element of D is not equal to base);

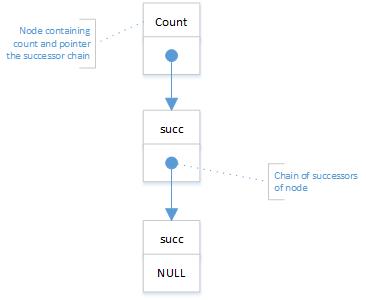
}

}

Data Structures:

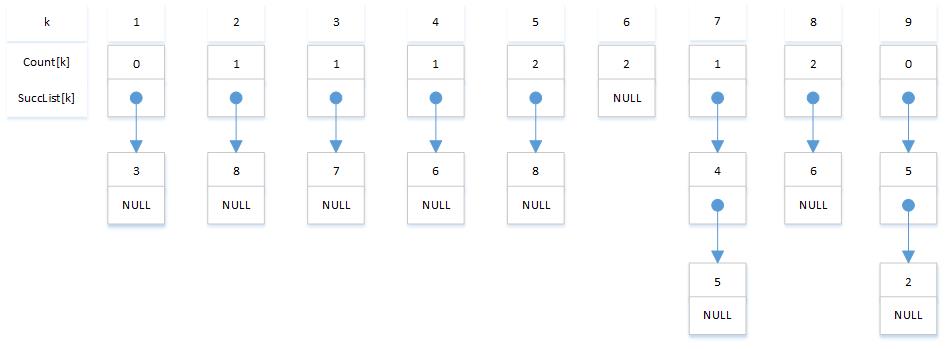
The most important data structure used is the Node. A node contains count of nodes it succeeds and a list of nodes it precedes. The idea for this has been taken from [2]

Below diagram shows the structure of a single node.



For example, for 9 objects numbered {1, 2, 3, 4, 5, 6, 7, 8, 9} and precedence relations as

So the array of N nodes will be like one shown below with k as index.



Extra work:

* The program do not assume that precedence means that q < j. It works even if q > j. For example if 4 then in the permutations 4 will appear before 3.

## References

[1] “A Structured Program to Generate All Topological Sorting Arrangements” by Donald E. Knuth and Jayme L. Szwarcfiter. <http://www.cs.ncl.ac.uk/publications/trs/papers/61.pdf>

[2] “The Art of Computer Programming: Fundamental Algorithms (Volume 1) “ By Donald E. Knuth.