#include<bits/stdc++.h>

using namespace std;

vector<int> par,rank\_;

int find\_par(int x){

if(par[x]==x){

return x;

}

return par[x] = find\_par(par[x]);

}

bool sameset(int a,int b){

int x = find\_par(a);

int y = find\_par(b);

return x==y;

}

void union\_(int x,int y){

x = find\_par(x);

y = find\_par(y);

if(x!=y){

if(rank\_[x]>rank\_[y]){

par[y] = x;

}

else if(rank\_[x]<rank\_[y]){

par[x] = y;

}

else{

par[y] = x;

rank\_[x]++;

}

}

}

bool comp(vector<int> const &a, vector<int> const &b){

return a[2]<b[2];

}

int main(){

int n;

int e;

cin>>n>>e;

par.resize(n);

rank\_.resize(n);

for(int i=0;i<n;i++){

par[i] = i;

rank\_[i] = 0;

}

vector<vector<int> > graph(e,vector<int>(3));

for(int i=0;i<e;i++){

cin>>graph[i][0]>>graph[i][1]>>graph[i][2];

}

sort(graph.begin(),graph.end(),comp);

int cost = 0;

for(auto it: graph){

if(find\_par(it[0])!=find\_par(it[1])){

union\_(it[0],it[1]);

cost+= it[2];

}

}

int parentOfAll = find\_par(0);

for (int i = 1; i <n; i++)

{

if (parentOfAll != find\_par(i))

{

cost = -1;

}

}

cout<<cost;

}