# Greedy Algorithms | Set 2 (Kruskal’s Minimum Spanning Tree Algorithm)

*What is Minimum Spanning Tree?*
Given a connected and undirected graph, a *spanning tree* of that graph is a subgraph that is a tree and connects all the vertices together. A single graph can have many different spanning trees. A *minimum spanning tree (MST)* or minimum weight spanning tree for a weighted, connected and undirected graph is a spanning tree with weight less than or equal to the weight of every other spanning tree. The weight of a spanning tree is the sum of weights given to each edge of the spanning tree.

*How many edges does a minimum spanning tree has?*
A minimum spanning tree has (V – 1) edges where V is the number of vertices in the given graph.

*What are the applications of Minimum Spanning Tree?*
See this for applications of MST.

Below are the steps for finding MST using Kruskal's algorithm

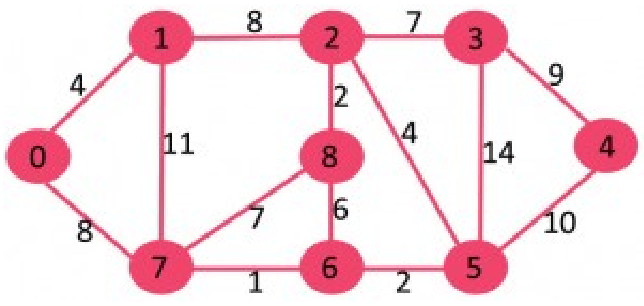
```
1. Sort all the edges in non-decreasing order of their weight.

2. Pick the smallest edge. Check if it forms a cycle with the spanning tree
formed so far. If cycle is not formed, include this edge. Else, discard it.

3. Repeat step#2 until there are (V-1) edges in the spanning tree.
```

The step#2 uses Union-Find algorithm to detect cycle. So we recommend to read following post as a prerequisite.
Union-Find Algorithm | Set 1 (Detect Cycle in a Graph)
Union-Find Algorithm | Set 2 (Union By Rank and Path Compression)

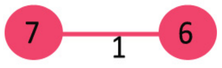
The algorithm is a Greedy Algorithm. The Greedy Choice is to pick the smallest weight edge that does not cause a cycle in the MST constructed so far. Let us understand it with an example: Consider the below input graph.



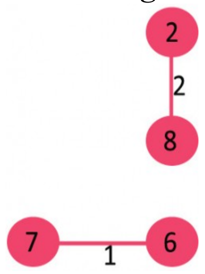
The graph contains 9 vertices and 14 edges. So, the minimum spanning tree formed will be having (9 – 1) = 8 edges.

```
After sorting:
Weight   Src    Dest
1         7      6
2         8      2
2         6      5
4         0      1
4         2      5
6         8      6
7         2      3
7         7      8
8         0      7
8         1      2
9         3      4
10        5      4
11        1      7
14        3      5
```

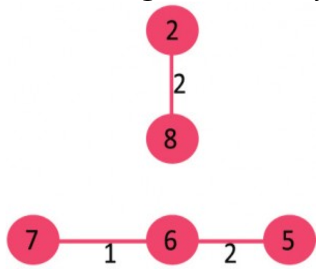
Now pick all edges one by one from sorted list of edges
**1.** *Pick edge 7-6:* No cycle is formed, include it.



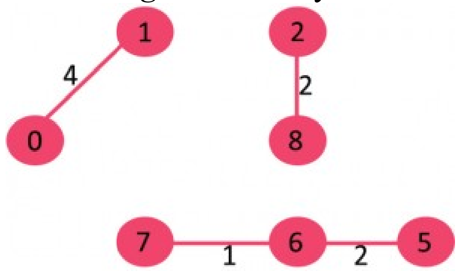
**2.** *Pick edge 8-2:* No cycle is formed, include it.



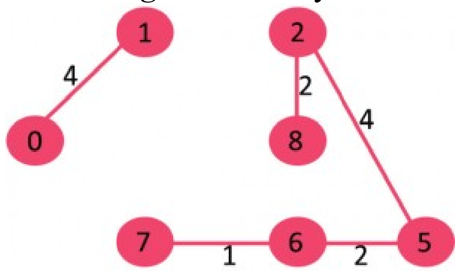
**3.** *Pick edge 6-5:* No cycle is formed, include it.



**4.** *Pick edge 0-1:* No cycle is formed, include it.

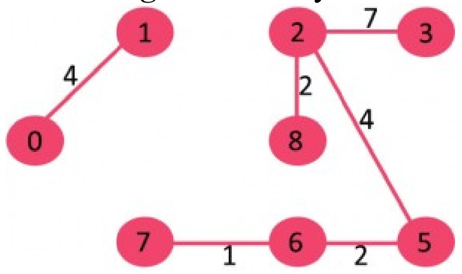


**5.** *Pick edge 2-5:* No cycle is formed, include it.



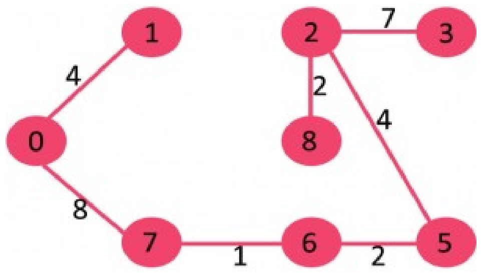
**6.** *Pick edge 8-6:* Since including this edge results in cycle, discard it.

**7.** *Pick edge 2-3:* No cycle is formed, include it.



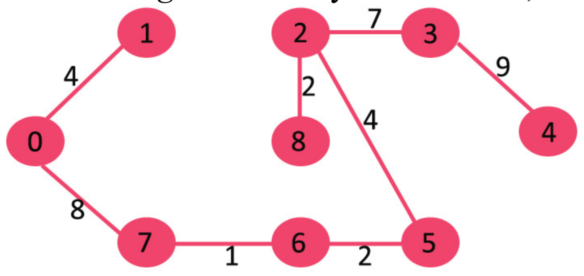
**8.** *Pick edge 7-8:* Since including this edge results in cycle, discard it.

**9.** *Pick edge 0-7:* No cycle is formed, include it.



**10.** *Pick edge 1-2:* Since including this edge results in cycle, discard it.

**11.** *Pick edge 3-4:* No cycle is formed, include it.



Since the number of edges included equals (V – 1), the algorithm stops here.

```
// Kruskal's algortihm to find Minimum Spanning Tree of a given connected,
// undirected and weighted graph
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// a structure to represent a weighted edge in graph
struct Edge
{
    int src, dest, weight;
};

// a structure to represent a connected, undirected and weighted graph
struct Graph
{
    // V-> Number of vertices, E-> Number of edges
    int V, E;

    // graph is represented as an array of edges. Since the graph is
    // undirected, the edge from src to dest is also edge from dest
    // to src. Both are counted as 1 edge here.
    struct Edge* edge;
};

// Creates a graph with V vertices and E edges
struct Graph* createGraph(int V, int E)
{
    struct Graph* graph = (struct Graph*) malloc( sizeof(struct Graph) );
    graph->V = V;
    graph->E = E;
    graph->edge = (struct Edge*) malloc( graph->E * sizeof( struct Edge ) );

    return graph;
}

// A structure to represent a subset for union-find
struct subset
{
    int parent;
    int rank;
};

// A utility function to find set of an element i
// (uses path compression technique)
int find(struct subset subsets[], int i)
{
    // find root and make root as parent of i (path compression)
    if (subsets[i].parent != i)
        subsets[i].parent = find(subsets, subsets[i].parent);

    return subsets[i].parent;
}

// A function that does union of two sets of x and y
// (uses union by rank)
void Union(struct subset subsets[], int x, int y)
{
    int xroot = find(subsets, x);
    int yroot = find(subsets, y);

    // Attach smaller rank tree under root of high rank tree
    // (Union by Rank)
    if (subsets[xroot].rank < subsets[yroot].rank)
        subsets[xroot].parent = yroot;
    else if (subsets[xroot].rank > subsets[yroot].rank)
        subsets[yroot].parent = xroot;

    // If ranks are same, then make one as root and increment
    // its rank by one
    else
    {
        subsets[yroot].parent = xroot;
        subsets[xroot].rank++;
    }
}

// Compare two edges according to their weights.
// Used in qsort() for sorting an array of edges
int myComp(const void* a, const void* b)
{
    struct Edge* a1 = (struct Edge*)a;
    struct Edge* b1 = (struct Edge*)b;
    return a1->weight > b1->weight;
```

```
}

// The main function to construct MST using Kruskal's algorithm
void KruskalMST(struct Graph* graph)
{
    int V = graph->V;
    struct Edge result[V];  // Tnis will store the resultant MST
    int e = 0;  // An index variable, used for result[]
    int i = 0;  // An index variable, used for sorted edges

    // Step 1:  Sort all the edges in non-decreasing order of their weight
    // If we are not allowed to change the given graph, we can create a copy
    // array of edges
    qsort(graph->edge, graph->E, sizeof(graph->edge[0]), myComp);

    // Allocate memory for creating V ssubsets
    struct subset *subsets =
        (struct subset*) malloc( V * sizeof(struct subset) );

    // Create V subsets with single elements
    for (int v = 0; v < V; ++v)
    {
        subsets[v].parent = v;
        subsets[v].rank = 0;
    }

    // Number of edges to be taken is equal to V-1
    while (e < V - 1)
    {
        // Step 2: Pick the smallest edge. And increment the index
        // for next iteration
        struct Edge next_edge = graph->edge[i++];

        int x = find(subsets, next_edge.src);
        int y = find(subsets, next_edge.dest);

        // If including this edge does't cause cycle, include it
        // in result and increment the index of result for next edge
        if (x != y)
        {
            result[e++] = next_edge;
            Union(subsets, x, y);
        }
        // Else discard the next_edge
    }

    // print the contents of result[] to display the built MST
    printf("Following are the edges in the constructed MST\n");
    for (i = 0; i < e; ++i)
        printf("%d -- %d == %d\n", result[i].src, result[i].dest,
                                                 result[i].weight);
    return;
}
```

```
// Driver program to test above functions
int main()
{
    /* Let us create following weighted graph
             10
        0--------1
        |  \     |
       6|   5\   |15
        |      \ |
        2--------3
            4       */
    int V = 4;  // Number of vertices in graph
    int E = 5;  // Number of edges in graph
    struct Graph* graph = createGraph(V, E);


    // add edge 0-1
    graph->edge[0].src = 0;
    graph->edge[0].dest = 1;
    graph->edge[0].weight = 10;

    // add edge 0-2
    graph->edge[1].src = 0;
    graph->edge[1].dest = 2;
    graph->edge[1].weight = 6;

    // add edge 0-3
    graph->edge[2].src = 0;
    graph->edge[2].dest = 3;
    graph->edge[2].weight = 5;

    // add edge 1-3
    graph->edge[3].src = 1;
    graph->edge[3].dest = 3;
    graph->edge[3].weight = 15;

    // add edge 2-3
    graph->edge[4].src = 2;
    graph->edge[4].dest = 3;
    graph->edge[4].weight = 4;

    KruskalMST(graph);

    return 0;
}
```

```
Following are the edges in the constructed MST
2 -- 3 == 4
0 -- 3 == 5
0 -- 1 == 10
```

**Time Complexity:** O(ElogE) or O(ElogV). Sorting of edges takes O(ELogE) time. After sorting, we iterate through all edges and apply find-union algorithm. The find and union operations can take atmost O(LogV) time. So overall complexity is O(ELogE + ELogV) time. The value of E can be atmost V^2, so O(LogV) are O(LogE) same. Therefore, overall time complexity is O(ElogE) or O(ElogV)

References:
http://www.ics.uci.edu/~eppstein/161/960206.html
http://en.wikipedia.org/wiki/Minimum_spanning_tree

This article is compiled by Aashish Barnwal and reviewed by GeeksforGeeks team. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

## Related Topics:

- Assign directions to edges so that the directed graph remains acyclic
- K Centers Problem | Set 1 (Greedy Approximate Algorithm)
- Find the minimum cost to reach destination using a train
- Applications of Breadth First Traversal
- Optimal read list for given number of days
- Print all paths from a given source to a destination
- Minimize Cash Flow among a given set of friends who have borrowed money from each other
- Boggle (Find all possible words in a board of characters)

Tags: Graph, Greedy Algorithm, Kruskal, Kruskal'sAlgorithm

Tweet 5 +1 4

**Writing code in comment?** Please use **ideone.com** and share the link here.

**52 Comments** GeeksforGeeks 1 Login

Recommend Share Sort by Newest

Join the discussion…

**siddharth rajpal** • 5 days ago

instead of using a union find algorithm, can one use an array to denote 1 for all nodes in the MST and 0 for those nodes that aren't a part of the MST. Whenever we encounter a new edge, we see if both the nodes (the source and the destination) are 1 or not. If they are both 1 then they form a cycle, otherwise they do not. This should take O(1) time, even better than O(log n) time by the union-find algorithm. Please tell me if i am wrong.

• Reply • Share ›

**Pushpinder Garg** siddharth rajpal • 5 days ago

You are assuming that MST is formed .

• Reply • Share ›

**Andy Toh** • 9 days ago

My compile-time solution, whose output agrees with the solution above.

http://ideone.com/C6Fyec

• Reply • Share ›

**alex** • 17 days ago

Please double check for errors when the set is too large and check for memory leaks.

• Reply • Share ›

**krishna** • 20 days ago

here is the simple c++ implementation using STL

http://competitiveprogrammer.b...

• Reply • Share ›

**prashant jha** • a month ago

http://uva.onlinejudge.org/ext...

my sol:http://ideone.com/kGCJqS

• Reply • Share ›

**Techie Me** • a month ago

Here is some detailed discussion on the category of this problem and a simplified version of analysis of the algorithm.

http://techieme.in/minimum-spa...

• Reply • Share ›

**Sanghwa Jung** • 4 months ago

my javacode is http://javamusician.blogspot.k...

1 • Reply • Share ›

**Kshitij Bhutani** • 7 months ago

plz explain me what is mycomp function doing here?

• Reply • Share ›

**Anurag Singh** Kshitij Bhutani • 6 months ago

In Kruskal's Algorithm, we need to sort the edges. Here "qsort" function is used for that. "qsort" function needs a "compare" function as an argument to know how to sort the elements it needs to sort. "myComp" is used for that purpose. You can see that "myComp" is passed to "qsort" as the last argument.

• Reply • Share ›

**helper** • 7 months ago

my java code with some slight changes: i have first stored the graph in the traditional adjacency list representation and then converted it into edges[] and provided that as an interface to the method Kruskal.
http://ideone.com/TezRoE

• Reply • Share ›

**helper** helper • 6 months ago

the earlier code was deleted by ideone....pls go to http://ideone.com/QEMfNH

• Reply • Share ›

**Devesh Agrawal** • 7 months ago

Why this Union function is complex.

Cant we use something like this:

```
void Union(subset subsets[], int x, int y)
{

int xroot = find(subsets, x);
int yroot = find(subsets, y);
subsets[xroot].parent = yroot;

}
```

Is there any case where this can fail???

• Reply • Share ›

**Anurag Singh** Devesh Agrawal • 7 months ago

No. It won't.
What you are doing is fine and this is already discussed at
Union-Find Algorithm | Set 1 (Detect Cycle in a Graph)
The method used here is
Union-Find Algorithm | Set 2 (Union By Rank and Path Compression)

• Reply • Share ›

**helper** Anurag Singh • 7 months ago

to bring down the time complexity of union to O(lg n) and hence Kruskal algo to O(mlg n)... @devesh for your solution, it is O(n) for find and hence O(mn) for Kruskal.

• Reply • Share ›

**ajayv** • 7 months ago

in java
http://ideone.com/KmIJmO

• Reply • Share ›

**ajayv** • 7 months ago

in java

```
import java.util.*;

class subset{
int par;
int rank;
}
class triple{
int src;
int dest;
int weight;

public triple(int x,int y,int z){
src = x;
dest = y;
weight = z;
}
}
```

see more

1 • Reply • Share ›

**Anurag Singh** ajayv • 7 months ago

It's good to post code on http://ideone.com and share the link here.
Code posted here are not readable and Indent style is not maintained.

• Reply • Share ›

**ajayv** • 7 months ago

while loop condition should be while(e < V)

• Reply • Share ›

**Anurag Singh** ajayv • 7 months ago

e is no of edges in MST and it will be "V-1". e starts from 0 (ZERO) and going to "V-2" for "V-1" edges. SO e is varying from 0 to V-2 i.e. ( e < V-1).

• Reply • Share ›

**ajayv** Anurag Singh • 7 months ago

yes, i got that after posting this :) thanks

• Reply • Share ›

**Arjun K** • 8 months ago

you can check this > generate minimum spanning tree using Kruskal method in c
http://www.msccomputerscience....

• Reply • Share ›

**Sreenivas Doosa** • 8 months ago

In return statement of myComp funtion '>' should be replaced with '-'.

i.e.

return a1->weight - b1->weight;

• Reply • Share ›

**Anurag Singh** • 8 months ago

Java: http://ideone.com/WTlrCH

• Reply • Share ›

**Guest** • 8 months ago

I'm not using parent function in finding out the rank..

• Reply • Share ›

**Guest** • 8 months ago

I think this code is better than above one ..

can anyone point out the error..

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct graph

{

int E,V;

struct edge *E1;

};

struct edge

see more

• Reply • Share ›

**SIDDHARTHA SARKAR** • a year ago

The value of E can be at most V*(V-1), not V*V in an undirected graph.

• Reply • Share ›

**Guest** • a year ago

The value of V can be at most V(V-1) for undirected graphs.
Not V*V.

• Reply • Share ›

**Alex T** • a year ago

myComp function should return three different values - 0, 1, -1, so the std::qsort function determines the place of each element. Therefore, the corrected myComp function is presented here:

```
int myComp(const void* a, const void* b)
{
Edge* a1 = (Edge*)a;
Edge* b1 = (Edge*)b;
if (a1->weight > b1->weight) return 1;
if (a1->weight == b1->weight) return 0;
if (a1->weight < b1->weight) return -1;
}
```

Thanks for attention.

2 • Reply • Share ›

**kaushik Lele** • a year ago

My Java implementation of Kruskal algo is as follows.
1) I am using map to keep track of already covered nodes.
2) Edge class implements Comparable so that edges can be sorted.
3) Program asks for weight for distance between every pair of node(as non-directional; so every pair is asked just once)
4) User input zero indicates that no such link exist
http://ideone.com/IdwASw

```
------- input output console -------------------

Enter the number of vertices
4
Enter distance between node 1 and 2
6
Enter distance between node 1 and 3
8
Enter distance between node 1 and 4
6
```

see more

^ | ˅ • Reply • Share ›

**Anurag Singh** ➦ kaushik Lele • 8 months ago

It looks like you are not using disjoint-set data structure and so this will not probably work, based on the discussion at:
http://www.geeksforgeeks.org/g...

^ | ˅ • Reply • Share ›

**kaushik Lele** • a year ago

/* How to add code here directly without using ideone.com

*/

^ | ˅ • Reply • Share ›

**Rahul** • a year ago

I have a doubt, how will while loop will break, if there is a cycle in the graph ? as we are taking break condition as e < V-1 and e only increments when there is no cycle till now.

^ | ˅ • Reply • Share ›

**Anurag Singh** ➦ Rahul • 8 months ago

Here graph is "connected and undirected", so for a graph with V nodes, there will be atleast V-1 edge (A connected V node graph will be acyclic with V-1 edges). When edge count is more, there will be cycle. So it is guaranteed that e will become V-1 at some point and it will come out of the loop.
It will make an infinite loop only when graph is not connected, but this condition is ruled out, as graph must be connected.

In case, graph is not connected, then we can modify this program to get MSF (Minimum Spanning Forest), where you need to find connected components 1st, then run MST program on each connected component.

^ | ˅ • Reply • Share ›

**Anitesh Kumar** • a year ago

Please correct me if i am wrong. The compare function used for ::qsort(), mycomp() should be static.
I am coding with VC++ and using VS2008 IDE. If i do not make mycomp() function static i am getting following CT error:
"function call missing argument list; use '&Graph::mycomp' to create a pointer to member".
Please reply.

^ | ˅ • Reply • Share ›

**Vikas Malviya** ➦ Anitesh Kumar • 9 months ago

at least in g++, this is not a requirement

• Reply • Share ›

**Guest** • a year ago

package kruskalunionfind;

class Edge
{
int src,dest,weight;
}

class Graph
{
// V-> Number of vertices, E-> Number of edges
int V, E;

// graph is represented as an array of edges. Since the graph is
// undirected, the edge from src to dest is also edge from dest
// to src. Both are counted as 1 edge here.
Edge edge[]=null;

public void createGraph(int v,int e)

see more

• Reply • Share ›

**angel** • 2 years ago

Hi,

I have one doubt,why do we need to use Union find algorithm for loop detection,if both vertics of current edge are already visited ,then definately they would create loop.so why to spend extra log n processing time in finding of loop.I made a small program of kruskals,in which i used one visited array of vertices only to detect cycle.Please let me know if i am missing something

void MinSpanningTree_Kruskals(struct edge *Graph_edge,int count)

{

bool visited[SIZE]={0,};

int i=0,k=0;

struct edge final_edges[SIZE]={0,};

while(k<size-1&& i<count)="" {="" if(!visited[graph_edge[i].dest]||!visited[graph_edge[i].src])="" {="" visited[graph_edge[i].dest]="1;" visited[graph_edge[i].src]="1;" final_edges[k++]="Graph_edge[i];" }="" i++;="" }="" printf("\nkruskals="" algorithm\n");="

for(i="0;i&lt;SIZE-1;i++)" {=""
printf("\nsrc="%d,dest=%d,weight=%d\n",final_edges[i].src,final_edges[i].dest,final_edges[i].wei
}="" }="">

3 • Reply • Share ›

**sudshekhar** angel • 2 years ago

Hi,

Consider the following :

0 1 3
1 2 5
2 3 4
3 4 6
4 5 7
5 6 8
6 0 9

The first two are the src and dest and third is the weight.
Here, we first add the edge (0,1) and then (2,3). Then we get the edge (1,2) which has had both its vertices's visited, BUT it doesn't form a loop. (since the two components are independent till now) and it will be a part of the MST.

This is why we need the union-find algorithm.

4 • Reply • Share ›

**Kuntal** • 2 years ago

```
 #include<stdio.h>
#include<malloc.h>
#define EDGESIZE 10
struct edge {
int cost;
int ver1;
int ver2;
};
struct edge *edges;
int *vertex;
void partition(int low,int high){
int mid;
if(low<high){
mid=(low+high)/2;
partition(low,mid);
partition(mid+1,high);
mergeSort(low,mid,high);
```

see more

• Reply • Share ›

**JV** • 2 years ago

does rank variable has any importance, in the subset struct

• Reply • Share ›

**GuruSimhe** JV • 2 years ago

Dude, go through Union by rank algorithm first. :-)

1 • Reply • Share ›

**piki** • 2 years ago

what is wrong with my code plz tell me when i am trying to submit it on SPOJ it is giving 0 point so please tell me what is wrong with this

Thanks in advance

```
 /* #include<stdio.h>
#include<iostream>
#include<queue>
#include<vector>
#include<algorithm>
#include<string.h>

using namespace std;

struct Edge
{
  int start;
  int end;
```

see more

• Reply • Share ›

**piki** • 2 years ago

what is wrong with my code plz tell me when i am trying to submit it on SPOJ it is giving 0 point so please tell me what is wrong with this

Thanks in advance

• Reply • Share ›

**piki** • 2 years ago

what is wrong with my code is it correct or not when i am trying to submit it on spoj it is giving 0 point so plz tell me my mistake

Thanks in advance

• Reply • Share ›

**piki** • 2 years ago

```
 /* #include<stdio.h>
#include<iostream>
#include<queue>
#include<vector>
#include<algorithm>
#include<string.h>

using namespace std;

struct Edge
{
  int start;
  int end;
  int weight;
};

struct compare
{
```

see more

• Reply • Share ›

**piki** ➦ piki • 2 years ago

what is wrong with my code plz tell me when i am trying to submit it on SPOJ it is giving 0 point so please tell me what is wrong with this

Thanks in advance

• Reply • Share ›

**ATul** • 2 years ago

In the condition

```
while ( e < V - 1 )
```

it should be

```
while ( i < V - 1 )
```

because i is the index of gettig edges.

• Reply • Share ›

**ATul** ATul • 2 years ago

sorry i misguied it E with V

• Reply • Share ›

**John** • 2 years ago

Why doesn't this work with

Number of E=3
Number of V=3

SRC DEST WEIGHT
7 8 3
7 9 2
8 9 1

4 • Reply • Share ›

Load more comments

- 
- 
- 
- 
    - Interview Experiences
    - Advanced Data Structures
    - Dynamic Programming
    - Greedy Algorithms
    - Backtracking
    - Pattern Searching
    - Divide & Conquer
    - Mathematical Algorithms
    - Recursion
    - Geometric Algorithms
-

## Popular Posts

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Follow @GeeksforGeeks

## Recent Comments

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago
- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 2 hours ago
- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 2 hours ago