#define \_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

#include<set>

#include<vector>

#include<map>

#include<cstdio>

#include<algorithm>

using namespace std;

vector<int> a;

map<int, char> m;

int main()

{

int n;

cin >> n;

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

{

}

}

int inc(const void\* a, const void\* b)

{

return \*(int\*)a - \*(int\*)b;

}

int inc(const void\* a, const void\* b)

{

if ((\*(node\*)a).one != (\*(node\*)b).one)

return (\*(node\*)a).one > (\*(node\*)b).one ? 1 : -1;

else return (\*(node\*)a).two - (\*(node\*)b).two;

}

qsort(a, n, 4, inc);//递增

typedef struct

{

int a;

}AA;

//并查集

int fa[100005], rank[100005];

void init(int n)

{

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

{

fa[i] = i;

rank[i] = 1;

}

}

int find(int x)

{

if (x == fa[x])

return x;

else {

fa[x] = find(fa[x]); //父节点设为根节点

return fa[x]; //返回父节点

}

}

void merge(int i, int j)

{

int x = find(i), y = find(j); //先找到两个根节点

if (rank[x] <= rank[y])

fa[x] = y;

else

fa[y] = x;

if (rank[x] == rank[y] && x != y)

rank[y]++; //如果深度相同且根节点不同，则新的根节点的深度+1

}

//树状数组

int lowbit(int i) {

return i & (-i);

}

void update(int i, int x) {

while (i <= maxn) {

b[i] = b[i] + x;

i = i + lowbit(i);

}

}

int question(int x)

{

int sum = 0;

while (x > 0) {

sum = sum + b[x];

x = x - lowbit(x);

}

return sum;

}

int Isprime(int n)

{

for (int i = 2;i < sqrt(n);i++)

{

if (n % i == 0)

return 0;

}

return 1;

}