LAB SESSION 06:

Date of the Session:__/__/____ **Time of the Session:____to_____**

Pre-Lab:

1. Our TASC has the information that a scientist has gone crazy and was planning a gas leakage in Delhi. We do not know the exact location where he was planning to do this attack. When Srikanth Tiwari tried to catch him, The Scientist tried to kill himself and has gone into coma in this process. Now no one knows the location of the attack except he has kept all the information in a file in his private server, but it is encoded using Huffman coding. So, help NIA decode the information. Given the root of the graph and the encoded information write a function to decode it. The function will have input parameters of root node and encoded string.

   Input
   Encoded String-1001011
   Output
   ABACA

```
Class Node:
def __init__(self, freq, symbol, left=none, right=none):
    self.freq = freq
    self.symbol = symbol
    self.left = left
    self.right = right
    self.huff = ''

def printNode(node, val = ''):
    newval = val + str(node.huff)
    if(node.left
        print Nodes(Node left, nuu val)
    if (node.right)
        print Nodes(node right, new val)
    if (not node.left and not node.right):
        printf(Node.symbol → {newval}")
```

In-Lab:

1) You are a Human resource manager working in a Startup. You are tasked with to utilize the best of the working professionals to get the maximum profit for different jobs of a Project.
An array of jobs is given where every job has a deadline and associated profit if the job is finished before the deadline. It is also given that every job takes single unit of time, so the minimum possible deadline for any job is 1.How to maximize total profit if only one job can be scheduled at a time.
Write a code for the following problem.

Input
7
a b c d e f g
3 4 4 2 3 1 2
35 30 25 20 15 12 5
Output
110
d c a b

```
def printJobsequencing (arr, t)
    n = len(arr)
    for i in range(n):
        for j in range(n-1-i)
        if (arr[j][1] < arr[j+1][1]):
            arr[j], arr[j+1] = a[j+1], a[j]
result [False] * t
job = ['i'] * t
for j in range(min(i-1, arr[i][1]-1, -1, -1)
    if (result[j] is false):
        result[i] = True
        Job[i] = arr[i][0]
        break:
print(Job)
arr = [['a', 3, 35], ['b', 2, 4, 30], ['c', 3, 4, 25], ['d', 4, 2, 20]
    ['e', 3, 15], ['f', 1, 12], ['g', 2, 5]
```

print JobSequencing (arr, 3)

i/p

7

a b c d e f g

3 4 4 2 3 1 2

13 30 25 20 15 12 5

o/p

1 1 0

d c a b

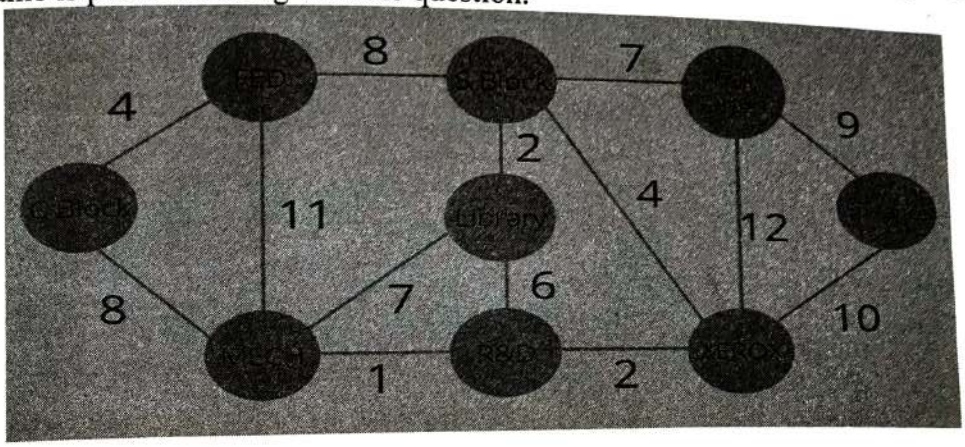
```
def printsolution(self, dist):
    print("vertex distance from source")
    for node in range(self.V):
        print(node, "-", dist[node])

def minDistance(self, dist, sptset):
    min = sys.maxsize
    for v in range(self.V):
        if dist[v] < min and sptset[v] == False:
            min = dist[v]
            min_index = v
    return min_index

def dijkstra(self, src):
    distance = [sys.maxsize] * self.V
    dist[src] = 0
    sptset = [False] * self.V
```

2) Surya is a student at KL University, He is currently standing in the C-Block. He has 8 friends who are situated at different Blocks (Places) inside the university and he wishes to talk to each of them in person. The distances are represented on the undirected graph which is provided below. He wishes to take the shortest distance for each place from his location. Help him in meeting every one of his friends by developing a program that can determine the shortest distance between the C-Block and every other place on the graph. Remember that the path is not required.

Hint: Use Dijkstra's algorithm to calculate the distances between the C-Block and every other place Output for the same is provided along with the question.



Output:

Vertex	Distance from Source
C Block	0
FED Block	4
S Block	12
Main Canteen	19
Entrance	21
Xerox	11
R&D Block	9
Mech Block	8
Library	14

```
import sys
class graph():
def __init__(self, vertices):
    self.V = vertices
self.graph = [[0 for column in range(vertices)]
for you in range(vertices)]
```

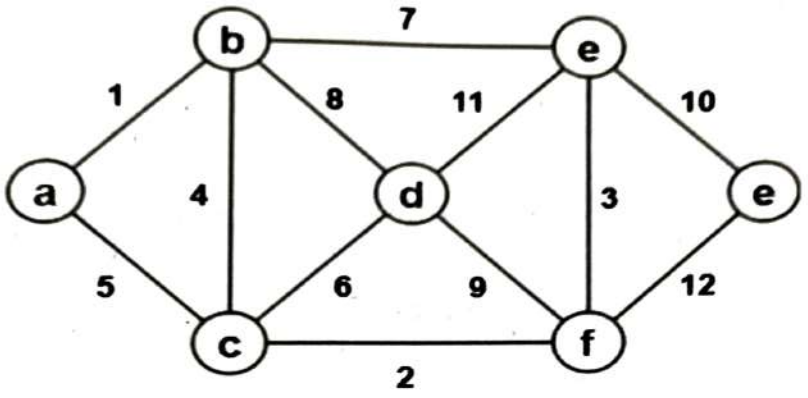
```
#include <stdio.h>
#include <limits.h>
#define vertices

int minimum_key(int k[], int mst[])
{
    int minimum= INT_MAX, min;
    for (i=0; i<vertices; i++){
        if(mst[i]==0 && k[i] < minimum)
            minimum = k[i], min=i
        return min
    }
void point (int g[vertices][vertices])
{
    int parent[vertices];
    int k[vertices];
    int msk[vertices];
    int i, count, u, v;
    for (i=0; i< vertices; i++)
        k[i]= INT_MAX, mst[i]=0
    k[0]=0;
    parent[0]=-1

for(count=0; count<vertices-1; count++)
{
        v= minimum.key[k,mst);
            mst[v]=1;
```

Post-Lab:

1. Mr. Tripathiis a network cable operator. He now shifted to a new city where he needs to work on designing the cable lines to each street/colony. Given the graph and each node represents a street, help him to design an optimal cable line using prim's algorithm. Write a program to solve this.



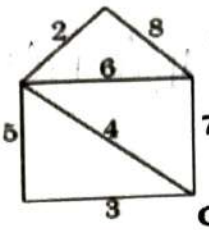

```
for (v=0 & vertices; v++)
  if (g[u][v] && mst[v]==0 && g[u][v] < k[u])
      parent[v] = u
      k[v] = g[u][v];
}
for (i=1; i<vertices; i++)
    print ("%d %d %d\n", parent[i], g[i], parent[i]);
}
void main ( ) {
    int g[vertices][vertices] = {{3, 2, 4, 9, 0},
    {5, 4, 2, 1, 0, 4}, {0, 4, 1, 0 24}
    {8, 10, 6, 2, 10}, {1, 0, 8, 11, 10}
    print (g);
}
```

```
def union(self, parent, rank, x, y):
    x.root = self.find(parent, x)
    y.root = self.find(parent, y)
    if rank[x.root] < rank[y.root]:
        parent[x.root] = y[root]
    elif rank[x.root] > rank[y.root]:
        parent[y.root] = x.root
    else:
        parent[y.root] = x.root
        rank[x.root] += 1
```

```
def kruskalmst(self):
    result = []
    i = 0
    e = 0
    self.graph = sorted(self.graph,
    key = lambda item: item[2])
    parent = []
    rank = []
    for node in range(self.V):
        parent.append(node)
        rank.append(0)
    while e < self.V - 1:
        u, v, w = self.graph[i]
        i = i + 1
        x = self.find(parent, u)
        y = self.find(parent,
        if x != y:
            e = e + 1
            result.append([u, v,
            self.union(parent, ran
```

```
g = Graph(8)
g.addEdge(0, 1, 2)
g.addEdge(0, 2, 6)
g.addEdge(0, 3, 4)
g.addEdge(0, 4, 5)
g.addEdge(0, 2, 8)
g.addEdge(2, 3, 7)
```

2. Mr.Tripathi done with designing the cable lines but now there comes a new task, that is working on street lines. Help him again to find weight of the graph using Kruskal algorithm. Write a program to solve this.



Graph G

```
from collection import defaultdict

class graph:
    def __init__(self, vertices)
        self.V = vertices
        self.graph = []
    def add edge (self, u, v, w):
        self.graph.append([u, v, w])
    def find (self. parent, i)
        if (parent[i] == i)
            return i
        return self.find (parent, parent[i])
```

(For Evaluator's use only)

Comment of the Evaluator (if Any)	Evaluator's Observation
	Marks Secured:_____ out of_____ Full Name of the Evaluator: P. Anuja Signature of the Evaluator Date of Evaluation: