Number Theory - 3

Q.1)- https://www.codechef.com/COOK126B/problems/PTUPLES

-> Naive Approach

(a,b,c)

a,b,c->primes number and (a+b)=c;

1. Except 2 all the prime numbers are odd.
2. If all a,b,c are odd prime.

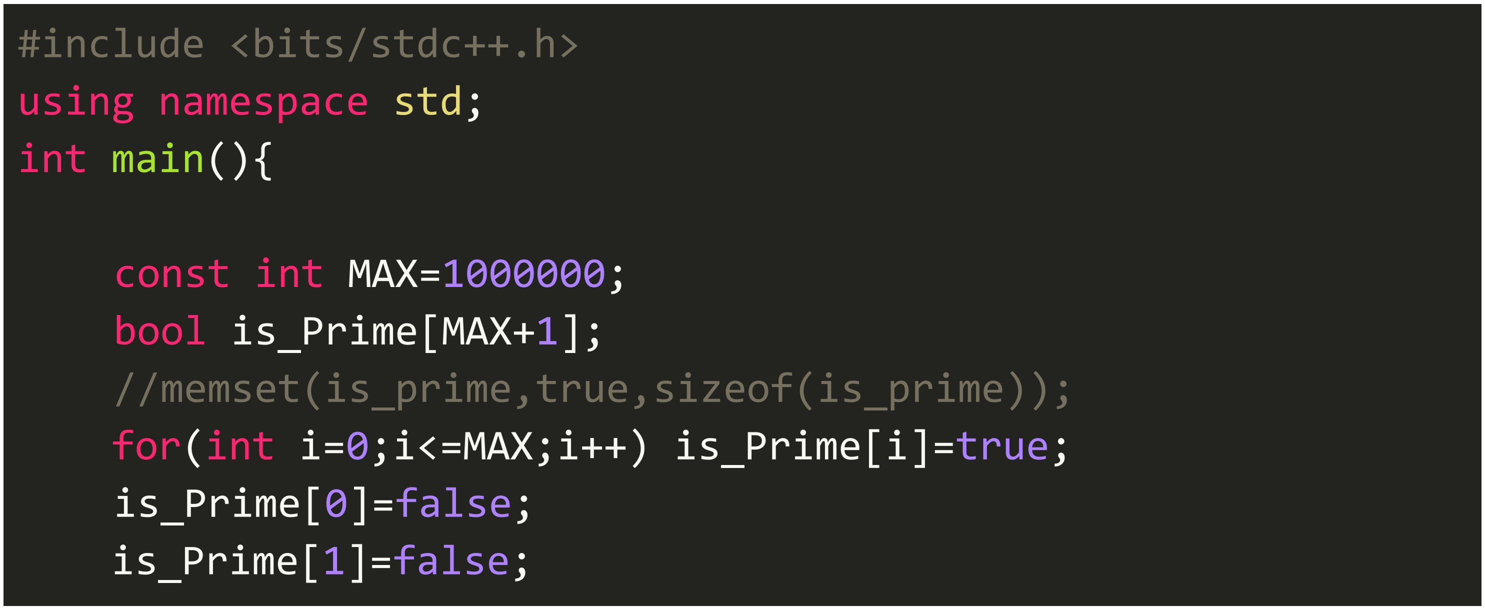
   Then for (a+b=c) (odd+odd=(even)=odd) not Possible.

   That's Why we must have to put one even-prime-number on LHS.

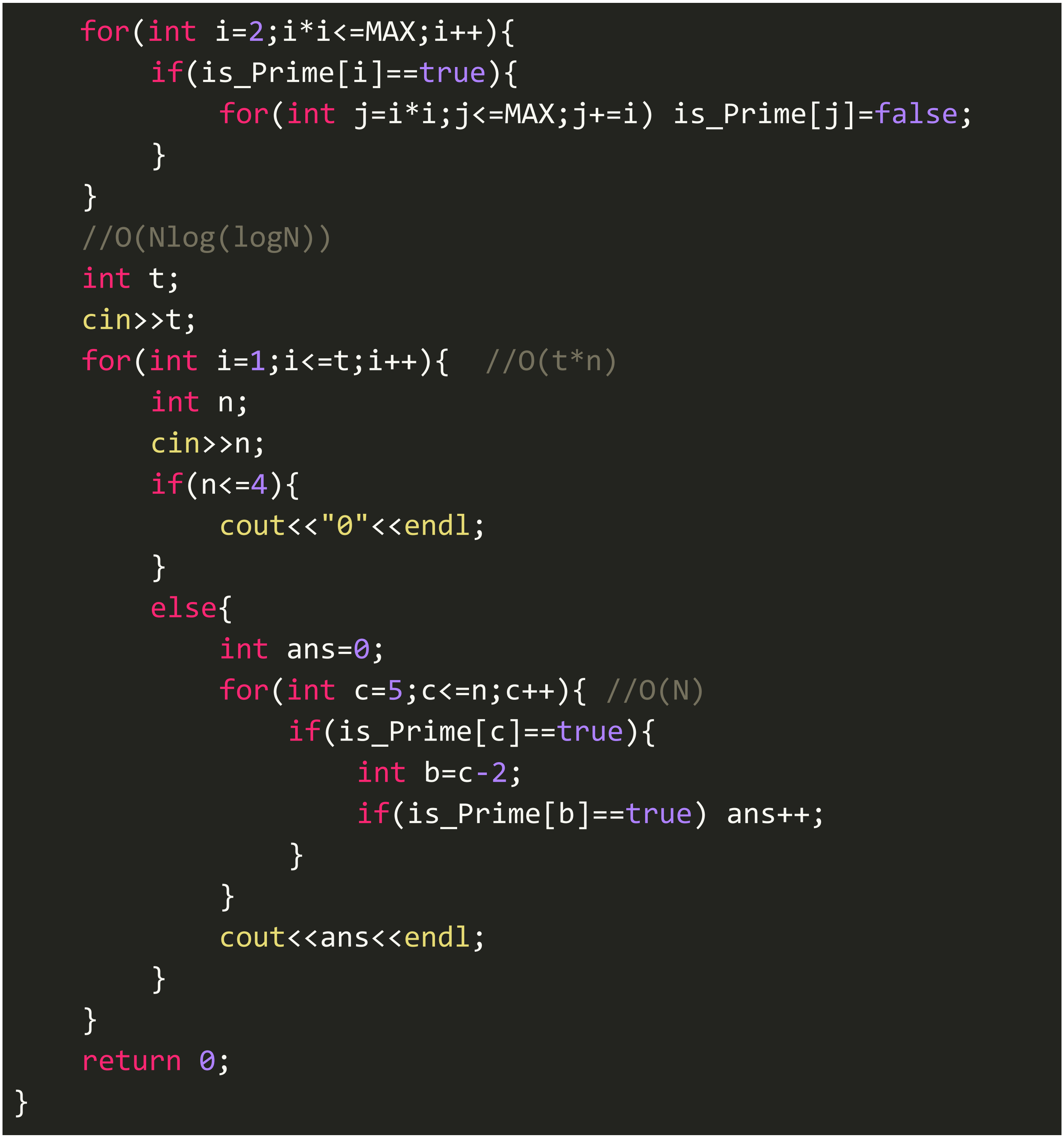
   And fortunately we have only one even-prime-number and which is 2.

   (2+b)=c; b=(c-2);

   i.e a=2;



```
#include <bits/stdc++.h>
using namespace std;
int main(){

    const int MAX=1000000;
    bool is_Prime[MAX+1];
    //memset(is_prime,true,sizeof(is_prime));
    for(int i=0;i<=MAX;i++) is_Prime[i]=true;
    is_Prime[0]=false;
    is_Prime[1]=false;
```



```
    for(int i=2;i*i<=MAX;i++){
        if(is_Prime[i]==true){
            for(int j=i*i;j<=MAX;j+=i) is_Prime[j]=false;
        }
    }
    //O(Nlog(logN))
    int t;
    cin>>t;
    for(int i=1;i<=t;i++){  //O(t*n)
        int n;
        cin>>n;
        if(n<=4){
            cout<<"0"<<endl;
        }
        else{
            int ans=0;
            for(int c=5;c<=n;c++){ //O(N)
                if(is_Prime[c]==true){
                    int b=c-2;
                    if(is_Prime[b]==true) ans++;
                }
            }
            cout<<ans<<endl;
        }
    }
    return 0;
}
```

// Overall Time complexity - O(T*N);

This is a slow Solution

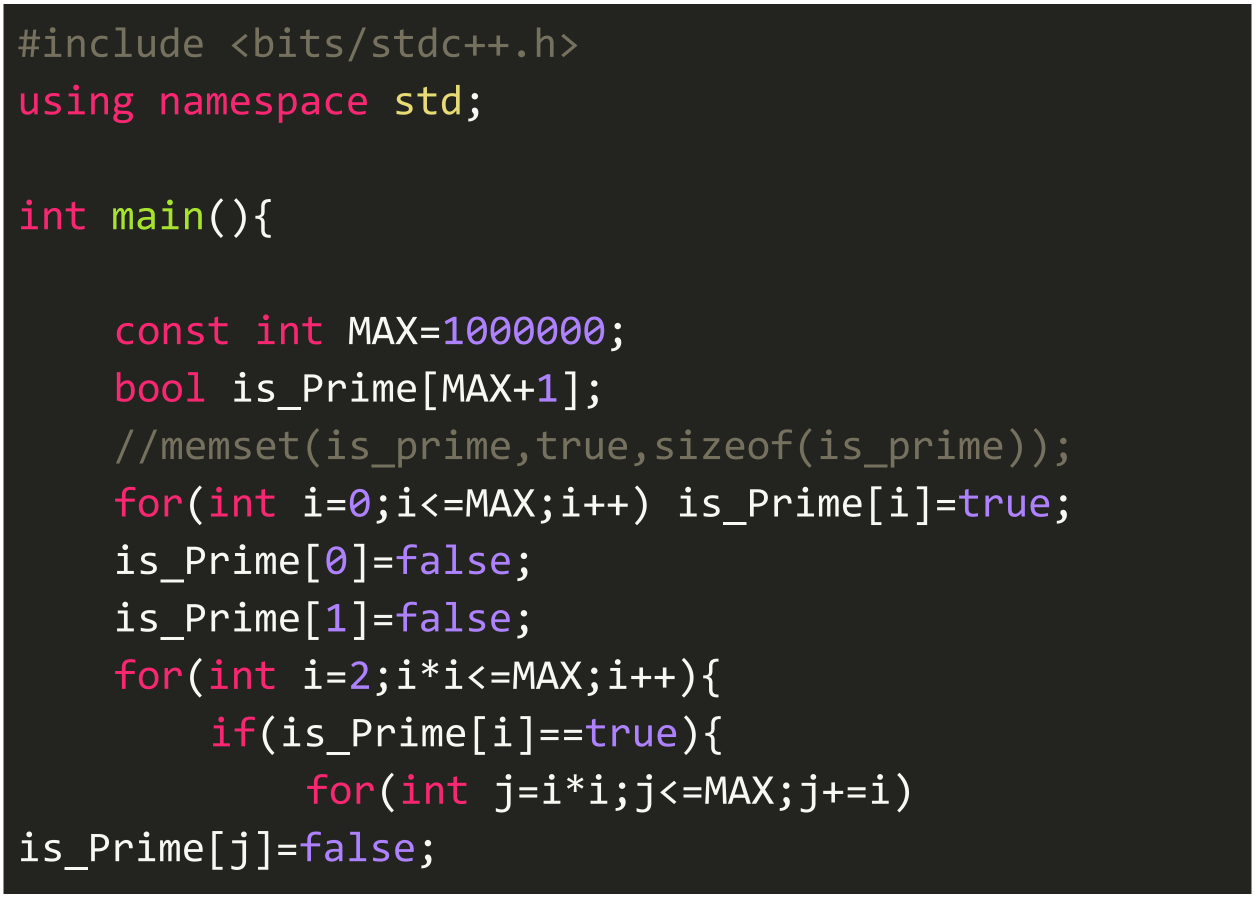
-> Points to Optimize the code.

Tuples[i]-> all possible tuples using numbers (1,2,3,4,5....i);

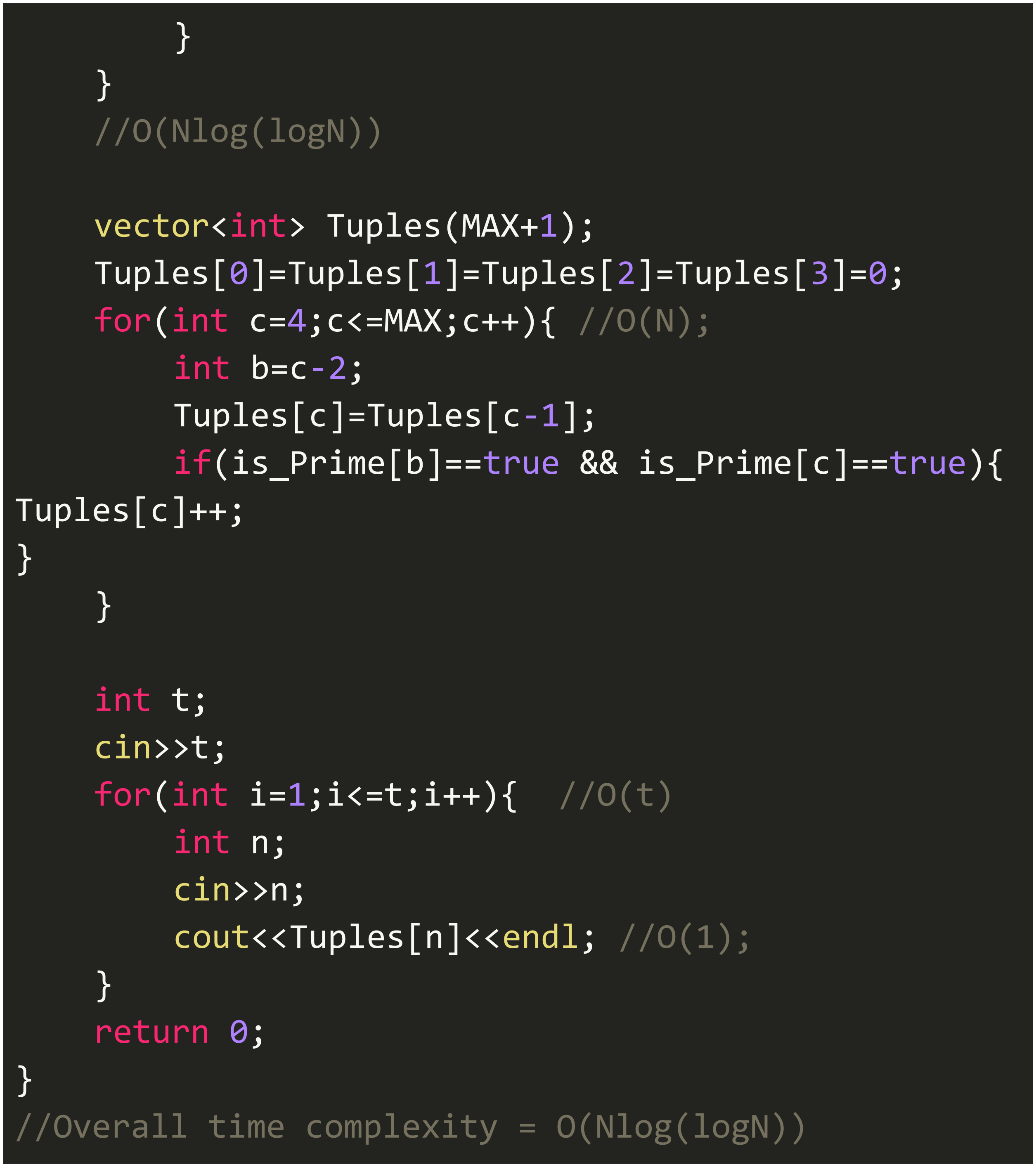
Points-> 1,2,3,4,5,6,7,8....n-1,n;

Tuple[n]=Tuples[n-1]+(the tuple form by using n);

5=>(2,3,5);
6=> (2,3,5)+(is it possible to get one tuple using 6?);



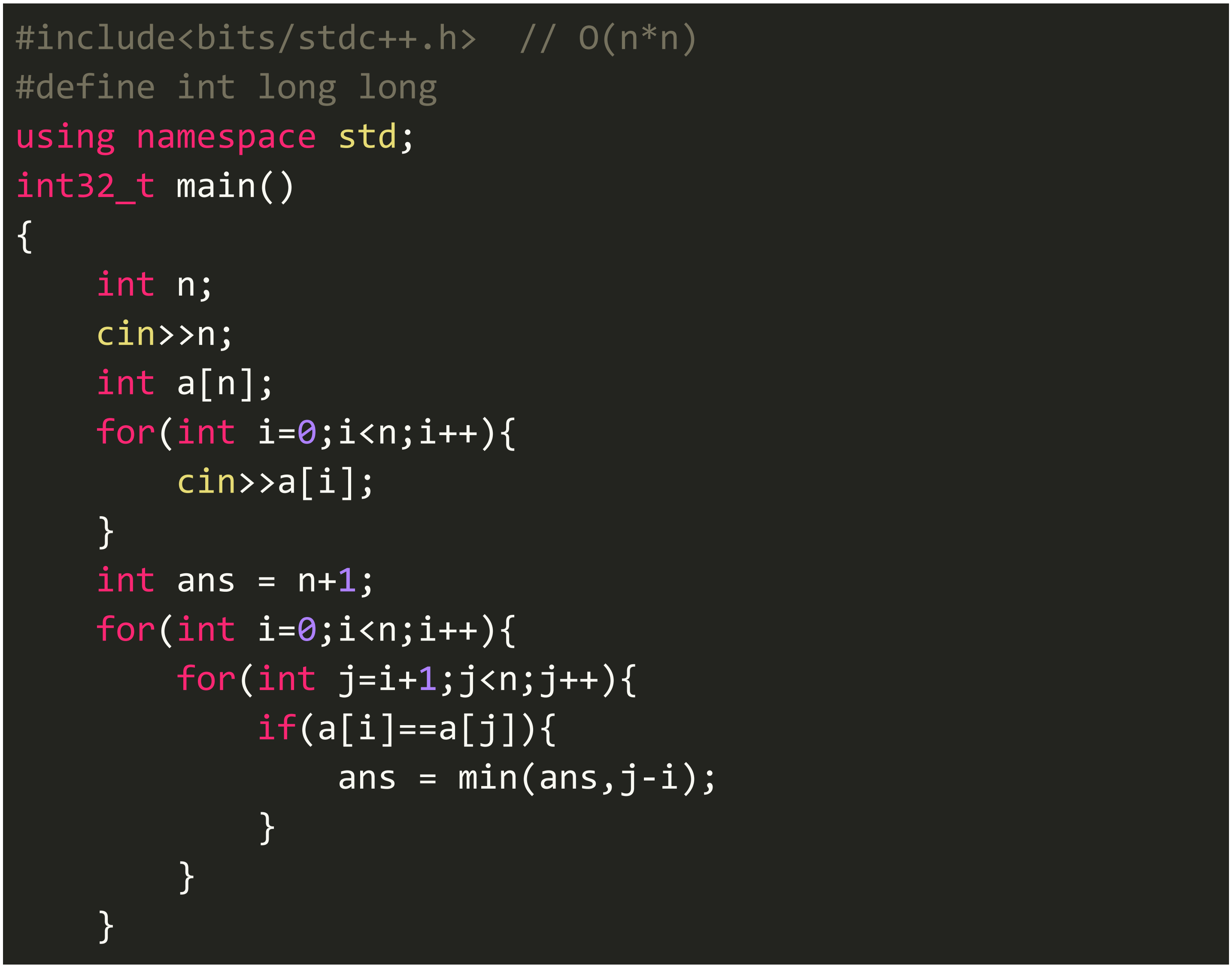
```
#include <bits/stdc++.h>
using namespace std;

int main(){

    const int MAX=1000000;
    bool is_Prime[MAX+1];
    //memset(is_prime,true,sizeof(is_prime));
    for(int i=0;i<=MAX;i++) is_Prime[i]=true;
    is_Prime[0]=false;
    is_Prime[1]=false;
    for(int i=2;i*i<=MAX;i++){
        if(is_Prime[i]==true){
            for(int j=i*i;j<=MAX;j+=i)
is_Prime[j]=false;
```



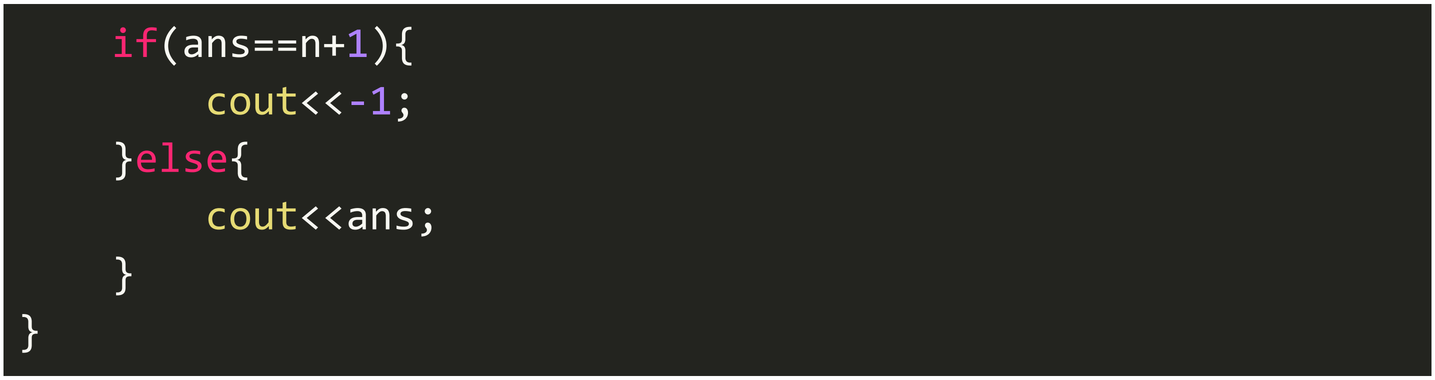
```
        }
    }
    //O(Nlog(logN))

    vector<int> Tuples(MAX+1);
    Tuples[0]=Tuples[1]=Tuples[2]=Tuples[3]=0;
    for(int c=4;c<=MAX;c++){ //O(N);
        int b=c-2;
        Tuples[c]=Tuples[c-1];
        if(is_Prime[b]==true && is_Prime[c]==true){
Tuples[c]++;
}
    }

    int t;
    cin>>t;
    for(int i=1;i<=t;i++){  //O(t)
        int n;
        cin>>n;
        cout<<Tuples[n]<<endl; //O(1);
    }
    return 0;
}
//Overall time complexity = O(Nlog(logN))
```

Fast Solution (Optimized Code)

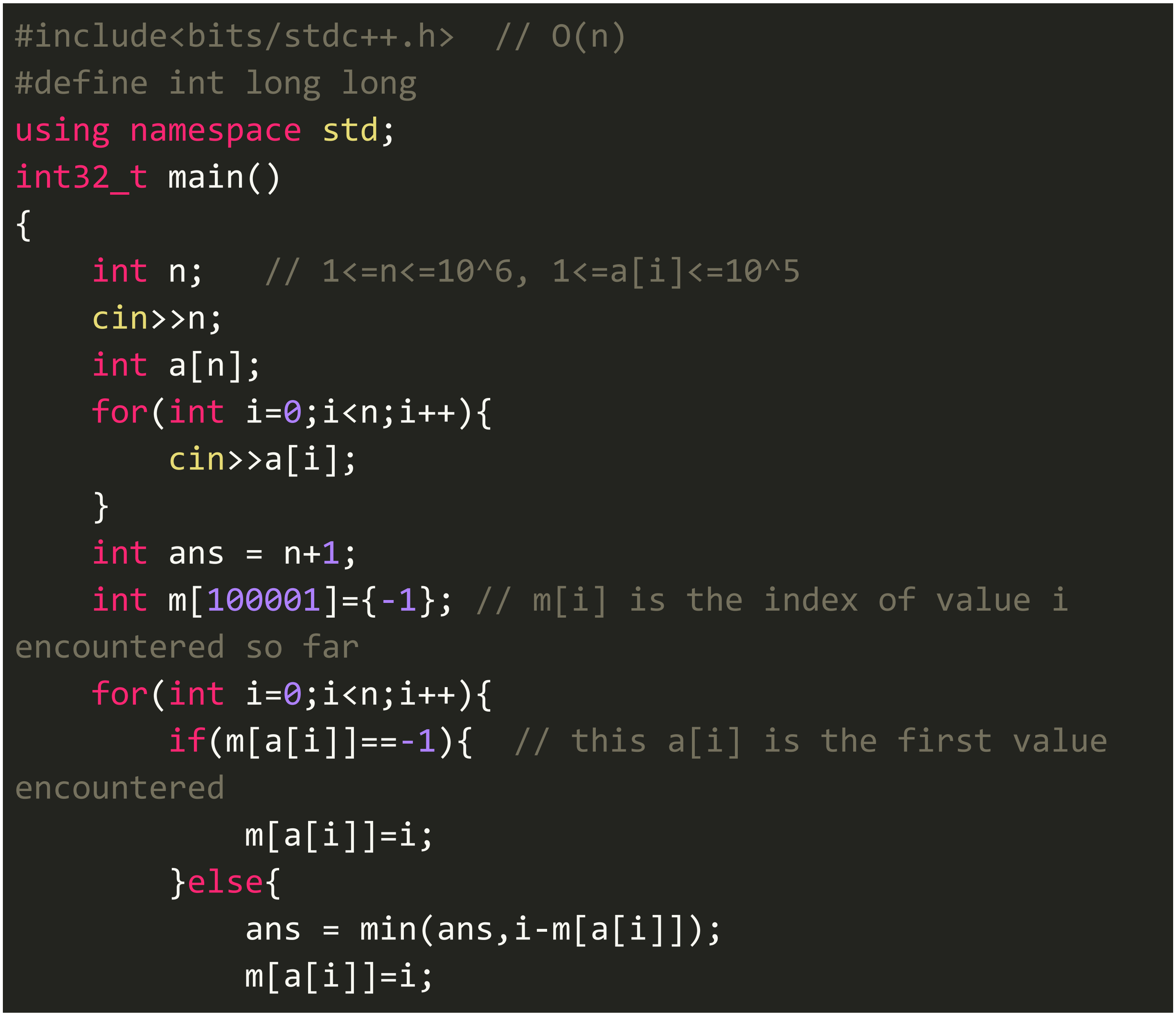
Q.2)-
https://www.hackerrank.com/challenges/minimum-distances/problem



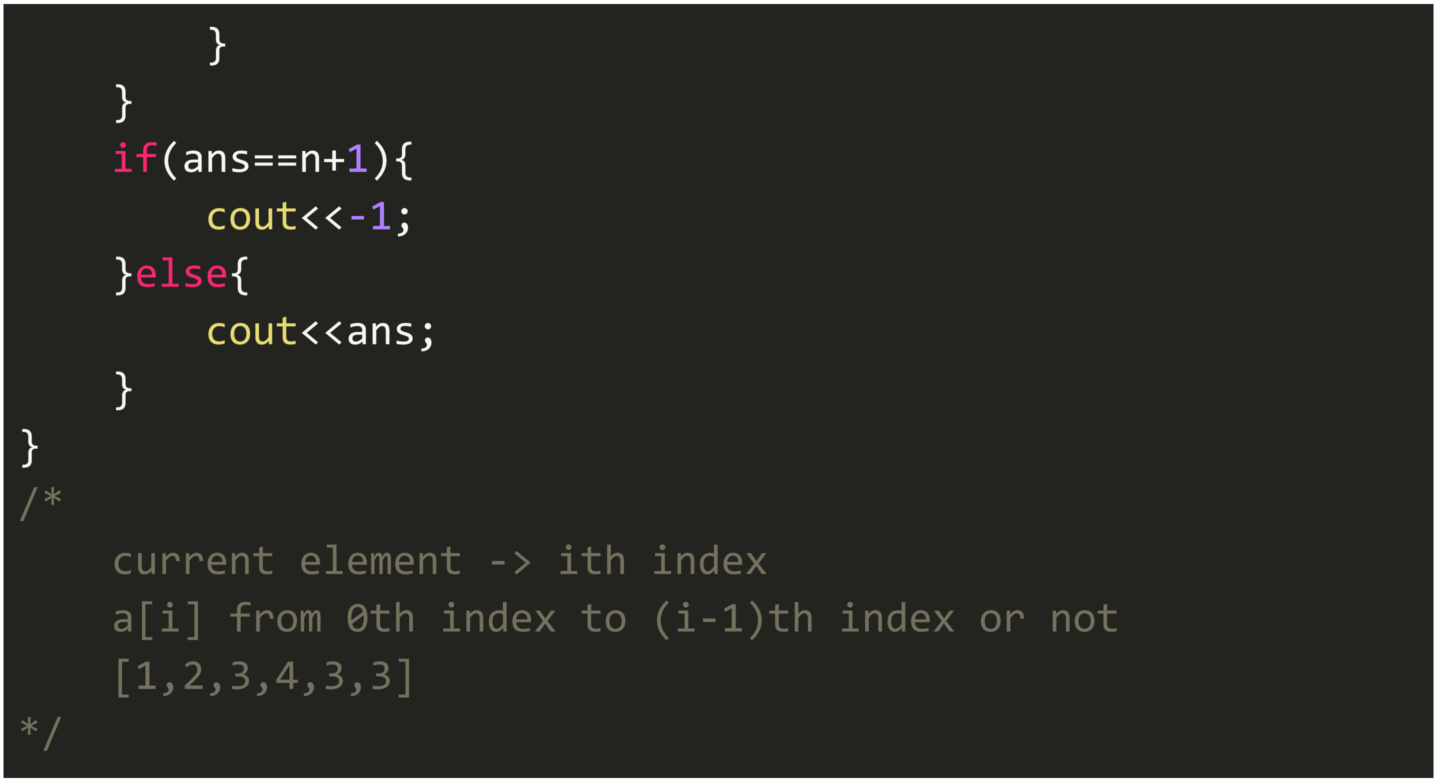
```cpp
#include<bits/stdc++.h>  // O(n*n)
#define int long long
using namespace std;
int32_t main()
{
    int n;
    cin>>n;
    int a[n];
    for(int i=0;i<n;i++){
        cin>>a[i];
    }
    int ans = n+1;
    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++){
            if(a[i]==a[j]){
                ans = min(ans,j-i);
            }
        }
    }
```



```cpp
    if(ans==n+1){
        cout<<-1;
    }else{
        cout<<ans;
    }
}
```



```cpp
#include<bits/stdc++.h>  // O(n)
#define int long long
using namespace std;
int32_t main()
{
    int n;   // 1<=n<=10^6, 1<=a[i]<=10^5
    cin>>n;
    int a[n];
    for(int i=0;i<n;i++){
        cin>>a[i];
    }
    int ans = n+1;
    int m[100001]={-1}; // m[i] is the index of value i
encountered so far
    for(int i=0;i<n;i++){
        if(m[a[i]]==-1){  // this a[i] is the first value
encountered
            m[a[i]]=i;
        }else{
            ans = min(ans,i-m[a[i]]);
            m[a[i]]=i;
```



```
        }
    }
    if(ans==n+1){
        cout<<-1;
    }else{
        cout<<ans;
    }
}
/*
    current element -> ith index
    a[i] from 0th index to (i-1)th index or not
    [1,2,3,4,3,3]
*/
```

Q.3)-
https://codeforces.com/problemset/problem/230/B

We know that prime numbers are positive integers that have exactly two distinct positive divisors. Similarly, we'll call a positive integer t T-prime, if t has exactly three distinct positive divisors.

You are given an array of n positive integers. For each of them determine whether it is T-prime or not.

The first line contains a single positive integer, n ($1 \le n \le 10$^5), showing how many numbers are in the array. The next line contains n space-separated integers xi ($1 \le xi \le 10$^12).

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is advised to use the `cin`, `cout` streams or the `%I64d` specifier.

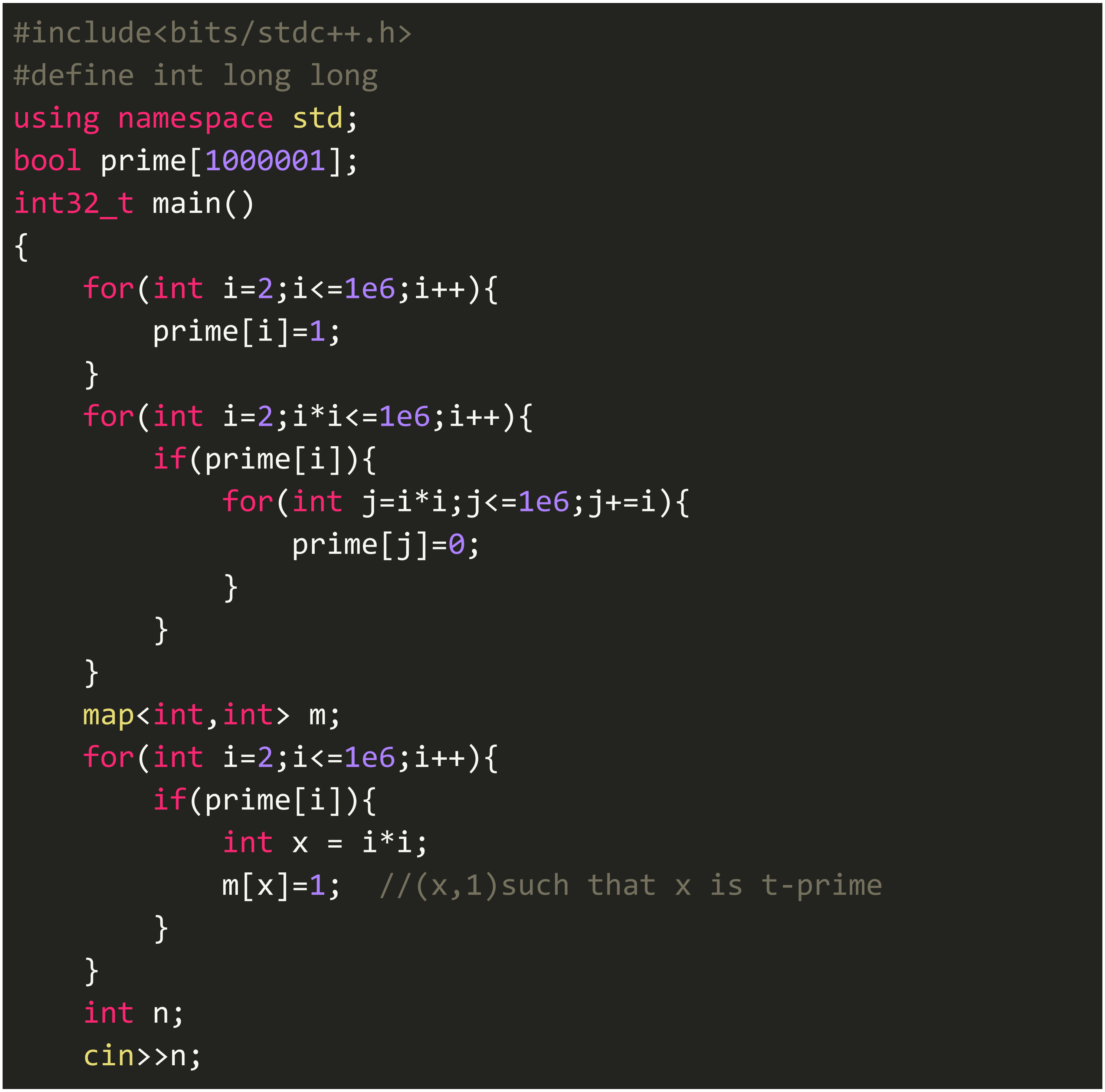
Eg 4-> 1,2,4

9 -> 1,3,9

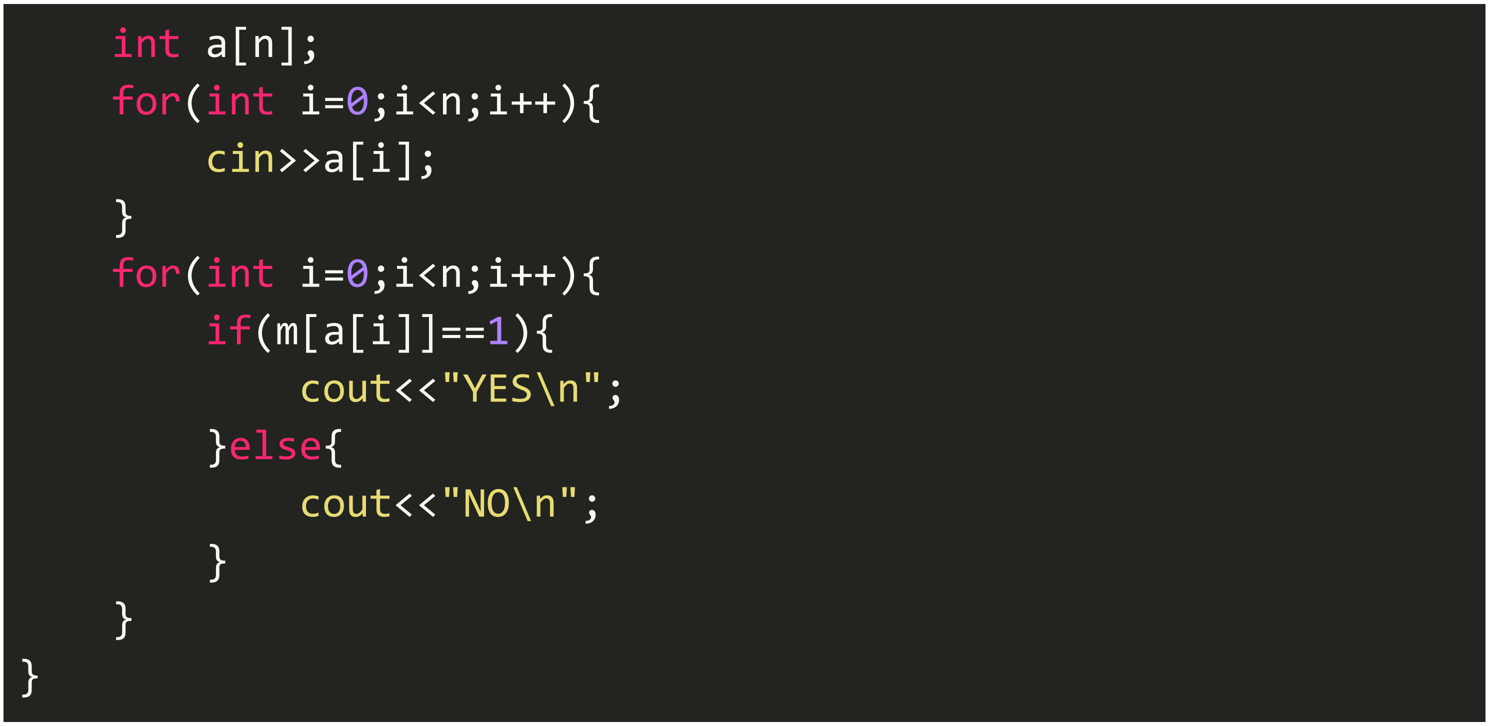
16-> 1,2,4,8,16

25-> 1,5,25

Sol:-



```
#include<bits/stdc++.h>
#define int long long
using namespace std;
bool prime[1000001];
int32_t main()
{
    for(int i=2;i<=1e6;i++){
        prime[i]=1;
    }
    for(int i=2;i*i<=1e6;i++){
        if(prime[i]){
            for(int j=i*i;j<=1e6;j+=i){
                prime[j]=0;
            }
        }
    }
    map<int,int> m;
    for(int i=2;i<=1e6;i++){
        if(prime[i]){
            int x = i*i;
            m[x]=1;  //(x,1)such that x is t-prime
        }
    }
    int n;
    cin>>n;
```



```
    int a[n];
    for(int i=0;i<n;i++){
        cin>>a[i];
    }
    for(int i=0;i<n;i++){
        if(m[a[i]]==1){
            cout<<"YES\n";
        }else{
            cout<<"NO\n";
        }
    }
}
```