```c
//Floyd
#include<stdio.h>
#include<stdlib.h>
#include<limits.h>
#define MAX 10000
int n;
void display(int d[][n]){
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            if(d[i][j]==MAX)
                printf("%6s","INF");
            else
                printf ("%6d", d[i][j]);
        }
        printf("\n");
    }
}
void floyd(int graph[][n])
{
    int d[n][n], i, j, k;
    for(i = 0; i < n; i++){
        for(j = 0; j < n; j++){
            if(graph[i][j]==-1)
                d[i][j]=MAX;
            else
                d[i][j]=graph[i][j];
        }
    }
    printf("Matrix: D(0)\n");
    display(d);
    for(k=0;k<n;k++){
        for(i=0;i<n;i++){
            for(j=0;j<n;j++){
```

```
                if(d[i][j] > d[i][k]+d[k][j])
                    d[i][j]=d[i][k] + d[k][j];
            }
        }
        printf("\nD(%d) Solution Matrix : \n",k+1);
        display(d);
    }
}
int main()
{
    printf("\n Enter the Number of vertices :- ");
    scanf("%d",&n);
    int D[n][n];
    printf("\nEnter the adjacency matrix (enter -1 if no direct path) :- \n");
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            scanf("%d",&D[i][j]);
        }
    }
    floyd(D);
    return 0;
}
```

```
 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.
```

```c
//Warshall
#include<stdio.h>
#include<stdlib.h>
const int MAX = 100;
void Warshall(int graph[MAX][MAX],int n)
{
    int i,j,k;
    for (k=0; k<n; k++){
        for (i=0; i<n; i++){
            for (j=0; j<n; j++){
                if (graph[i][j] || (graph[i][k] && graph[k][j]))
                    graph[i][j] = 1;
            }
        }
    }
}
int main()
{
    int i,j,n;
    int graph[MAX][MAX];
    printf("Enter the number of vertices : ");
    scanf("%d",&n);
    printf("Enter the adjacency matrix :-\n");
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            scanf("%d",&graph[i][j]);
    Warshall(graph,n);
    printf("\nThe transitive closure for the given graph is :-\n");
    for (i=0; i<n; i++){
        for (j=0; j<n; j++)
            printf("%d\t",graph[i][j]);
        printf("\n");
    }
    return 0;
}
```

```
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.
```

```c
//Knapsack
#include <stdio.h>
#include<conio.h>
int max(int a,int b)
{
    return (a > b) ? a : b;
}
int knapSack(int W,int wt[],int val[],int n)
{
    int i, w;
    int K[n+1][W+1];
    for (i=0;i<=n;i++){
        for (w=0;w<=W;w++){
            if(i==0 || w==0)
                K[i][w]=0;
            else if(wt[i-1]<=w)
                K[i][w]=max(K[i-1][w],val[i-1]+K[i-1][w-wt[i-1]]);
            else
                K[i][w] = K[i-1][w];
        }
    }
    return K[n][W];
}
int main()
{
    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  }
1110
1111
1112
1113
```

```
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.
```