```
#include <iostream>
#include <iomanip>

using namespace std;

const int MAX = 10;
int path[MAX];
static int k=0;
int count = 0;
int perm[120][7];
int tourcost[120];

void swap (int *x, int *y)
{
    int temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

void DepthFirstSearch(int currentVertex, int v[MAX], int g[MAX]
[MAX], int n)
{
     int i;
     v[currentVertex]=1;
     path[k++]=currentVertex;
     for (i=0; i<n; i++)
        if (g[currentVertex][i] && !v[i])
            DepthFirstSearch(i,v,g,n);
}

void permute(int *a, int i, int n)
{
   int j,k;
   if (i == n)
   {
       for(k=0;k<=n;k++)
       {
       //start from 2nd column
       perm[count][k+1] = a[k];
       }
       count++;
   }
   else
   {
    for (j = i; j <= n; j++)
       {
      swap((a+i), (a+j));
      permute(a, i+1, n);
      swap((a+i), (a+j)); //backtrack
       }
   }
}
```

```
int AppTSP(int n, int cost[MAX][MAX])
{
    int i, j, u, v, min,Excost=0;
    int sum, k, t[MAX][2], p[MAX], d[MAX], s[MAX],tree[MAX][MAX];
    int source, count;
    int visited[MAX];
    for (i=0; i<n; i++)
        visited[i] = 0;

    min = 9999;
    source = 0;

    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
        {
            if(cost[i][j] != 0 && cost[i][j] <= min)
            {
                min = cost[i][j];
                source = i;
            }
        }
    }

    for(i=0; i<n; i++)
    {
        d[i] = cost[source][i];
        s[i] = 0;
        p[i] = source;
    }
    s[source] = 1;
    sum = 0;
    k = 0;
    count = 0;

    while (count != n-1)
    {
        min = 9999;
        u = -1;
        for(j=0; j<n; j++)
        {
            if(s[j] == 0)
            {
                if(d[j] <= min)
                {
                     min = d[j];
                     u = j;
                }
            }
        }

        t[k][0] = u;
        t[k][1] = p[u];
        k++;
        count++;
        sum += cost[u][p[u]];
        s[u] = 1;

        for(v=0; v<n; v++)
        {
            if(s[v]==0 && cost[u][v]<d[v])
            {
                d[v] = cost[u][v];
                p[v] = u;
            }
        }
    }

    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
        {
        tree[i][j]=0;
        }
    }

    if(sum >= 9999)
    cout << "\nSpanning tree does not exist";
    else
    {
    for(i=0; i<k; i++)
    {
        tree[t[i][0]][t[i][1]] = tree[t[i][1]][t[i][0]] =1;
    }
    }

    DepthFirstSearch(0,visited,tree,n);

    cout << "\n The Approximate Minimum Cost tour is" << endl;
    for(i=0;i<=k;i++)
    {
    cout << path[i] << "->";
    Excost += cost[path[i]][path[i+1]];
    }
    cout << path[0];
    Excost += cost[path[i]][path[0]];
    cout << "\n The Approximate Minimum Cost of the tour is" <<
Excost << endl;
    return Excost;
}

int main(void)
{
    int a[MAX][MAX] = { { 0,  4,  8,  9, 12},
                { 4,  0,  6,  8,  9},
                { 8,  6,  0, 10, 11},
                { 9,  8, 10,  0,  7},
                {12,  9, 11,  7,  0}};
    int NumOfCity = 5;
    int interCities = 4,i,j;
    int mct=999,mctIndex,Appmct;

    //Source and destination is 0 remaining are intermediary cities
    int city[4] = {1,2,3,4};
    permute(city, 0, interCities-1);
    for(i=0;i<24;i++)
    {
    for(j=0;j<5;j++)
    {
        tourcost[i]+= a[perm[i][j]][perm[i][j+1]];
    }
    if( mct > tourcost[i])
    {
        mct = tourcost[i];
        mctIndex = i;
    }
    }

    cout << "\n The Exact Minimum Cost tour is" << endl;
    for(i=0;i<NumOfCity;i++)
    cout << perm[mctIndex][i] << "->";
    cout << perm[mctIndex][i];
    cout << "\n The Exact Minimum Cost of the tour is" << mct <<
endl;

    Appmct = AppTSP(NumOfCity,a);
    cout << "\n The error in Approximation is " << Appmct - mct << "
units" << endl;
    cout << "\n The Accuracy ratio is " << (float)Appmct / mct <<
endl;
    cout << "\n The Approximate tour is "<<(((float)Appmct / mct) -
1)*100<< " percent longer than the optimal tour" << endl;

    return 0;
}
```

OUTPUT:

```
The Exact Minimum Cost tour is

0->1->2->4->3->0

The Exact Minimum Cost of the tour is37

The Approximate Minimum Cost tour is

0->1->2->3->4->0

The Approximate Minimum Cost of the tour is39

The error in Approximation is 2 units
The Accuracy ratio is 1.05405

The Approximate tour is 5.40541 percent longer than the optimal tour
```