**CODE**

#include <iostream>

using namespace std;

#define max 1000000

bool\* sieve;

bool check\_prime(int num){

if(num>max) return false;

return sieve[num];

}

int compute\_zigzag\_sum(int n,int\*\* matrix){

int sum=0;

for(int i=0;i<=2\*(n-1);i++){

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

int row,col;

if((i&1)==0){

row=i-j;

col=j;

}

else{

row=j;

col=i-j;

}

if(row>=0 && row<n && col>=0 && col<n){

int val=\*(\*(matrix+row)+col);

if(check\_prime(val)){

sum-=val;

}

else{

sum+=val;

}

}

}

}

return sum;

}

void generate\_sieve(){

sieve=new bool[max+1];

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

sieve[i]=true;

}

sieve[0]=sieve[1]=false;

for(int i=2;i\*i<=max;i++){

if(sieve[i]){

for(int j=i\*i;j<=max;j+=i){

sieve[j]=false;

}

}

}

}

int main(){

int n;

cout<<"Enter matrix size n(2 <= n <= 100):";

cin>>n;

if(n<2 || n>100){

cout<<"Invalid matrix size"<<endl;

return 0;

}

//matrix input by user

int\*\* matrix =new int\*[n];

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

\*(matrix+i)=new int[n];

}

cout<<"Enter "<< n\*n <<" matrix elements:"<<endl;

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

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

int val;

cin>>val;

if(val<1 || val>max){

cout<<"Element out of range"<<endl;

return 0;

}

\*(\*(matrix+i)+j)=val;

}

}

//check\_prime() -using sieve of eratosthenes

//compute\_zigzag\_sum()

generate\_sieve();

int result=compute\_zigzag\_sum(n,matrix);

cout<<"Zigzag Traversal sum:"<<result<<endl;

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

delete[] \*(matrix + i);

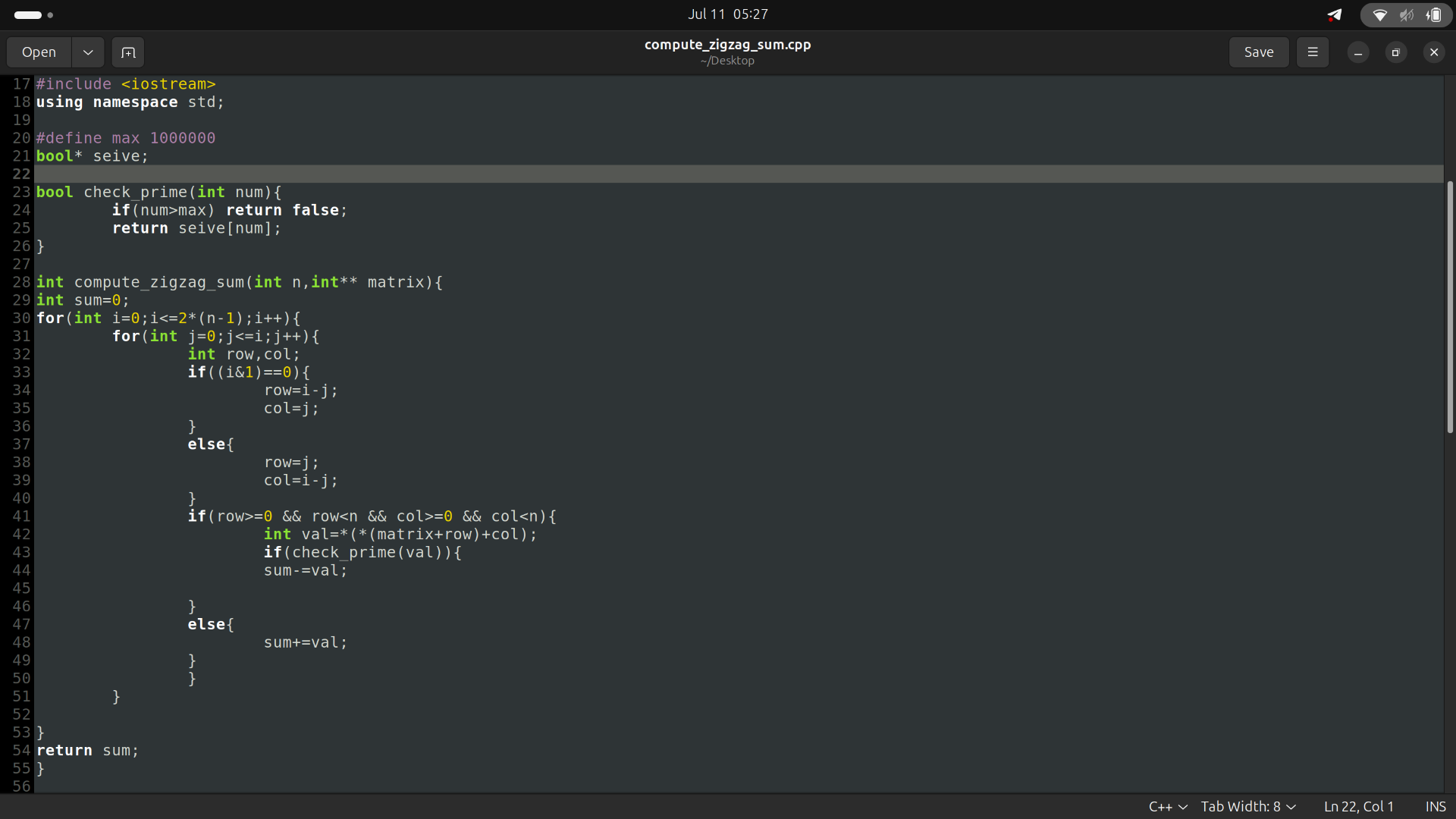
}

delete[] matrix;

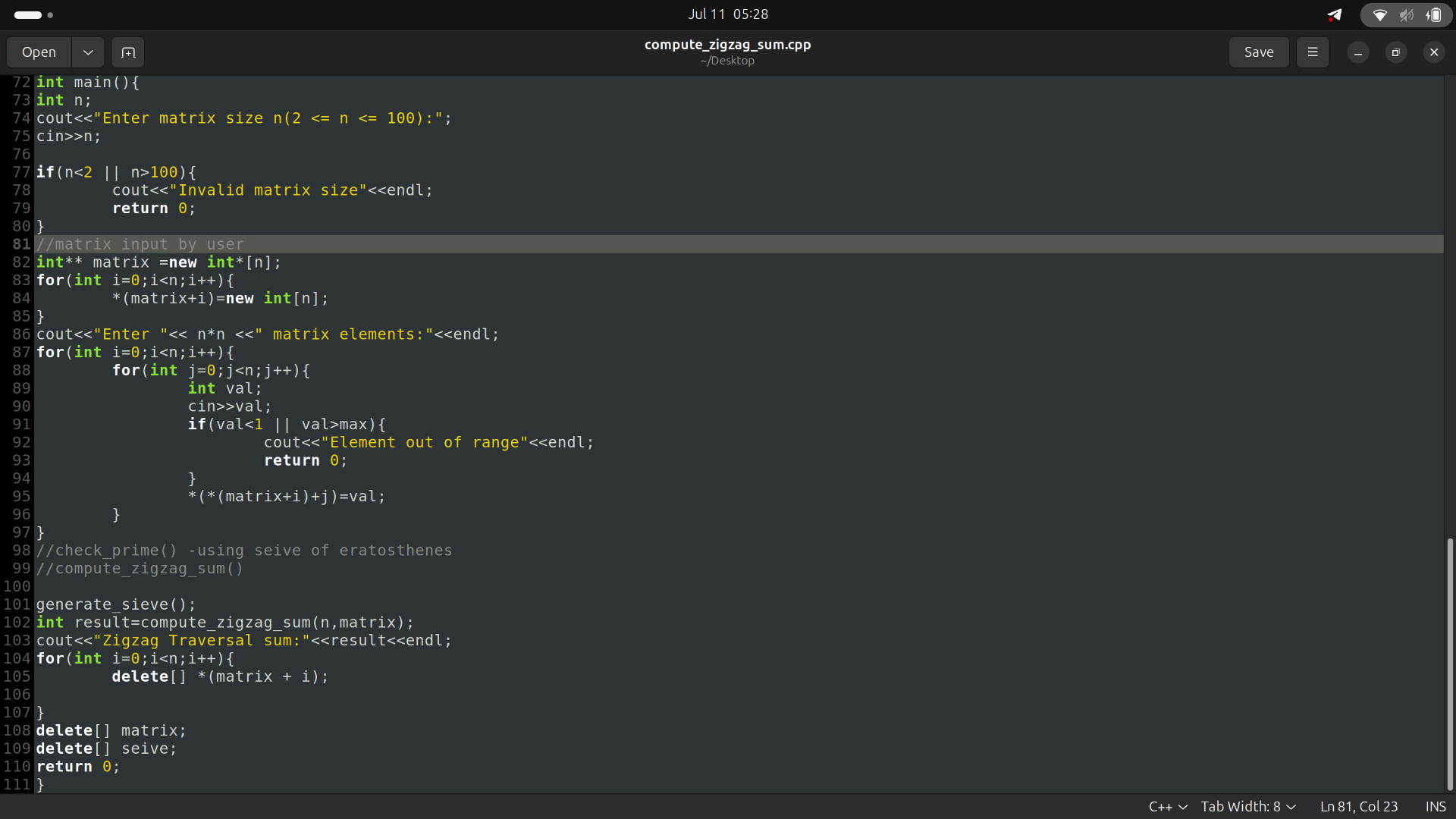
delete[] sieve;

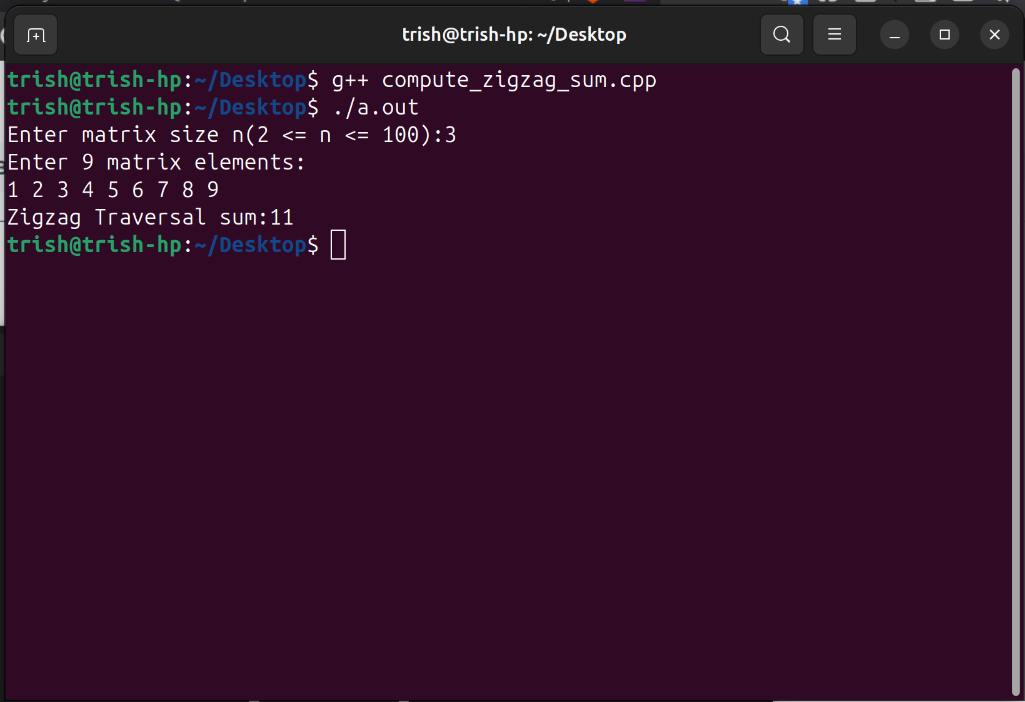
return 0;

}









**Github Repository Link :  
https://github.com/Trishna2005Das/GFG-iServeU-coding-round.git**