

Traveling Salesman Problem (TSP – using Dynamic Programming)

```
#include <iostream>

#include <climits>

using namespace std;

int n;

int dist[10][10];

int dp[1 << 10][10];

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

int tsp(int mask, int pos) {
    if(mask == (1 << n) - 1)
        return dist[pos][0];

    if(dp[mask][pos] != -1)
        return dp[mask][pos];

    int ans = INT_MAX;
    for(int city = 0; city < n; city++) {
        if((mask & (1 << city)) == 0) {
            int newAns = dist[pos][city] +
                tsp(mask | (1 << city), city);
            ans = min(ans, newAns);
        }
    }
}
```

```

    }

    return dp[mask][pos] = ans;
}

int main() {
    cout << "Enter number of cities: ";
    cin >> n;

    cout << "Enter distance matrix:\n";
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            cin >> dist[i][j];

    for(int i = 0; i < (1 << n); i++)
        for(int j = 0; j < n; j++)
            dp[i][j] = -1;

    cout << "Minimum travel cost: " << tsp(1, 0);
    return 0;
}

```

Output :

Output

```
* Enter number of cities: 3
Enter distance matrix:
9 8 7
5 4 6
3 7 9
Minimum travel cost: 17

=== Code Execution Successful ===
```