**Code:**

#include <bits/stdc++.h>

using namespace std;

#define V 4

int cost = 0;

int start = 0, goal = 0;

vector<int> visited;

int graph[][V] = {

{0,10,15,20},

{10,0,35,25},

{15,35,0,30},

{20,25,30,0}

};

void calculate\_cost()

{

for(int i=0;i<visited.size();i++)

{

cost = cost + graph[visited[i]][visited[i+1]];

}

cout<<"cost = "<<cost;

}

void best\_first\_search()

{

int i,m;

vector< pair<int,int> > straight\_distance;

straight\_distance.push\_back(make\_pair(1,6));

straight\_distance.push\_back(make\_pair(2,4));

straight\_distance.push\_back(make\_pair(3,2));

cout<<"straight distance from starting node A "<<endl;

for(int i = 0; i < straight\_distance.size();i++)

{

printf("%c = %d\n",straight\_distance[i].first+65,straight\_distance[i].second);

}

int size=straight\_distance.size();

visited.push\_back(start);

printf("%c-->",start+65);

while(size--)

{

int saveIndex = 0;

for(int i = 1; i < straight\_distance.size();i++)

{

if(straight\_distance[saveIndex].second > straight\_distance[i].second)

{

saveIndex = i;

}

}

int smallestValue = straight\_distance[saveIndex].second;

for(int i = 0; i < straight\_distance.size();i++)

{

if(straight\_distance[i].second == smallestValue)

{

visited.push\_back(straight\_distance[i].first);

printf("%c-->",straight\_distance[i].first+65);

straight\_distance.erase(straight\_distance.begin() + saveIndex);

}

}

}

visited.push\_back(goal);

printf("%c\n",goal+65);

}

int main()

{

int n = 4,i,j;

cout<<"FOUR CITIES"<<endl<<"A,B,C,D"<<endl<<"cost"<<endl;

for(i=0;i<4;i++)

{

for(j=0;j<4;j++)

{

printf("%c=%c = %d\n",i+65,j+65,graph[i][j]);

}

printf("\n");

}

cout<<"PATH\n";

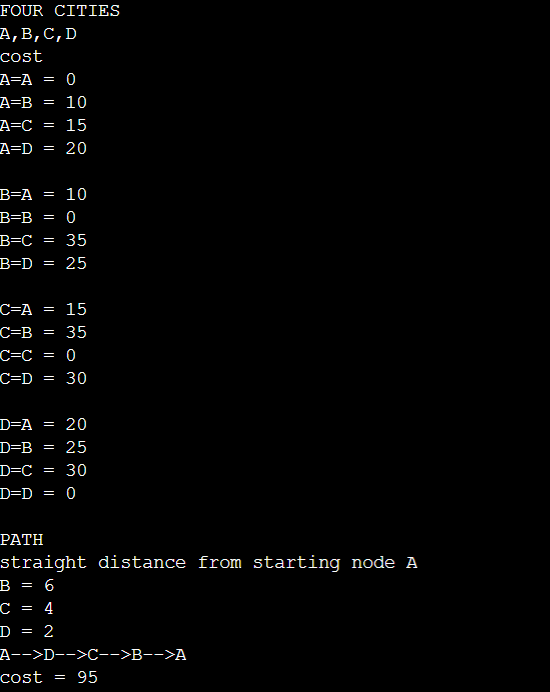
best\_first\_search();

calculate\_cost();

return 0;

}

**Output:**

****