**Data Structures**

Printing Adjacency matrix of Directed And

Undirected graphs

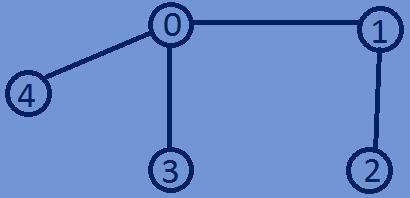
**Function for printing Adjacency matrix of Directed and Undirected graphs:**

**void** printAdjacencyMatrix(graph \*array,**int** n){  
 **for** (**int** i = 0; i < n; i++) {  
 **for** (**int** j = 0; j < n; j++) {  
 printf("%d ", \*(array + i\*n + j));  
 }  
 printf("\n");  
 }  
}

**Program for printing Adjacency matrix of Directed and Undirected graphs:**

#include<stdio.h>  
#include<stdlib.h>  
#include<time.h>  
**typedef int** graph;  
  
//constructing a undirectedgraph  
graph \*buildUndirectedGraph (**int** n) {  
 **int** i,j;  
 graph \*array = (graph \*) malloc(n \* n \* **sizeof**(graph));  
 srand((**unsigned**)time(NULL));  
 **for** (i = 0; i < n; i++) {  
 **for** (j = 0; j < n; j++) {  
 **if** (i == j) {  
 \*(array + i \* n + j) = 0;  
 } **else if** (i != j) {  
 **int** temp=rand()%2;  
 \*(array + i \* n + j) =temp;  
 \*(array + j \* n + i) =temp;  
 }  
 }  
 }  
 **return** array;  
}  
//constructing a directed graph  
graph \*buildDirectedGraph(**int** n){  
 **int** i,j;  
 graph \*array = (graph \*) malloc(n \* n \* **sizeof**(graph));  
 srand((**unsigned**)time(NULL));  
 **for** (i = 0; i < n; i++) {  
 **for** (j = 0; j < n; j++) {  
 **if** (i == j) {  
 \*(array + i \* n + j) = 0;  
 } **else if** (i != j) {  
 **int** temp=rand()%2;  
 \*(array + i \* n + j) =temp;  
 }  
 }  
 }  
 **return** array;  
}  
**void** printAdjacencyMatrix(graph \*array,**int** n){  
 **for** (**int** i = 0; i < n; i++) {  
 **for** (**int** j = 0; j < n; j++) {  
 printf("%d ", \*(array + i\*n + j));  
 }  
 printf("\n");  
 }  
}  
**int** main(){  
 **int** n=5;  
 graph \*array;  
 array=buildUndirectedGraph(n);  
 printAdjacencyMatrix(array,n);  
 array=buildDirectedGraph(n);printf("\n");  
 printAdjacencyMatrix(array,n);  
 **return** 0;  
}

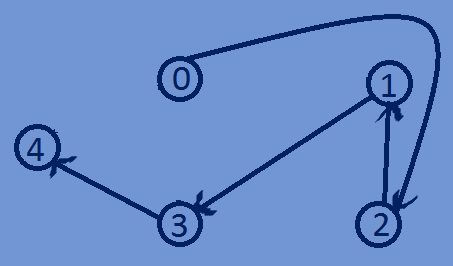
**Output:**

0 1 0 1 1

1 0 1 0 0

1. 1 0 0 0
2. 0 0 0 0

1 0 0 0 0

0 0 1 0 0

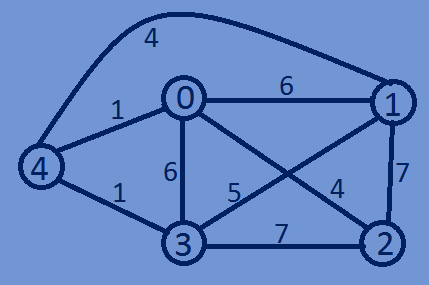
0 0 0 1 0

1. 1 0 0 0
2. 0 0 0 1

0 0 0 0 0

**Implementing of weighted Graphs(Directed and Undirected )using Adjacency Matrix:**

#include<stdio.h>  
#include<stdlib.h>  
#include<time.h>  
**typedef int** graph;  
  
//constructing a weighted undirectedgraph  
graph \*buildWUndirectedGraph (**int** n) {  
 **int** i,j;  
 graph \*array = (graph \*) malloc(n \* n \* **sizeof**(graph));  
 srand((**unsigned**)time(NULL));  
 **for** (i = 0; i < n; i++) {  
 **for** (j = 0; j < n; j++) {  
 **if** (i == j) {  
 \*(array + i \* n + j) = 0;  
 } **else if** (i != j) {  
 **int** temp=rand()%8;  
 \*(array + i \* n + j) =temp;  
 \*(array + j \* n + i) =temp;  
 }  
 }  
 }  
 **return** array;  
}  
//constructing a weighted directed graph  
graph \*buildWDirectedGraph(**int** n){  
 **int** i,j;  
 graph \*array = (graph \*) malloc(n \* n \* **sizeof**(graph));  
 srand((**unsigned**)time(NULL));  
 **for** (i = 0; i < n; i++) {  
 **for** (j = 0; j < n; j++) {  
 **if** (i == j) {  
 \*(array + i \* n + j) = 0;  
 } **else if** (i != j) {  
 **int** temp=rand()%8;  
 \*(array + i \* n + j) =temp;  
 }  
 }  
 }  
 **return** array;  
}  
**void** printAdjacencyMatrix(graph \*array,**int** n){  
 **for** (**int** i = 0; i < n; i++) {  
 **for** (**int** j = 0; j < n; j++) {  
 printf("%d ", \*(array + i\*n + j));  
 }  
 printf("\n");  
 }  
}  
**int** main(){  
 **int** n=5;  
 graph \*array;  
 array=buildWUndirectedGraph(n);  
 printAdjacencyMatrix(array,n);  
 array=buildWDirectedGraph(n);printf("\n");  
 printAdjacencyMatrix(array,n);  
 **return** 0;  
}

**Output:**

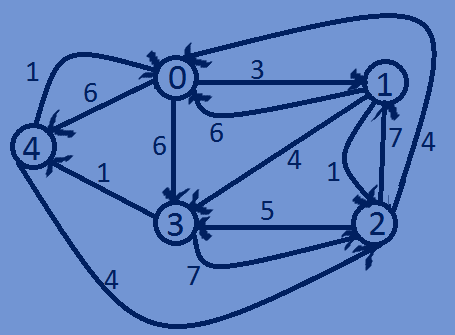
0 6 4 6 1

6 0 7 5 4

4 7 0 7 0

6 5 7 0 1

1. 4 0 1 0



1. 3 0 6 6

6 0 1 4 0

4 7 0 5 0

1. 0 7 0 1

1 0 4 0 0