

```

968 //Floyd
969 #include<stdio.h>
970 #include<stdlib.h>
971 #include<limits.h>
972 #define MAX 10000
973 int n;
974 void display(int d[][n]){
975     for(int i = 0; i < n; i++){
976         for(int j = 0; j < n; j++){
977             if(d[i][j]==MAX)
978                 printf("%6s", "INF");
979             else
980                 printf ("%6d", d[i][j]);
981         }
982         printf("\n");
983     }
984 }
985 void floyd(int graph[][n])
986 {
987     int d[n][n], i, j, k;
988     for(i = 0; i < n; i++){
989         for(j = 0; j < n; j++){
990             if(graph[i][j]==-1)
991                 d[i][j]=MAX;
992             else
993                 d[i][j]=graph[i][j];
994         }
995     }
996     printf("Matrix: D(0)\n");
997     display(d);
998     for(k=0;k<n;k++){
999         for(i=0;i<n;i++){
1000             for(j=0;j<n;j++){

```

```

1001         if(d[i][j] > d[i][k]+d[k][j])
1002             d[i][j]=d[i][k] + d[k][j];
1003     }
1004 }
1005 printf("\nD(%d) Solution Matrix : \n",k+1);
1006 display(d);
1007 }
1008 }
1009 int main()
1010 {
1011     printf("\n Enter the Number of vertices :- ");
1012     scanf("%d",&n);
1013     int D[n][n];
1014     printf("\nEnter the adjacency matrix (enter -1 if no direct path) :- \n");
1015     for(int i=0;i<n;i++){
1016         for(int j=0;j<n;j++){
1017             scanf("%d",&D[i][j]);
1018         }
1019     }
1020     floyd(D);
1021     return 0;
1022 }

```

Enter the Number of vertices :- 4

Enter the adjacency matrix (enter -1 if no direct path) :-

0 -1 3 -1

2 0 -1 -1

-1 7 0 1

6 -1 -1 0

Matrix: D(0)

0	INF	3	INF
2	0	INF	INF
INF	7	0	1
6	INF	INF	0

D(1) Solution Matrix :

0	INF	3	INF
2	0	5	INF
INF	7	0	1
6	INF	9	0

D(2) Solution Matrix :

0	INF	3	INF
2	0	5	INF
9	7	0	1
6	INF	9	0

D(3) Solution Matrix :

0	10	3	4
2	0	5	6
9	7	0	1
6	16	9	0

D(4) Solution Matrix :

0	10	3	4
2	0	5	6
7	7	0	1
6	16	9	0

Process returned 0 (0x0) execution time : 81.630 s

Press any key to continue.

```

1030 //Warshall
1031 #include<stdio.h>
1032 #include<stdlib.h>
1033 const int MAX = 100;
1034 void Warshall(int graph[MAX][MAX],int n)
1035 {
1036     int i,j,k;
1037     for (k=0; k<n; k++){
1038         for (i=0; i<n; i++){
1039             for (j=0; j<n; j++){
1040                 if (graph[i][j] || (graph[i][k] && graph[k][j]))
1041                     graph[i][j] = 1;
1042             }
1043         }
1044     }
1045 }
1046 int main()
1047 {
1048     int i,j,n;
1049     int graph[MAX][MAX];
1050     printf("Enter the number of vertices : ");
1051     scanf("%d",&n);
1052     printf("Enter the adjacency matrix :-\n");
1053     for (i=0; i<n; i++)
1054         for (j=0; j<n; j++)
1055             scanf("%d",&graph[i][j]);
1056     Warshall(graph,n);
1057     printf("\nThe transitive closure for the given graph is :-\n");
1058     for (i=0; i<n; i++){
1059         for (j=0; j<n; j++)
1060             printf("%d\t",graph[i][j]);
1061         printf("\n");
1062     }
1063     return 0;
1064 }

```


Enter the number of vertices : 4

Enter the adjacency matrix :-

0 0 1 0

0 0 0 1

1 0 0 0

0 1 0 0

The transitive closure for the given graph is :-

1 0 1 0

0 1 0 1

1 0 1 0

0 1 0 1

Process returned 0 (0x0) execution time : 33.908 s

Press any key to continue.

—

```

1068 //Knapsack
1069 #include <stdio.h>
1070 #include<conio.h>
1071 int max(int a,int b)
1072 {
1073     return (a > b) ? a : b;
1074 }
1075 int knapSack(int W,int wt[],int val[],int n)
1076 {
1077     int i, w;
1078     int K[n+1][W+1];
1079     for (i=0;i<=n;i++){
1080         for (w=0;w<=W;w++){
1081             if(i==0 || w==0)
1082                 K[i][w]=0;
1083             else if(wt[i-1]<=w)
1084                 K[i][w]=max(K[i-1][w],val[i-1]+K[i-1][w-wt[i-1]]);
1085             else
1086                 K[i][w] = K[i-1][w];
1087         }
1088     }
1089     return K[n][W];
1090 }
1091 int main()
1092 {
1093     int n,max;

```

```
1091 int main()
1092 {
1093     int n,max;
1094     printf("\nEnter the No of Items: ");
1095     scanf("%d",&n);
1096     int val[n],wt[n];
1097     printf("\nEnter the Weight and Profit of Items :- \n");
1098     for(int i=0;i<n;i++){
1099         printf("Weight of Item %d :- ",(i+1));
1100         scanf("%d",&wt[i]);
1101         printf("Value of Item %d :- ",(i+1));
1102         scanf("%d",&val[i]);
1103         printf("\n");
1104     }
1105     printf("\nEnter the Capacity :- ");
1106     scanf("%d",&max);
1107     printf("Maximum Profit :- %d",knapSack(max,wt,val,n));
1108     return 0;
1109 }
```

Enter the No of Items: 4

Enter the Weight and Profit of Items :-

Weight of Item 1 :- 5

Value of Item 1 :- 25

Weight of Item 2 :- 1

Value of Item 2 :- 32

Weight of Item 3 :- 9

Value of Item 3 :- 20

Weight of Item 4 :- 4

Value of Item 4 :- 30

Enter the Capacity :- 12

Maximum Profit :- 87

Process returned 0 (0x0) execution time : 37.282 s

Press any key to continue.