S.NO	EXPERIMENTS
1.	Design and implement C/C++ Program to find Minimum Cost Spanning Tree of a given connected undirected graph using Kruskal's algorithm.
2.	Design and implement C/C++ Program to find Minimum Cost Spanning Tree of a given connected undirected graph using Prim's algorithm.
3.	a. Design and implement C/C++ Program to solve All-Pairs Shortest Paths problem usingFloyd's algorithm. b. Design and implement C/C++ Program to find the transitive closure using Warshal' s algorithm.
4.	Design and implement C/C++ Program to find shortest paths from a given vertex in a weighted connected graph to other vertices using Dijkstra's algorithm.
5.	Design and implement C/C++ Program to obtain the Topological ordering of vertices in a given digraph
6.	Design and implement C/C++ Program to solve 0/1 Knapsack problem using Dynamic Programming method.
7.	Design and implement C/C++ Program to solve discrete Knapsack and continuous Knapsack problems using greedy approximation method.
8.	Design and implement C/C++ Program to find a subset of a given set S = {sl , s2,. .... ,sn} of n positive integers whose sum is equal to a given positive integer d.
9.	Design and implement C/C++ Program to sort a given set of n integer elements using Selection Sort method and compute its time complexity. Run the program for varied values of n> 5000 and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.
10	Design and implement C/C++ Program to sort a given set of n integer elements using Quick Sort method and compute its time complexity. Run the program for varied values of n> 5000 and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.
11.	Design and implement C/C++ Program to sort a given set of n integer elements using Merge Sort method and compute its time complexity. Run the program for varied values of n> 5000, and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.
12	Design and implement C/C++ Program for N Queen's problem using Backtracking.

1.Design and implement C/C++ Program to find Minimum Cost Spanning Tree of a given connected undirected graph using Kruskal's algorithm.

Program:

```
#include<stdio.h>
#define INF 999
#define MAX 100

int p[MAX],c[MAX][MAX],t[MAX][2];

int find(int v)
{
while(p[v])
v=p[v];
return v;
}

void union1(int i,int j)
{
p[j]=i;
}

void kruskal(int n)
{
int i,j,k,u,v,min,res1,res2,sum=0;
for(k=1;k<n;k++)
{
min=INF;
for(i=1;i<n-1;i++)
{
for(j=1;j<=n;j++)
{
if(i==j)continue;
if(c[i][j]<min)
{
u=find(i);
v=find(j); if(u!=v)
{
res1=i; res2=j;
min=c[i][j];
}
}
}
}
union1(res1,find(res2));
t[k][1]=res1;
t[k][2]=res2;
sum=sum+min;
}
printf("\nCost of spanning tree is=%d",sum);
printf("\nEdgesof spanning tree are:\n");

for(i=1;i<n;i++)
printf("%d -> %d\n",t[i][1],t[i][2]);
}
int main()
{
int i,j,n;
printf("\nEnter the n value:");
scanf("%d",&n); for(i=1;i<=n;i++)
p[i]=0;
printf("\nEnter the graph data:\n");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
scanf("%d",&c[i][j]);
kruskal(n);
return 0;
}
```

Input/Output:

```
Enter the n value:5
Enter the graph data:
0 10 15 9 999
10 0 999 17 15
15 999 0 20 999
9 17 20 0 18
999 15 999 18 0
Cost of spanning tree is=49

Edgesof spanning tree are:
1 -> 4
1 -> 2
1 -> 3
2 -> 5
```

2.Design and implement C/C++ Program to find Minimum Cost Spanning Tree of a given connected undirected graph using Prim's algorithm.

```
#include<stdio.h>
// #include<conio.h>
#define INF 999

int prim(int c[10][10],int n,int s)
{
int v[10],i,j,sum=0,ver[10],d[10],min,u;
for(i=1;i<=n;i++)
{
        ver[i]=s;
        d[i]=c[s][i];
        v[i]=0;

} v[s]=1;

ver[i]=s; d[i]=c[s][i]; v[i]=0;

for(i=1;i<=n-1;i++)
{
min=INF;
for(j=1;j<=n;j++)
if(v[j]==0 && d[j]<min)
{

min=d[j];
u=j;
}
v[u]=1;
sum=sum+d[u];
printf("\n%d -> %d sum=%d",ver[u],u,sum);

for(j=1;j<=n;j++)
if(v[j]==0 && c[u][j]<d[j])
{
d[j]=c[u][j];
ver[j]=u;
}
}
return sum;
}

void main()
{
int c[10][10],i,j,res,s,n;
clrscr();
printf("\nEnter n value:");
scanf("%d",&n);
printf("\nEnter the graph data:\n");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
scanf("%d",&c[i][j]);
printf("\nEnter the souce node:");
scanf("%d",&s);
res=prim(c,n,s);
printf("\nCost=%d",res);
getch();
}
```

Input/output:

```
Enter n value:3
Enter the graph data:
0 10 1
10 0 6
1 6 0
Enter the souce node:1
1 -> 3 sum=1
3 -> 2 sum=7
Cost=7
```

3. a. Design and implement C/C++ Program to solve All-Pairs Shortest Paths problem using Floyd's algorithm.
b. Design and implement C/C++ Program to find the transitive closure using Warshal' s algorithm.

3a

```
	#include<stdio.h>
	#include<conio.h>
	#define INF 999

	int min(int a,int b)
	{
	return(a<b)?a:b;
	}

	void floyd(int p[][10],int n)
	{
	int i,j,k;
	for(k=1;k<=n;k++)
	for(i=1;i<=n;i++)
	for(j=1;j<=n;j++)
	p[i][j]=min(p[i][j],p[i][k]+p[k][j]);
	}

	void main()
	{
	inta[10][10],n,i,j;
	clrscr();
	printf("\nEnter the n value:");
	scanf("%d",&n);
	printf("\nEnter the graph data:\n");
	for(i=1;i<=n;i++)
	for(j=1;j<=n;j++)
	scanf("%d",&a[i][j]);
	floyd(a,n);
	printf("\nShortest path matrix\n");
	 for(i=1;i<=n;i++)
	{

	for(j=1;j<=n;j++)
	printf("%d ",a[i][j]);
	printf("\n");
	}
	getch();
}
```

Input/Output:

```
Enter the n value:4
Enter the graph
data:
0 999 3 999
2 0 999 999
999 7 0 1
6 999 999 0
Shortest path
matrix0 10 3 4
2 0 5 6
7 7 0 1
6 16 9 0
```

3b.

```
#include<stdio.h>
void warsh(int p[][10],int n)
{
        int i,j,k;
        for(k=1;k<=n;k+
        +)
        for(i=1;i<=n;i++
        )
        for(j=1;j<=n;j++
        )
        p[i][j]=p[i][j] || p[i][k] && p[k][j];
}

int main()
{
        int a[10][10],n,i,j;

        printf("\nEnter the n value:");
        scanf("%d",&n);
        printf("\nEnter the graph data:\n");
        for(i=1;i<=n;i++)
        for(j=1;j<=n;j++)
        scanf("%d",&a[i][j]);
        warsh(a,n);
        printf("\nResultant path matrix\n");
        for(i=1;i<=n;i++)
        {
                for(j=1;j<=n;j++)
                printf("%d
                ",a[i][j]);
                printf("\n");
        }
        return 0;
        }
```

```
Input/Output:
Enter the n value:4
Enter the graph
data:
0 1 0 0
0 0 0 1
0 0 0 0
1 0 1 0
Resultant path
matrix1 1 1 1
1 1 1 1
0 0 0 0
1 1 1 1
```