

**Code :**

```
#include <iostream>
using namespace std;

bool checkprime(int n){
    if (n<=1) return 0; //1 & 0 are not prime number

    if (n==2) return 1; //2 is a prime number

    for (int i=2; i<= (n/2); i++){
        if (n%i ==0) return 0; //not a prime hence false
    }
    return 1; //if the for loop is iterated completely, then its a prime
number hence true
}

int compute_zigzag_sum(int** matrix, int n){
    int sum=0;
    for (int d=0;d<=2*(n-1);d++){
        //d is the diagonal number, there are total of 2*n diagonals
in a nxn matrix

        if (d%2==0){
            //up-right
            int i= (d<n)?d:n-1;
            //initialise row value to current diagonal numberor else
boundary value
            int j=d-i; //here for each diagonal, d=i+j
            //sum of row and column number gives diagonal number
            while(i>=0 && j<n){
                int x= (*(matrix+i)+j); //matrix[i][j]
                //pointer access for i-th row and j-th element
                sum += checkprime(x)? -x :x;
                i--;
                j++;
            }
            //traverse through the diagonal in upward right
direction
        }
```

```

        }
    }
    else{
        //down-left
        int j=(d<n)?d:n-1;
        //initialise column to current diagonal or else
boundary value
        int i=d-j; //since d=i+j

        while(j>=0&& i<n){
            int x= (*(matrix+i)+j);
            sum+= checkprime(x)?-x:x;
            i++;
            j--;
            //traverse through the diagonal in downward left
direction
        }
    }
}
return sum;
}

```

```

int main (){
    int n;
    cout<<"Enter the number of rows for square matrix : ";
    cin>>n;
    int **matrix = new int*[n];
    //declare a 2d array with n rows

    for (int i=0; i<n;i++){
        matrix[i]=new int[n];
        //declare each row as an array of size n
    }

    cout<<"Enter "<<n<<" x "<<n<<" elements for the matrix :
"<<endl;
    for (int i=0;i<n;i++){

```

```

        for (int j=0;j<n;j++){
            cin>>*(matrix+i+j);
            //this will take input for matrix [i][j]

        }
    }

    int sum = compute_zigzag_sum(matrix,n);
    cout<<"The total sum of the elements is : "<< sum<<endl;

    //free allocated memory space
    for (int i=0;i<n;i++){
        delete[] matrix[i];
    }
    delete [] matrix;

    return 0;
}

```

## Code Screenshot :

```
19 bool checkprime(int n){
20     if (n<=1) return 0; //1 & 0 are not prime number
21
22     if (n==2) return 1; //2 is a prime number
23
24     for (int i=2; i<= (n/2); i++){
25         if (n%i ==0) return 0; //not a prime hence false
26     }
27     return 1; //if the for loop is iterated completely, then its a prime number hence true
28 }
29
30 int compute_zigzag_sum(int** matrix, int n){
31     int sum=0;
32     for (int d=0; d<=2*(n-1); d++){
33         //d is the diagonal number, there are total of 2*n diagonals in a nxn matrix
34
35         if (d%2==0){
36             //up-right
37             int i= (d<n)?d:n-1;
38             //initialise row value to current diagonal number or else boundary value
39             int j=d-i; //here for each diagonal, d=i+j
40             //sum of row and column number gives diagonal number
41             while(i>=0 && j<n){
42                 int x= *((matrix+i)+j); //matrix[i][j]
43                 //pointer access for i-th row and j-th element
44                 sum += checkprime(x)? -x : x;
45                 i--;
46                 j++;
47                 //traverse through the diagonal in upward right direction
48             }
49         }
50         else{
51             //down-left
52             int j=(d<n)?d:n-1;
53             //initialise column to current diagonal or else boundary value
54             int i=d-j; //since d=i+j
55
56             while(j>=0 && i<n){
57                 int x= *((matrix+i)+j);
58                 sum+= checkprime(x)? -x : x;
59                 i++;
60                 j--;
61                 //traverse through the diagonal in downward left direction
62             }
63         }
64     }
65     return sum;
66 }
67
68 int main (){
69     int n;
```

## Code Output :

```
Jul 11 2:32 AM
sammisam8888@SamHP: ~/Desktop
sammisam8888@SamHP:~$ cd Desktop/
sammisam8888@SamHP:~/Desktop$ g++ gdg-zigzag.cpp
sammisam8888@SamHP:~/Desktop$ ./a.out
Enter the number of rows for square matrix : 5
Enter 5 x 5 elements for the matrix :
10 20 30 40 50
11 21 31 52 61
84 95 62 65 17
51 67 12 97 34
55 29 43 68 89
The total sum of the elements is : 304
sammisam8888@SamHP:~/Desktop$
```

## Github Repository Link :

<https://github.com/Sammisam8888/GDG-BBSR-2025-Submission>