```cpp
#include <bits/stdc++.h>
using namespace std;

struct Edge {
   int src, dest, weight;
};

class Bf{
    int V,E;
    vector<Edge> edges;
    public:
    void input(){
        cout<<"Enter number of vertices"<<endl;
        cin>>V;
        cout<<"Enter number of edges"<<endl;
        cin>>E;
        cout<<"Enter edges in the format src dest weight"<<endl;
        for(int i=0;i<E;i++){
            Edge edge;
            cout<<"Enter edge "<<i+1<<endl;
            cin>>edge.src>>edge.dest>>edge.weight;
            edges.push_back(edge);
        }
    }
    void bellman_ford(){
        cout<<"Enter source vertex"<<endl;
        int src;
        cin>>src;
        vector<int> output(V,INT_MAX);
        vector<int> parent(V);
        output[src]=0;
        for(int i=1;i<=V-1;i++){
            for(auto edge:edges){
                int u=edge.src;
                int v=edge.dest;
                int weight=edge.weight;
                if(output[u]!=INT_MAX && output[u]+weight<output[v]){
                    output[v]=output[u]+weight;
                    parent[v]=u;
                }
            }
        }
        for(auto edge:edges){
            int u=edge.src;
            int v=edge.dest;
            int weight=edge.weight;
            if(output[u]!=INT_MAX && output[u]+weight<output[v]){
                cout<<"Graph contains negative weight cycle"<<endl;
                return;
            }
        }
        cout<<"Vertex\tParent\tDistance from Source"<<endl;
        for(int i=0;i<V;i++){
            if(output[i]==INT_MAX){
                cout<<i<<"\t"<<parent[i]<<"\t"<<"INF"<<endl;
            }
            else{
                cout<<i<<"\t"<<parent[i]<<"\t"<<output[i]<<endl;
            }
        }

    }
};
```

```
int main(){
    Bf bf;
    bf.input();
    bf.bellman_ford();

    return 0;
}

/*
 Sample Input and Output:
   Enter number of vertices
5
Enter number of edges
10
Enter edges in the format src dest weight
Enter edge 1
0 1 6
Enter edge 2
0 3 7
Enter edge 3
1 2 5
Enter edge 4
1 4 -4
Enter edge 5
1 3 8
Enter edge 6
2 1 -2
Enter edge 7
4 2 7
Enter edge 8
4 0 2
Enter edge 9
3 2 -3
Enter edge 10
3 4 9
Enter source vertex
0
Vertex  Parent  Distance from Source
0       0       0
1       2       2
2       3       4
3       0       7
4       1       -2


Enter number of vertices
4
Enter number of edges
5
Enter edges in the format src dest weight
Enter edge 1
0 1 4
Enter edge 2
0 3 5
Enter edge 3
1 3 5
Enter edge 4
3 2 3
Enter edge 5
2 1 -10
Enter source vertex
0
Graph contains negative weight cycle
```

```
*/
```