SOL 1:

#include <stdio.h>

void sort(int n, int\* ptr)

{

int i, j, t;

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

{

for (j = i + 1; j < n; j++)

{

if (\*(ptr + j) < \*(ptr + i))

{

t = \*(ptr + i);

\*(ptr + i) = \*(ptr + j);

\*(ptr + j) = t;

}

}

}

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

{

printf("%d ", \*(ptr + i));

}

}

int main()

{

int j,i,arr[100];

printf("Enter no. of elements: ");

scanf("%d",&j);

printf("Enter the elements of the array: ");

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

{

scanf("%d",&arr[i]);

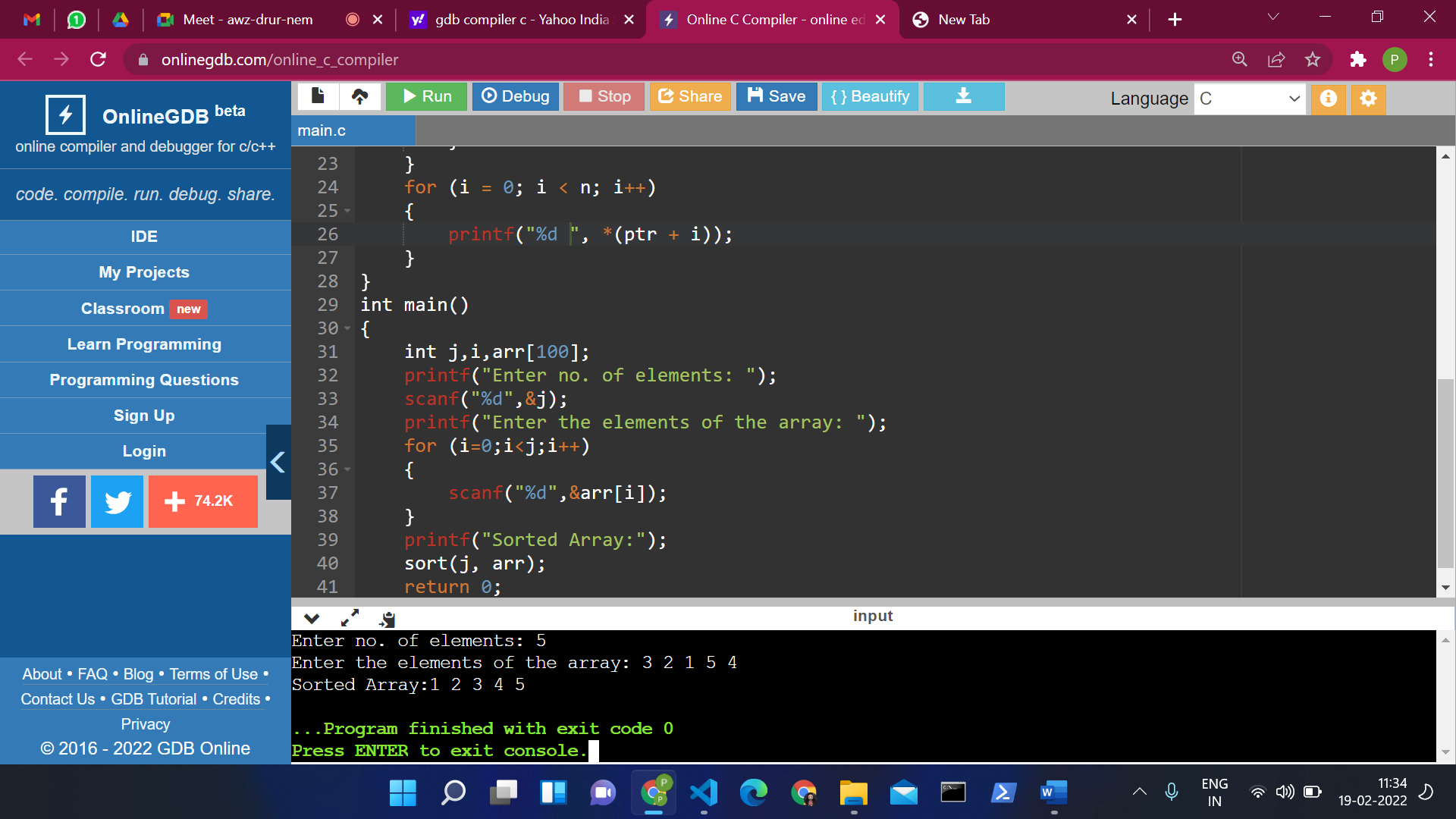
}

printf("Sorted Array:");

sort(j, arr);

return 0;

}



SOL 2:

#include<stdio.h>

int i,l;

int search(int ,int \*,int);

int main()

{

int n,m;

printf("Enter the size of array:");

scanf("%d",&n);

int a[n];

printf("Enter the elements of the array:");

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

{

scanf("%d",&a[i]);

}

printf("Enter the element to be searched:");

scanf("%d",&m);

search(n,a,m);

return 0;

}

int search(int n,int \*a,int m)

{

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

{

if(m==a[i])

{

l=1;

break;

}

}

if(l==1)

{

printf("%d is present in the array",m);

}

else

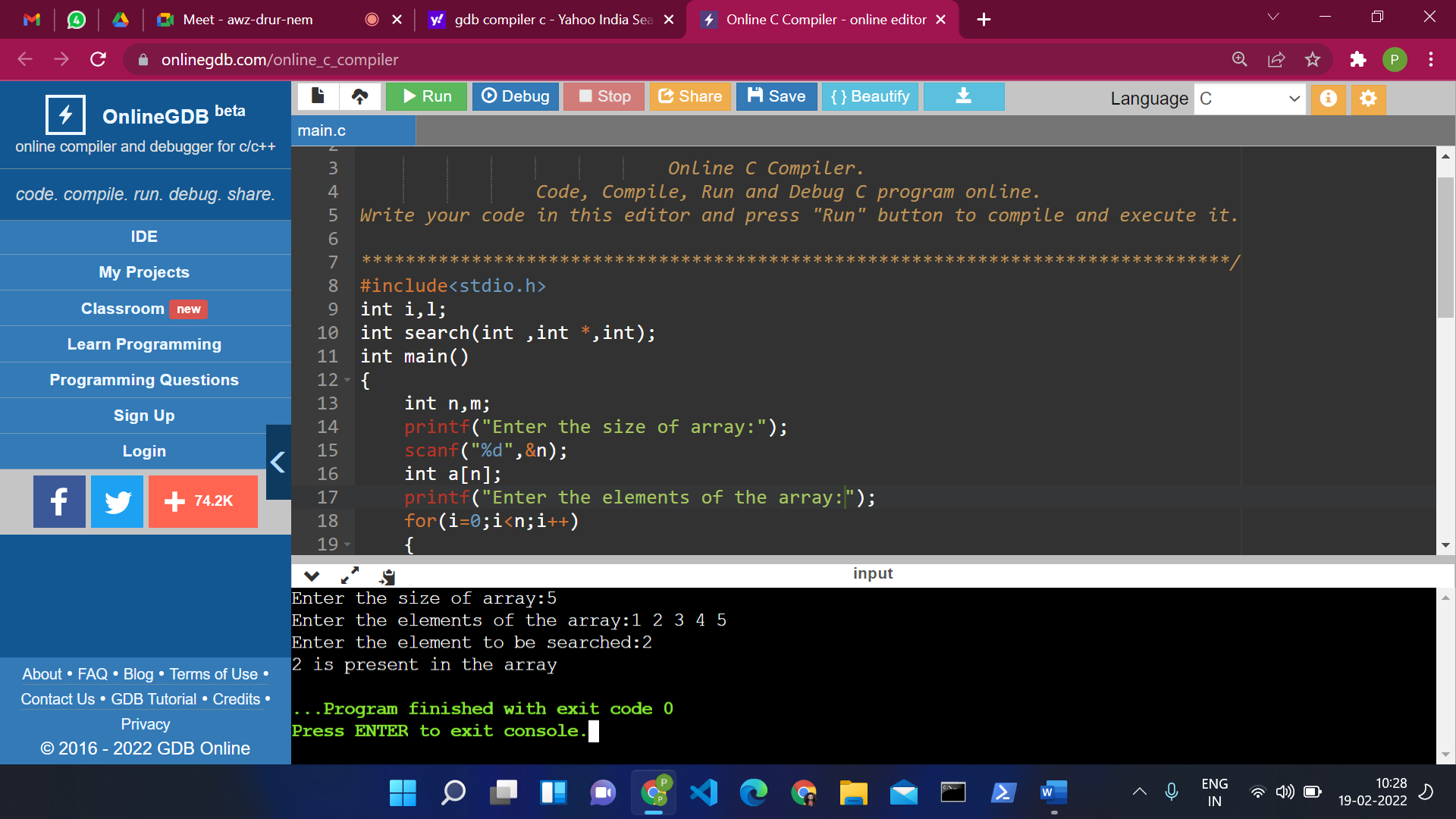
{

printf("%d is not present in the array",m);

}

return 0;

}



SOL 3:

#include <stdio.h>

#define ROW 3

#define COL 3

void matrixInput(int mat[][COL]);

void matrixPrint(int mat[][COL]);

void matrixMultiply(int mat1[][COL], int mat2[][COL], int res[][COL]);

int main()

{

int mat1[ROW][COL];

int mat2[ROW][COL];

int product[ROW][COL];

printf("Enter elements in first matrix of size %dx%d\n", ROW, COL);

matrixInput(mat1);

printf("Enter elements in second matrix of size %dx%d\n", ROW, COL);

matrixInput(mat2);

matrixMultiply(mat1, mat2, product);

printf("Product of both matrices is : \n");

matrixPrint(product);

return 0;

}

void matrixInput(int mat[][COL])

{

int row, col;

for (row = 0; row < ROW; row++)

{

for (col = 0; col < COL; col++)

{

scanf("%d", (\*(mat + row) + col));

}

}

}

void matrixPrint(int mat[][COL])

{

int row, col;

for (row = 0; row < ROW; row++)

{

for (col = 0; col < COL; col++)

{

printf("%d ", \*(\*(mat + row) + col));

}

printf("\n");

}

}

void matrixMultiply(int mat1[][COL], int mat2[][COL], int res[][COL])

{

int row, col, i;

int sum;

for (row = 0; row < ROW; row++)

{

for (col = 0; col < COL; col++)

{

sum = 0;

for (i = 0; i < COL; i++)

{

sum += (\*(\*(mat1 + row) + i)) \* (\*(\*(mat2 + i) + col));

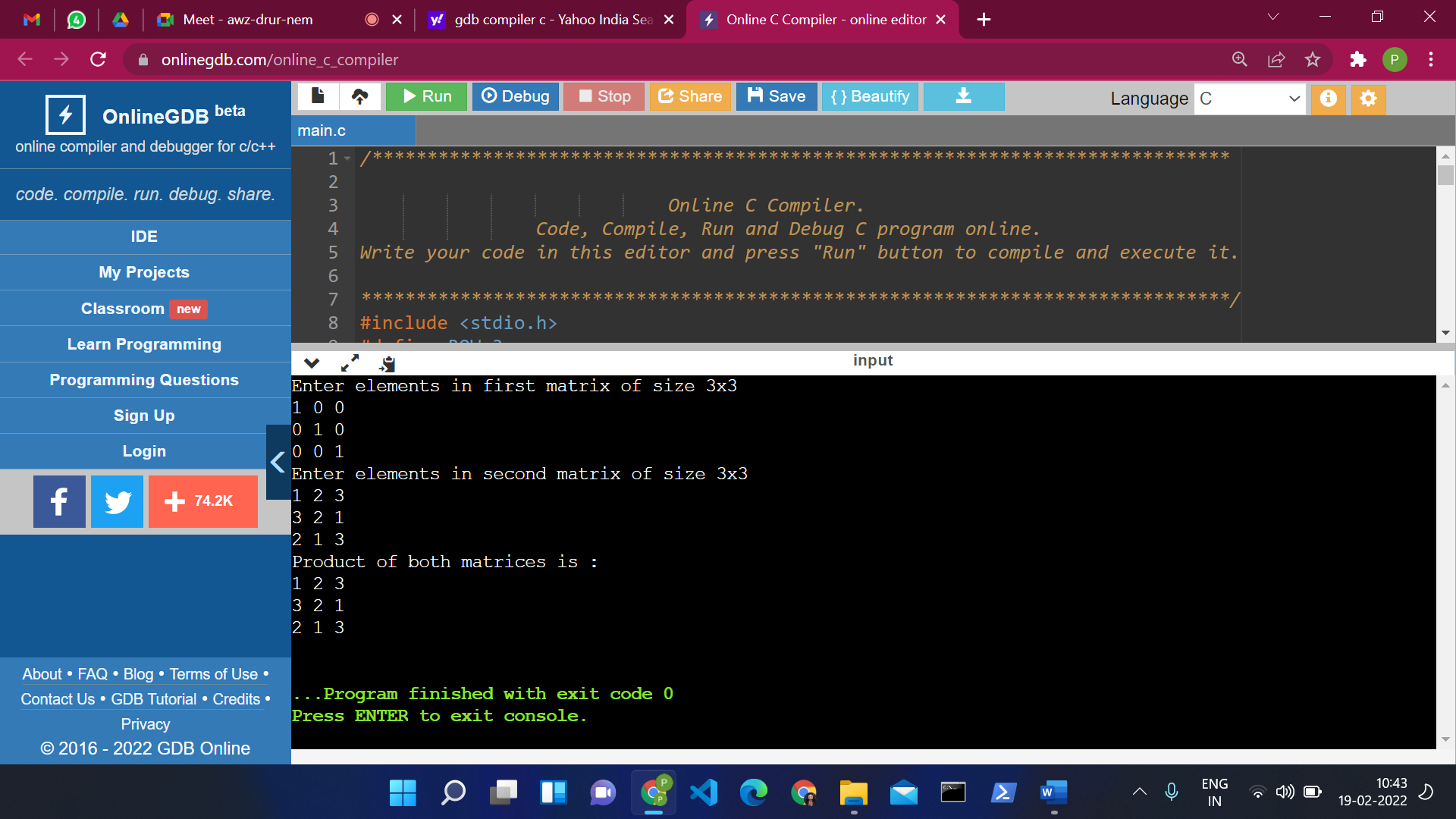
}

\*(\*(res + row) + col) = sum;

}

}

}



SOL 4:

#include<stdio.h>

void copy\_str(char\*, char\*);

main()

{

char str1[100], str2[100];

printf("Enter string to copy:");

gets(str1);

copy\_str(str2, str1);

printf("String after copying: %s", str2);

return 0;

}

void copy\_str(char \*str2, char \*str1)

{

while(\*str1)

{

\*str2 = \*str1;

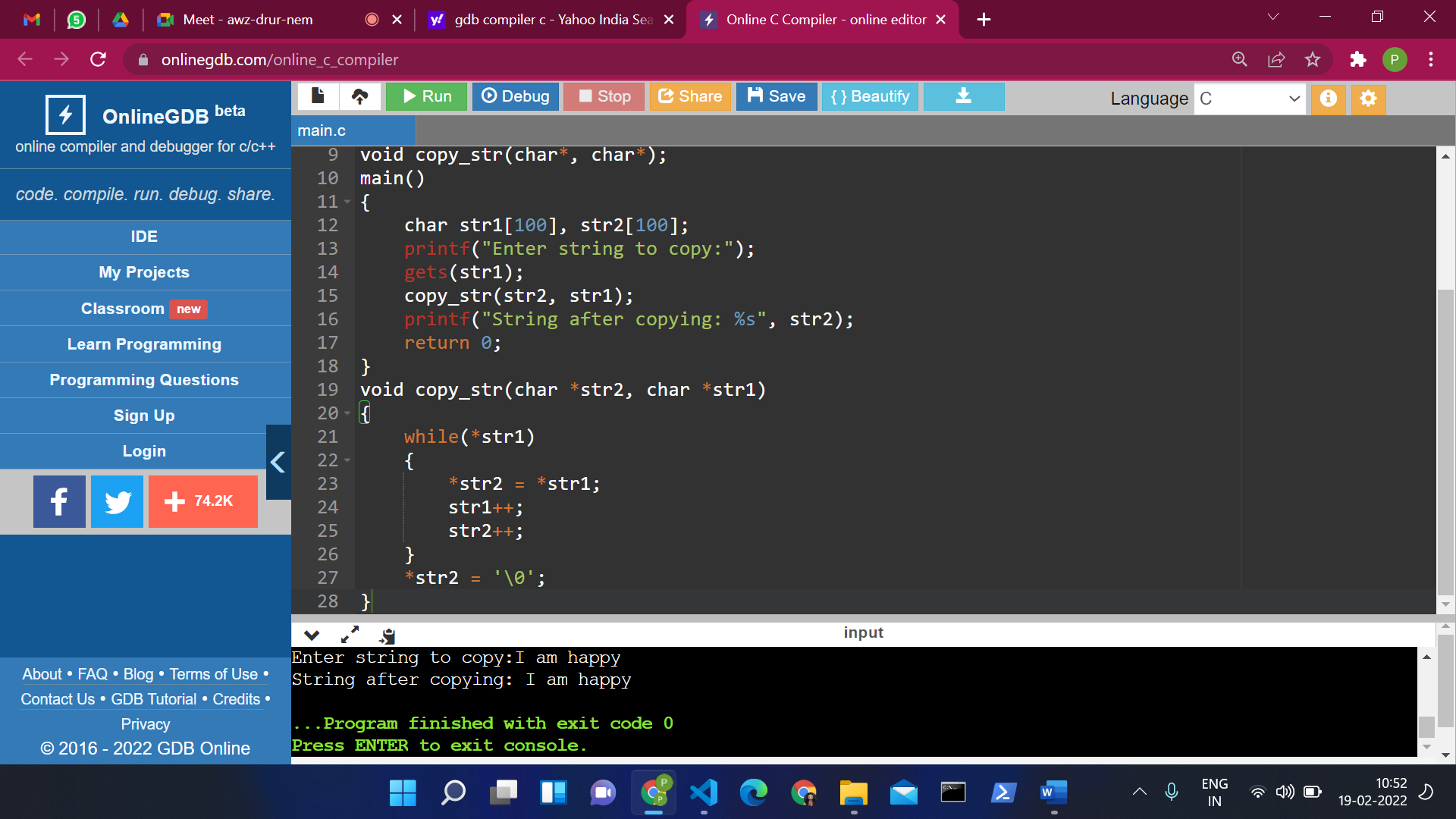
str1++;

str2++;

}

\*str2 = '\0';

}



SOL 5:

#include <stdio.h>

#include <string.h>

int main()

{

int len, i;

char str[100], \*start, \*end, ch;

printf("Enter a string: ");

gets(str);

len = strlen(str);

start = str;

end = str;

for (i = 0; i < len - 1; i++)

{

end++;

}

for (i = 0; i < len / 2; i++)

{

ch = \*end;

\*end = \*start;

\*start = ch;

start++;

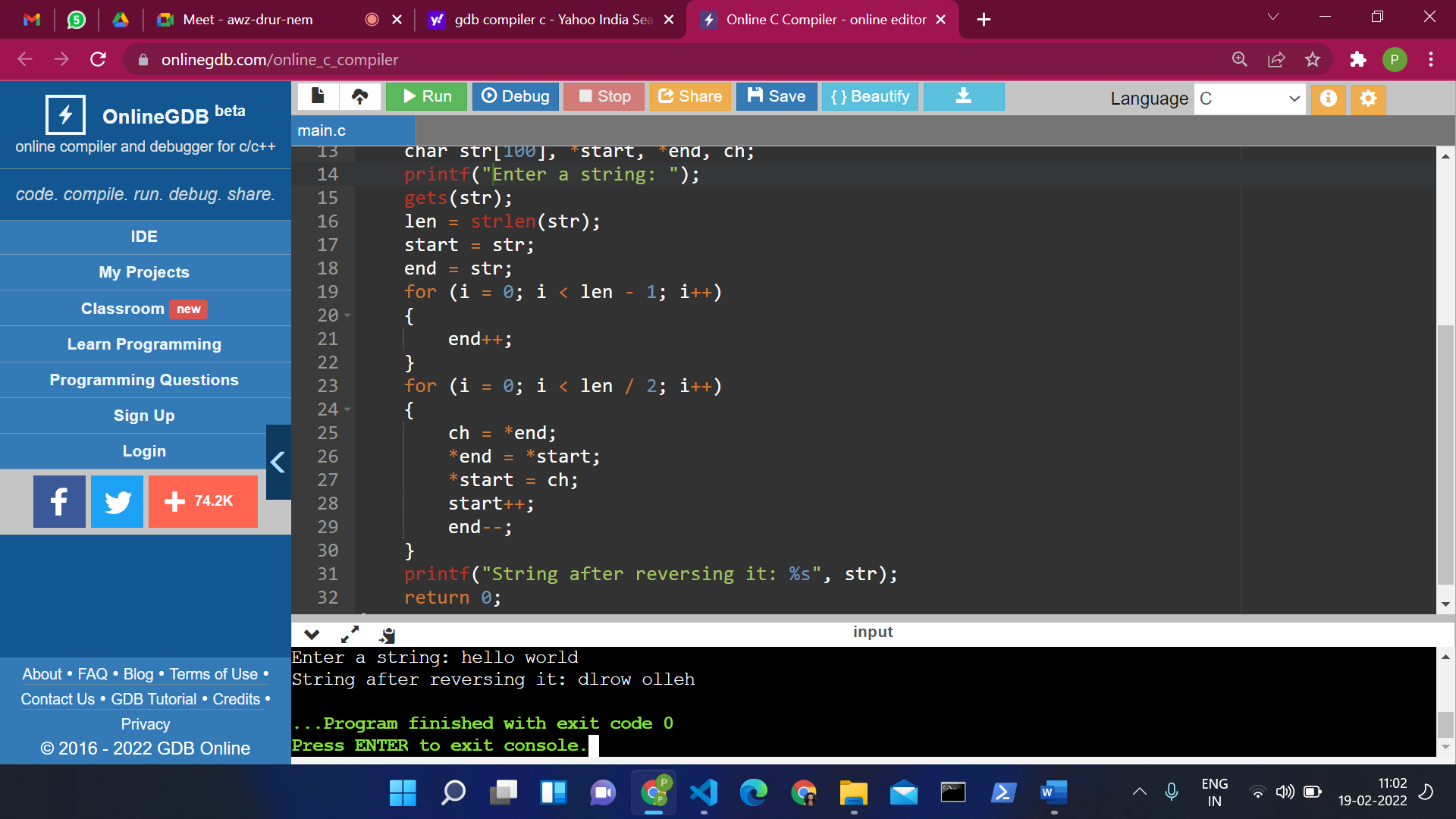
end--;

}

printf("String after reversing it: %s", str);

return 0;

}



SOL 6:

#include<stdio.h>

#include<conio.h>

void main()

{

int arr[100];

int i, sum = 0,n;

int \*ptr;

printf("Enter no. of elemnts:");

scanf("%d",&n);

printf("Enter the elements of the array: ");

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

{

scanf("%d", &arr[i]);

}

ptr = arr;

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

{

sum = sum + \*ptr;

ptr++;

}

printf("The sum of array elements : %d", sum);

}

