EXPERIMENT NO. : 3.2

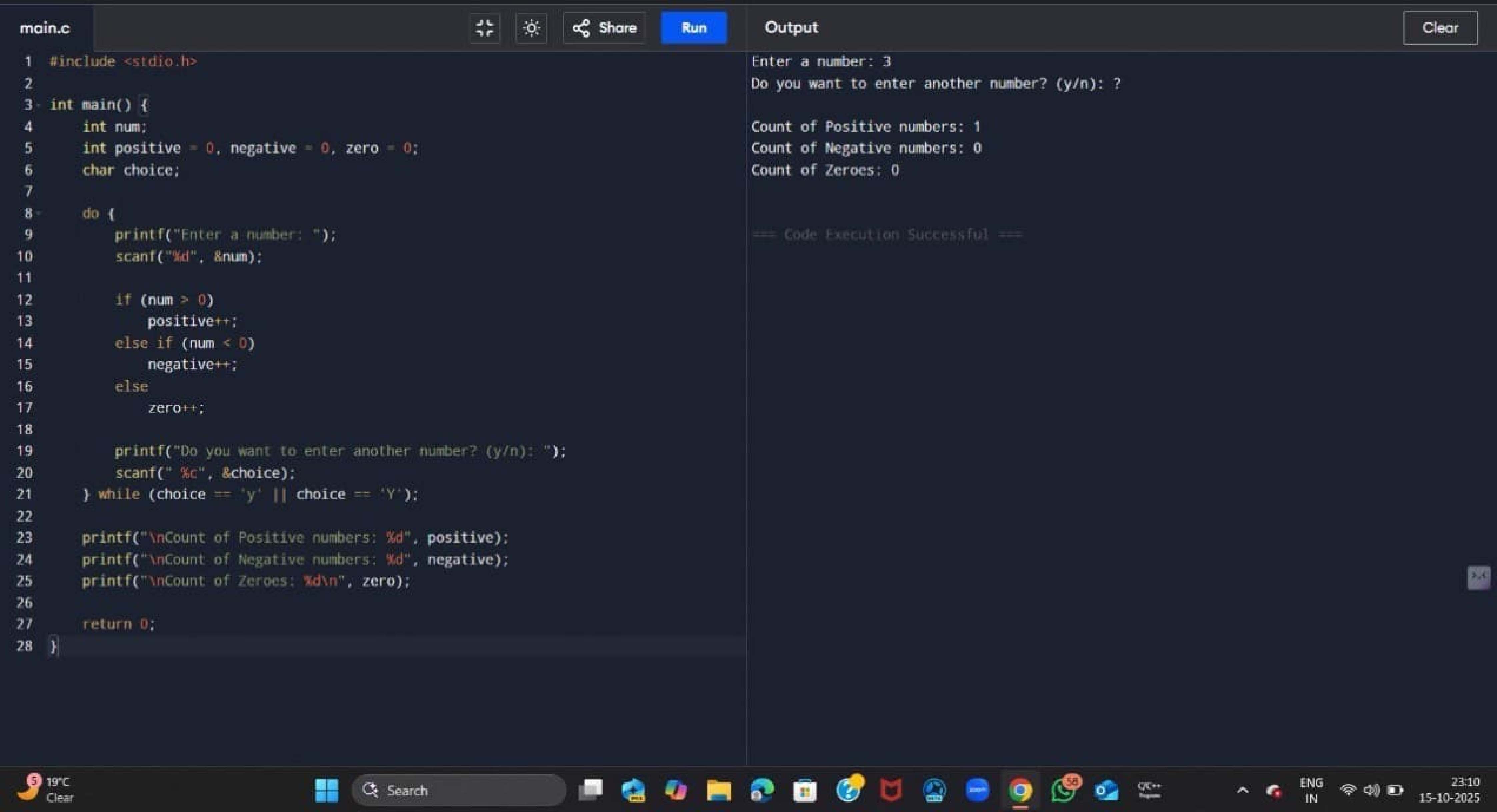
Loops

1) WAP to enter numbers till the user wants. At the end, it should display the count of positive, negative, and zeroes entered.

```
#include <stdio.h>

int main ( )
{
 int num;
 char choice;
 int pos = 0, neg = 0, zero = 0;

 do {
    printf ("enter a number : ");
    scanf ("%d", &num);

    if (num = 0)
    pos++;
    else if (num<0)
    neg++;
    else
    zero++;

    printf ("do you want to enter another number ?
                                   (y/n): ");
    scanf ("%c", &choice);
}
while (choice == 'y' || choice == 'y');

printf ("\n result \n");
printf ("positive : %d\n", pos);
printf ("negative : %d\n", neg);
printf ("zeros : %d\n", zero);

return 0;
}
```

Output :

```
enter a number : 3
do you want to enter another number ?
                                      (y/n) :

result
positive : 1
negative : 0
zeros    : 0
```



```c
#include <stdio.h>

int main() {
    int num;
    int positive = 0, negative = 0, zero = 0;
    char choice;

    do {
        printf("Enter a number: ");
        scanf("%d", &num);

        if (num > 0)
            positive++;
        else if (num < 0)
            negative++;
        else
            zero++;

        printf("Do you want to enter another number? (y/n): ");
        scanf(" %c", &choice);
    } while (choice == 'y' || choice == 'Y');

    printf("\nCount of Positive numbers: %d", positive);
    printf("\nCount of Negative numbers: %d", negative);
    printf("\nCount of Zeroes: %d\n", zero);

    return 0;
}
```

```
Enter a number: 3
Do you want to enter another number? (y/n): ?

Count of Positive numbers: 1
Count of Negative numbers: 0
Count of Zeroes: 0


=== Code Execution Successful ===
```

2). WAP to print the multiplication table of the number entered by the user. It should be in the correct formatting. Num * 1 = Num

```
#include <stdio.h>
int main ()
{
 int num, i;

 printf ("enter a number : ");
 scanf ("%d", &num);

 printf ("\n multiplication table of %d: \n", num);

 for (i=1; i<=10; i++){

   printf ("%d * %d = %d\n", num, i, num*i);
 }

 return 0;
}
```

Output :-

```
enter a number : 20

multiplication table of 20:

      20 * 1 = 20
      20 * 2 = 40
      20 * 3 = 60
```

20 * 4 = 80

20 * 5 = 100

20 * 6 = 120

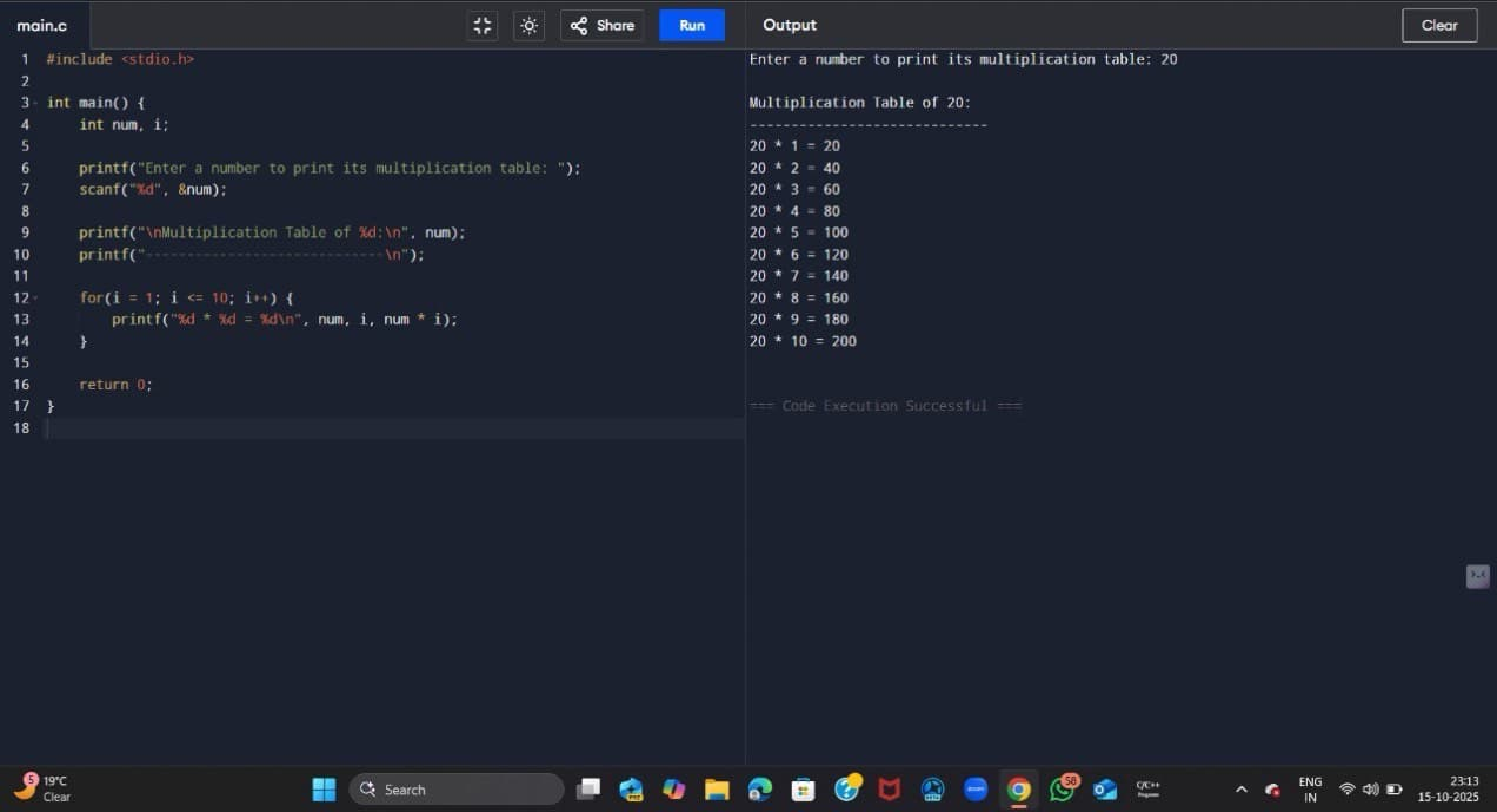
20 * 7 = 140

20 * 8 = 160

20 * 9 = 180

20 * 6 = 20

Teacher's Signature



```c
#include <stdio.h>

int main() {
    int num, i;

    printf("Enter a number to print its multiplication table: ");
    scanf("%d", &num);

    printf("\nMultiplication Table of %d:\n", num);
    printf("-----------------------------\n");

    for(i = 1; i <= 10; i++) {
        printf("%d * %d = %d\n", num, i, num * i);
    }

    return 0;
}
```

```
Enter a number to print its multiplication table: 20

Multiplication Table of 20:
-----------------------------
20 * 1 = 20
20 * 2 = 40
20 * 3 = 60
20 * 4 = 80
20 * 5 = 100
20 * 6 = 120
20 * 7 = 140
20 * 8 = 160
20 * 9 = 180
20 * 10 = 200


=== Code Execution Successful ===
```

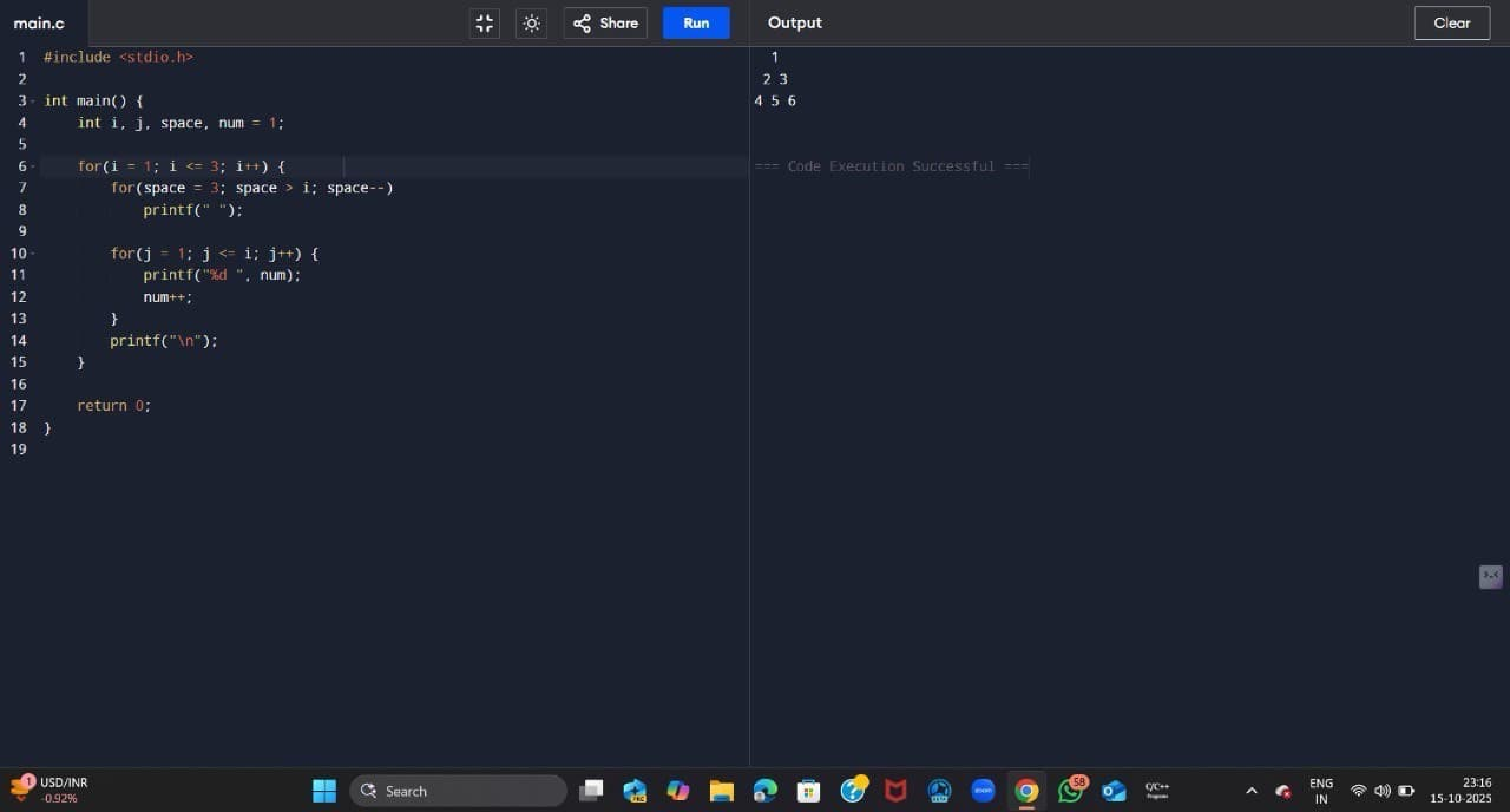
3) WAP to generate the following set of output

a)
```
        1
     2     3
  4     5     6
```

```
# include <stdio.h>
int main ( )
{
 int rows, i, j, space, num = 1;

 printf ("Enter number of rows : ");
 scanf (" %d", & rows );

 for (i = 1 ; i <= rows ; i ++) {
         printf ("   ");
 }
 for (j = 1 ; j <= 1 ; j ++) {
     printf ("%d\t", num);
     num ++ ;
 }
 printf ("\n");
 }
 return 0;
 }
```

Output :-

```
Enter number of rows : 3
             1
          2     3
       4     5     6
```



```c
#include <stdio.h>

int main() {
    int i, j, space, num = 1;

    for(i = 1; i <= 3; i++) {
        for(space = 3; space > i; space--)
            printf(" ");

        for(j = 1; j <= i; j++) {
            printf("%d ", num);
            num++;
        }
        printf("\n");
    }

    return 0;
}
```

```
  1
 2 3
4 5 6


=== Code Execution Successful ===
```

b)

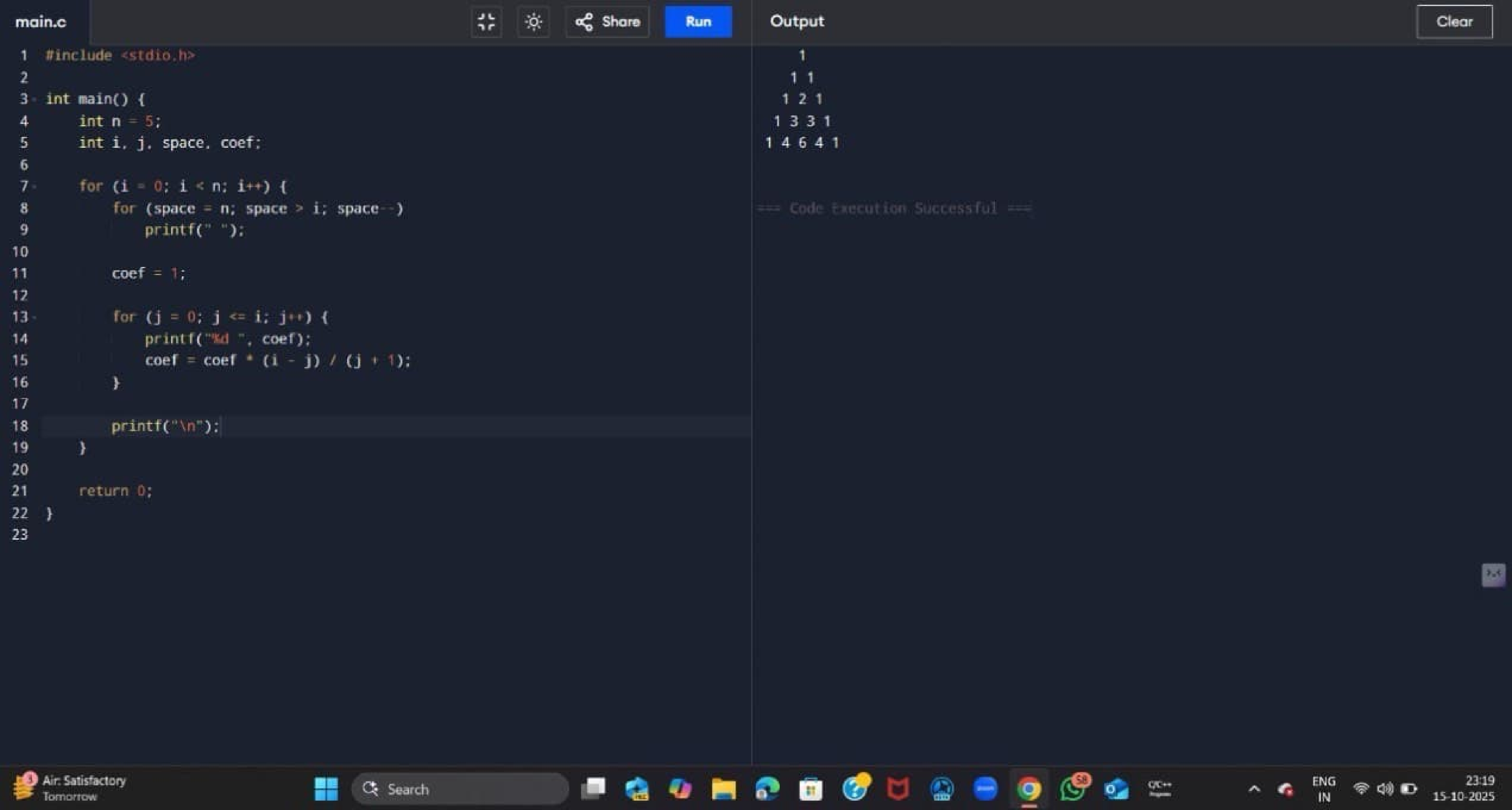
```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

```
# include <stdio.h>
int main ( )
{
int rows, coef = 1, space, i, j;

printf ("enter number of rows : ");
scanf (" %d", & rows);

for (i = 0 ; i < rows ; i++)
{
for (space = 1 ; space <= 5-i; space ++)
  printf ("     ");

 for (j = 0 ; j <= i ; j++) {
   if (j == 0 || i == 0)
   coef = 1 ;
   else
   coef = coef * (i-j+1) / j ;

   printf (" %4d\t", coef);
  }
  printf ("\n");
```

```
    }
    return 0;
}
```

Output :-

```
Enter number of rows : 5

            1
          1   1
        1   2   1
      1   3   3   1
    1   4   6   4   1
```



```c
#include <stdio.h>

int main() {
    int n = 5;
    int i, j, space, coef;

    for (i = 0; i < n; i++) {
        for (space = n; space > i; space--)
            printf(" ");

        coef = 1;

        for (j = 0; j <= i; j++) {
            printf("%d ", coef);
            coef = coef * (i - j) / (j + 1);
        }

        printf("\n");
    }

    return 0;
}
```

```
     1 
    1 1 
   1 2 1 
  1 3 3 1 
 1 4 6 4 1 


=== Code Execution Successful ===
```

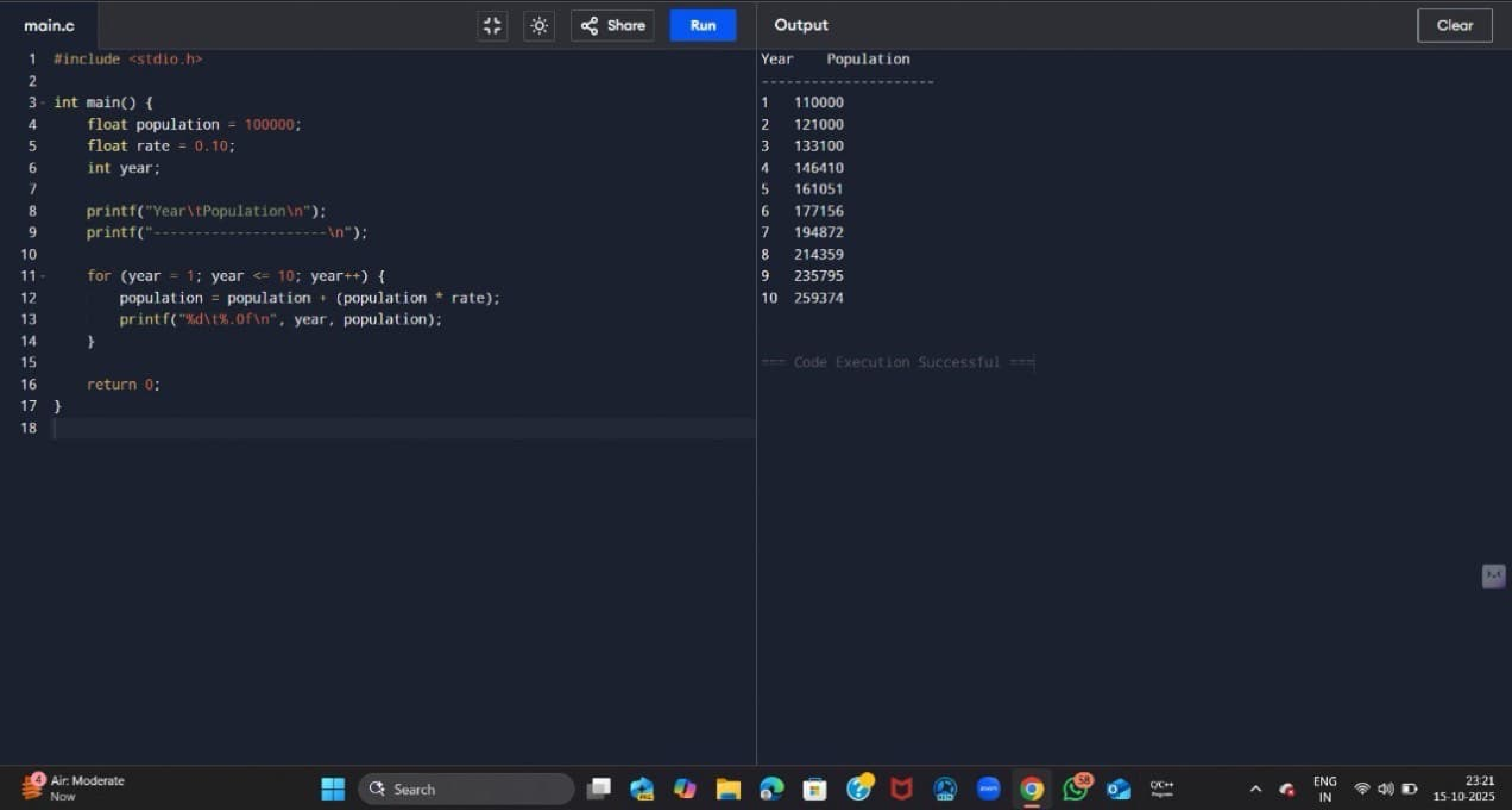
4). The population of a town is 100000. The population has increased steadily at the rate of 10 % per year for the last 10 years. Write a program to determine the population at the end of each year in the last decade

```
# include <stdio.h>
int main ( )
{
int year ;
double population = 100000;

printf ("Year \t Population \n");
for (year =1 ; year <= 10 ; year ++)
{
population = population * 1.10;
printf ("%d \t %.0f\n", year, population);
}

return 0;
}
```

Output :-

```
Year     Population

1        110000
2        121000
3        133100
4        146410
```

5	161051
6	177156
7	194872
8	214359
9	235795
10	259374



```c
#include <stdio.h>

int main() {
    float population = 100000;
    float rate = 0.10;
    int year;

    printf("Year\tPopulation\n");
    printf("--------------------\n");

    for (year = 1; year <= 10; year++) {
        population = population + (population * rate);
        printf("%d\t%.0f\n", year, population);
    }

    return 0;
}
```

```
Year    Population
--------------------
1       110000
2       121000
3       133100
4       146410
5       161051
6       177156
7       194872
8       214359
9       235795
10      259374


=== Code Execution Successful ===
```

5). Ramanujan Number is the smallest number that can be expressed as the sum of two cubes in two different ways. WAP to print all such numbers up to a reasonable limit.

Example of Ramanujan number : 1729

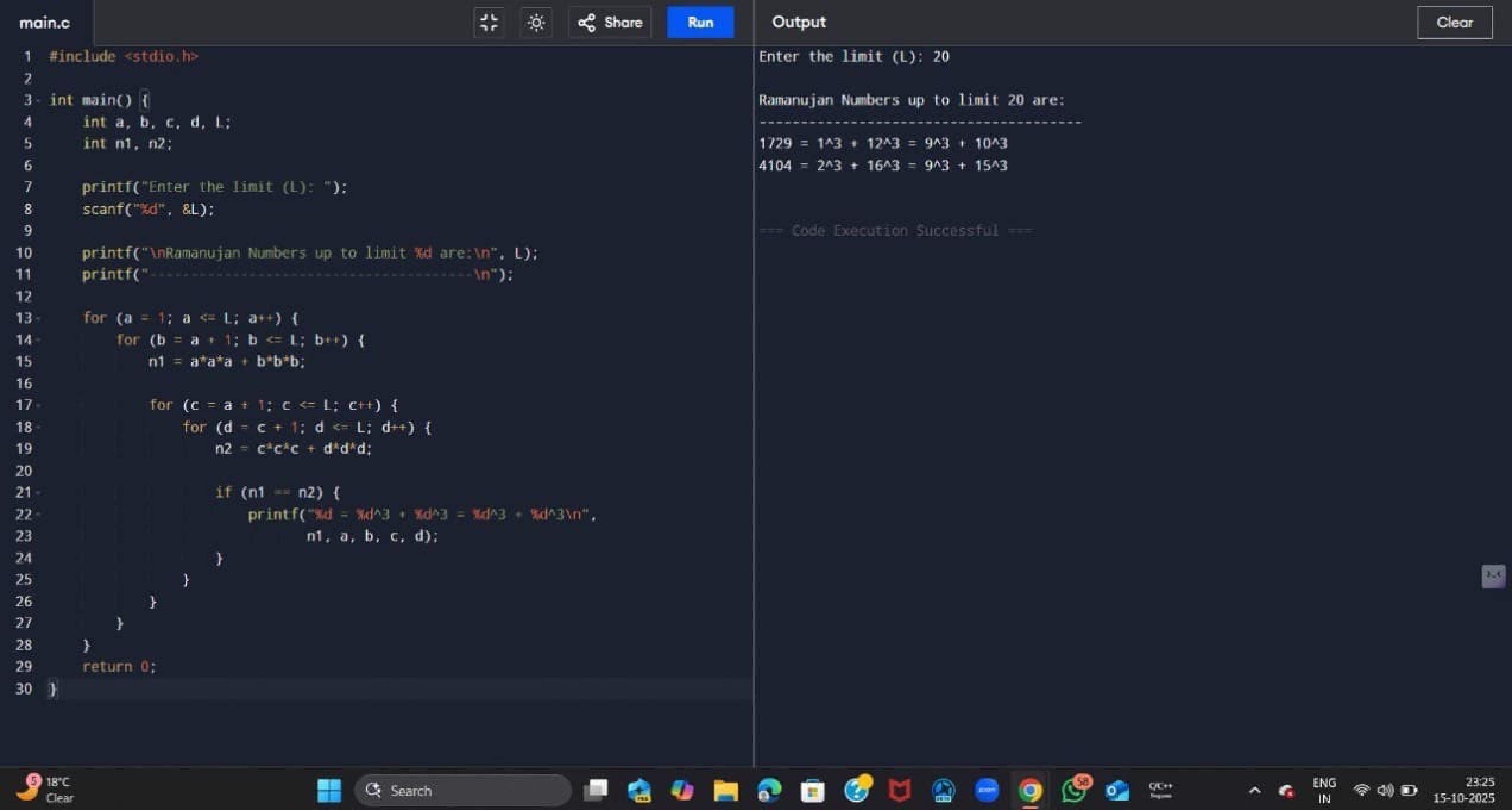
12 ^ 3 + 1 ^ 3 and 10 ^ 3 + 9 ^ 3 for a number L = 20 (that is limit).

```
# include <stdio.h>
int main ()
{
int a, b, c, d;
int limit = 10000;

printf ("Ramanujan Numbers up to %d : \n",
                                                limit );
for (a = 1; a*a * a < limit; a++) {
for ( b = a; b* b * b < limit ; b++) {
for ( c = a+1 ; c* c * c < limit ; c++) {
for ( d = c ; d * d * d < limit ; d++) {
    int sum1 = a*a* a + b* b* b;
    int sum 2 = c* c* c + d* d * d;
   if (sum 1 == sum 2 && sum1 <= limit) {
    printf ("%d = %d^3 + %d^3 = %d^3 + %d^3\n",
                          sum1, a, b, c, d);
            }
         }
```

```
                }
            }
    }

return 0;
}
```

Output :-

```
Ramanujan Numbers up to 10000:
1729 = 1^3 + 12^3 = 9^3 + 10^3
4104 = 2^3 + 16^3 = 9^3 + 15^3
```



```c
#include <stdio.h>

int main() {
    int a, b, c, d, L;
    int n1, n2;

    printf("Enter the limit (L): ");
    scanf("%d", &L);

    printf("\nRamanujan Numbers up to limit %d are:\n", L);
    printf("---------------------------------------\n");

    for (a = 1; a <= L; a++) {
        for (b = a + 1; b <= L; b++) {
            n1 = a*a*a + b*b*b;

            for (c = a + 1; c <= L; c++) {
                for (d = c + 1; d <= L; d++) {
                    n2 = c*c*c + d*d*d;

                    if (n1 == n2) {
                        printf("%d = %d^3 + %d^3 = %d^3 + %d^3\n",
                               n1, a, b, c, d);
                    }
                }
            }
        }
    }
    return 0;
}
```

```
Enter the limit (L): 20

Ramanujan Numbers up to limit 20 are:
---------------------------------------
1729 = 1^3 + 12^3 = 9^3 + 10^3
4104 = 2^3 + 16^3 = 9^3 + 15^3


=== Code Execution Successful ===
```