```cpp
#[[#include]]# <iostream>
#[[#include]]# <algorithm>
#[[#include]]# <iomanip>
#[[#include]]# <map>
#[[#include]]# <set>
#[[#include]]# <queue>
#[[#include]]# <stack>
#[[#include]]# <numeric>
#[[#include]]# <bitset>
#[[#include]]# <cmath>

static const int MOD = 1000000007;
using ll = long long;
using u32 = uint32_t;
using namespace std;

template<class T> constexpr T INF = ::numeric_limits<T>::max()/32*15+208;

int main() {

    return 0;
}
```

```cpp
template <typename T>
struct edge {
    int from, to; T cost;
    edge(int to, T cost) : from(-1), to(to), cost(cost) {}
    edge(int from, int to, T cost) : from(from), to(to), cost(cost) {}
};

template <typename T>
vector<T> dijkstra(int s,vector<vector<edge<T>>> &G){
    size_t n=G.size();
    vector<T> d(n, INF<T>);
    priority_queue<pair<T, int>,vector<pair<T, int>>,greater<>> Q;
    d[s]=0;
    Q.emplace(0,s);
    while(!Q.empty()){
        T cost; int i;
        tie(cost, i) = Q.top(); Q.pop();
        if(d[i] < cost) continue;
        for (auto &&e : G[i]) {
            auto cost2 = cost + e.cost;
            if(d[e.to] <= cost2) continue;
            d[e.to] = cost2;
            Q.emplace(d[e.to], e.to);
        }
    }
    return d;
}
```

```cpp
template<typename T>
T extgcd(T a, T b, T &x ,T &y){
    for (T u = y = 1, v = x = 0; a; ) {
        ll q = b/a;
        swap(x -= q*u, u);
        swap(y -= q*v, v);
        swap(b -= q*a, a);
    }
    return b;
}

```