//Somos insignificantes. Por mais que você programe sua vida, a qualquer momento tudo pode mudar.

// If you have God on your side,everything becomes clear

#include <bits/stdc++.h>

#include <ext/pb\_ds/assoc\_container.hpp>

#include <ext/pb\_ds/tree\_policy.hpp>

using namespace std;

using namespace \_\_gnu\_pbds;

typedef tree<int, null\_type, less\_equal<int>, rb\_tree\_tag, tree\_order\_statistics\_node\_update> ordered\_set;

const int MAX\_N = 1e5 + 5;

const int MAX\_L = 20; // ~ Log N

const long long MOD = 1e9 + 7;

const long long INF = 1e9 + 7;

typedef long long ll;

typedef vector<int> vi;

typedef pair<int,int> ii;

typedef vector<ii> vii;

typedef vector<vi> vvi;

#define fi first

#define popcount(x) \_\_builtin\_popcountll(x)

#define se second

#define LSOne(S) (S & (-S))

#define isBitSet(S, i) ((S >> i) & 1)

/\*void Sieve(int n)

{

//bool prime[n+1];

//memset(prime, true, sizeof(prime));

for (int p=2; p\*p<=n; p++)

{

if (prime[p] == true)

{

for (int i=p\*p; i<=n; i += p)

prime[i] = false;

}

}

} \*/

/\* struct cmp{

bool operator()(const pair<int,int>& v1,const pair<int,int>& v2) const{

}};\*/

long long binpow(long long a, long long b) {

a %=MOD;

long long res = 1;

while (b > 0) {

if (b & 1)

res = res \* a % MOD;

a = a \* a % MOD;

b >>= 1;

}

res=res%MOD;

return res;

}

void solve() {

ll n;

cin>>n;

ll a[n];

map<ll,ll>mp;

set<ll>s;

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

cin>>a[i];

mp[a[i]]++;

s.insert(a[i]);

}

sort(a,a+n);

bool f=1;

if(n==1){

cout<<"yes"<<endl;

return ;

}

if(a[n-1]>1 && a[0]<-1){

cout<<"no"<<endl;

return ;

}

if(a[n-1]>1 && mp[-1]>0){f=0;

}

if(a[0]<-1 && mp[-1]>0){

f=0;

}

if(mp[-1]>=2 && mp[1]==0){

f=0;

}

if(n>=2 && a[n-2]>1)f=0;

if(n>=2 && a[1]<-1)f=0;

if(f)cout<<"yes"<<endl;

else cout<<"no"<<endl;

}

int main() {

ios\_base::sync\_with\_stdio(0);

cin.tie(0); cout.tie(0);

#ifdef LOCAL

freopen("input.txt", "r", stdin);

freopen("output.txt", "w", stdout);

#endif

int tc; cin >> tc;

for (int t = 1; t <= tc; t++) {

//cout << "Case #" << t << ": ";

solve();

}

}