PROBLEMA 1

La complessità è O(nlog n).

#include <bits/stdc++.h>

using namespace std;

int count\_elem(int v[], int inizio, int fine, int elem);

int main(){

int n\_test;

int elem, dim;

int j=0;

int\* res;

cin>>n\_test;

res=new int[n\_test];

while(n\_test--){

cin>>elem;

cin>>dim;

int\* v= new int[dim];

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

cin>>v[i];

}

res[j]=count\_elem(v,0,dim-1,elem);

j++;

delete[] v;

}

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

cout<<res[i]<<endl;

}

delete[] res;

}

int count\_elem(int v[], int inizio, int fine, int elem){

if ((inizio > fine) || (inizio == fine && v[inizio] != elem)){

return 0;

}

if (inizio == fine && v[inizio] == elem){

return 1;

}

return count\_elem(v, inizio,(inizio + fine)/2,elem) + count\_elem(v,1+(inizio+fine)/2,fine,elem);

}

PROBLEMA 2

La complessità è O((S-P)^3)

#include <bits/stdc++.h>

using namespace std;

bool is\_prime(int n);

void somma\_primi(int N, int S, int P, int\* v, int k, int prec, int sum);

int main(){

int tot\_test;

int S, N, P;

int\* seq;

int index=0;

int n\_test=1;

cin>>tot\_test;

while(tot\_test--){

cin>>S>>N>>P;

seq=new int[N];

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

seq[i]=0;

}

cout<<"Caso di test "<<n\_test<<endl;

somma\_primi(N,S,P,seq,index,P+1,0);

n\_test++;

}

}

void somma\_primi(int N, int S, int P, int\* v, int k, int prec, int sum){

if(sum==S and k==N){

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

cout<<v[i]<<" ";

}

cout<<endl;

return;

}

if(sum>S or k>N){

return;

}

for(int i=prec; i<=S; i++){

if(is\_prime(i)){

v[k]=i;

sum+=i;

k++;

somma\_primi(N,S,P,v,k,i+1,sum);

//backtracking

k--;

sum-=i;

v[k]=0;

}

}

}

bool is\_prime(int n){

bool is\_prime=true;

if (n==0||n==1) {

is\_prime = false;

}

for(int i = 2; i <= n/2; ++i) {

if (n%i == 0) {

is\_prime = false;

break;

}

}

return is\_prime;

}