Here we will implement some procedures and programs to print different patterns or shapes in C program. All programs will be implemented using nested for loop.

## Contents

## Square Pattern

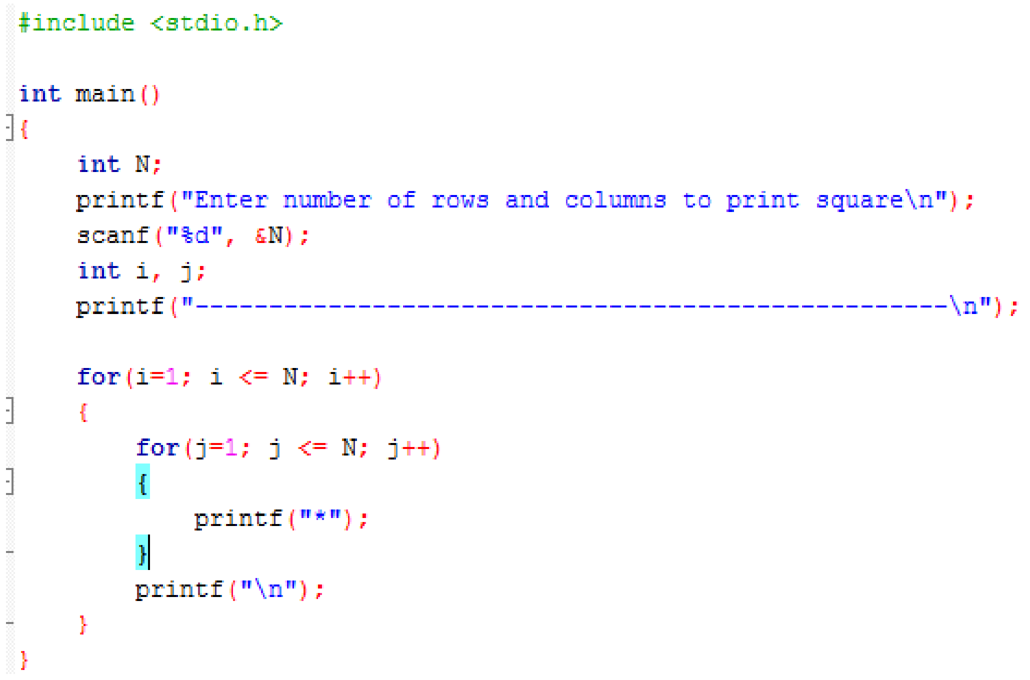
```
******
******
******
```

The pattern looks like the above star square section. Here the pattern is a matrix of n rows and columns.

Steps or algorithms to follow this program:

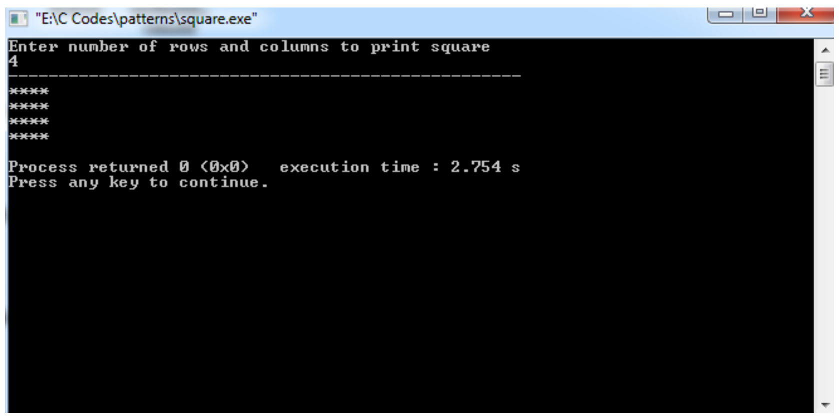
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run an outer loop from 1 to N. loop structure *for(i=1; i <= N; i++)*
3. To iterate through columns run an inner loop from 1 to N. loop structure *for(j=1; j <=N; j++)*
4. After printing a row, print a new line. go to step 2

**Program**:



```c
#include <stdio.h>

int main()
{
    int N;
    printf("Enter number of rows and columns to print square\n");
    scanf("%d", &N);
    int i, j;
    printf("---------------------------------------------------\n");

    for(i=1; i <= N; i++)
    {
        for(j=1; j <= N; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

**Output**



"E:\C Codes\patterns\square.exe"

```
Enter number of rows and columns to print square
4
----------------------------------------------------
****
****
****
****

Process returned 0 (0x0)   execution time : 2.754 s
Press any key to continue.
```

## Right Triangle Pattern

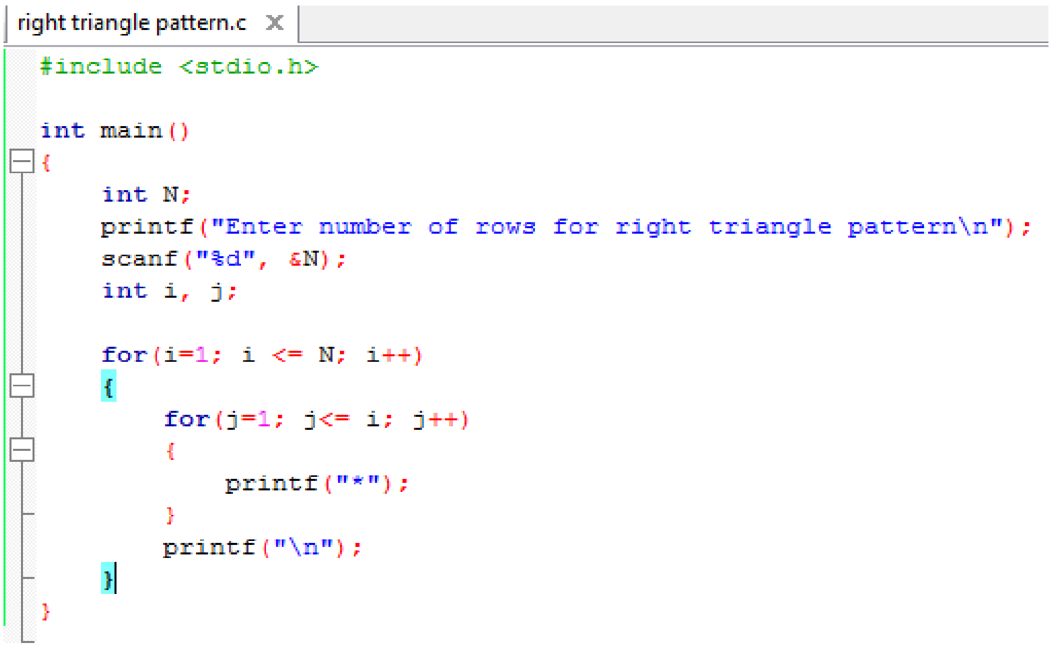
```
*
**
***
****
*****
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces.

Steps or algorithms to follow this program:

1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run an outer loop from 1 to N.
3. To iterate through columns run an inner loop from 1 to i.
4. After printing all columns with stars, print a new line. go to step 2

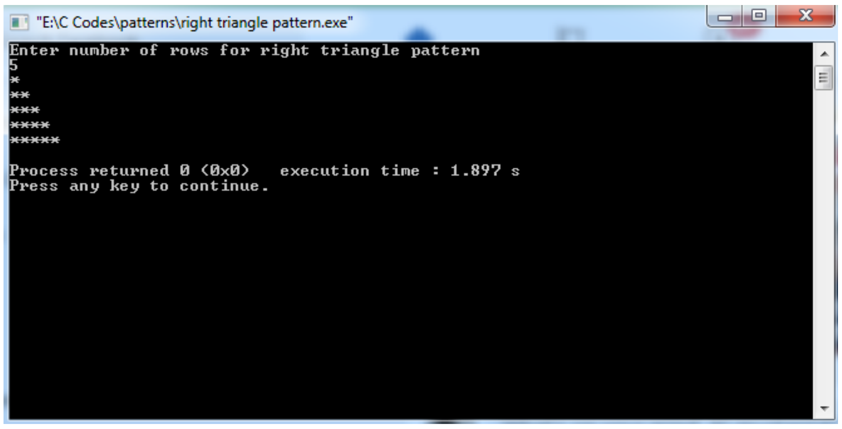
### Program



right triangle pattern.c

```c
#include <stdio.h>

int main()
{
    int N;
    printf("Enter number of rows for right triangle pattern\n");
    scanf("%d", &N);
    int i, j;

    for(i=1; i <= N; i++)
    {
        for(j=1; j<= i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

### Output



"E:\C Codes\patterns\right triangle pattern.exe"

```
Enter number of rows for right triangle pattern
5
*
**
***
****
*****

Process returned 0 (0x0)   execution time : 1.897 s
Press any key to continue.
```

## Reverse right triangle

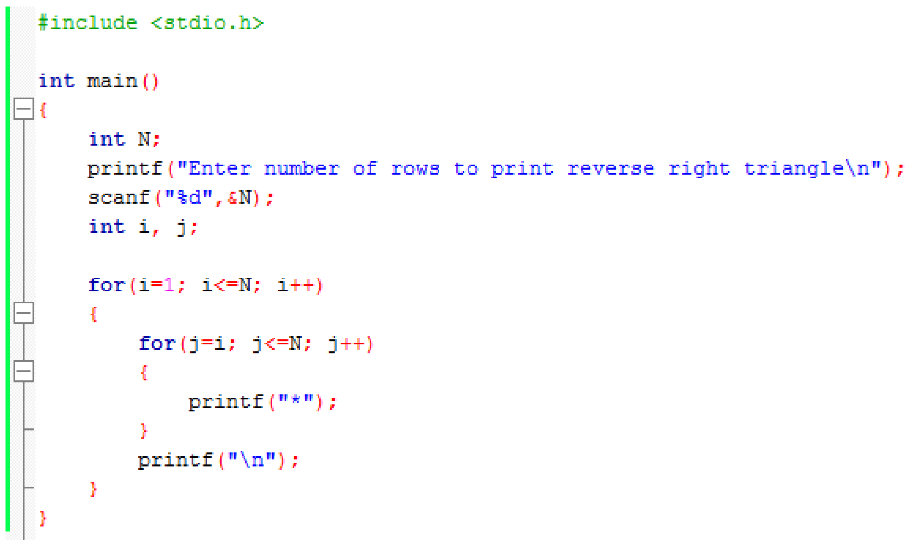
```
*****
****
***
**
*
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces.

Steps or algorithms to follow this program:

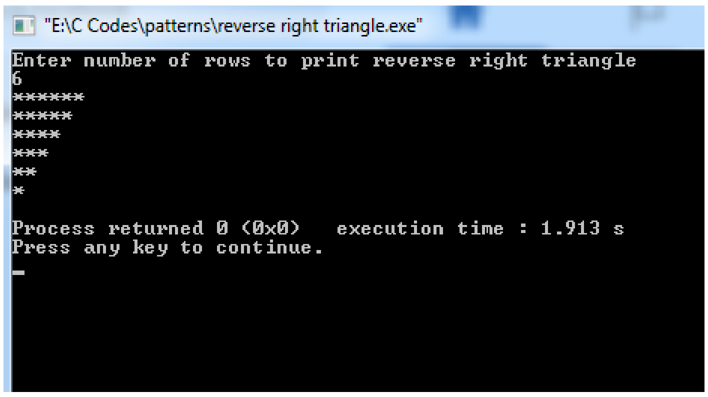
1. Input number of rows or read number of rows from user. Store it in a variable N.
2. To iterate rows run an outer loop from 1 to N.
3. To iterate columns run an inner loop from i to N.
4. After printing columns print a new line, go to step 2.

### Program



```c
#include <stdio.h>

int main()
{
    int N;
    printf("Enter number of rows to print reverse right triangle\n");
    scanf("%d",&N);
    int i, j;

    for(i=1; i<=N; i++)
    {
        for(j=i; j<=N; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

### Output



"E:\C Codes\patterns\reverse right triangle.exe"

```
Enter number of rows to print reverse right triangle
6
******
*****
****
***
**
*

Process returned 0 (0x0)   execution time : 1.913 s
Press any key to continue.
```

## Mirror Right Triangle pattern

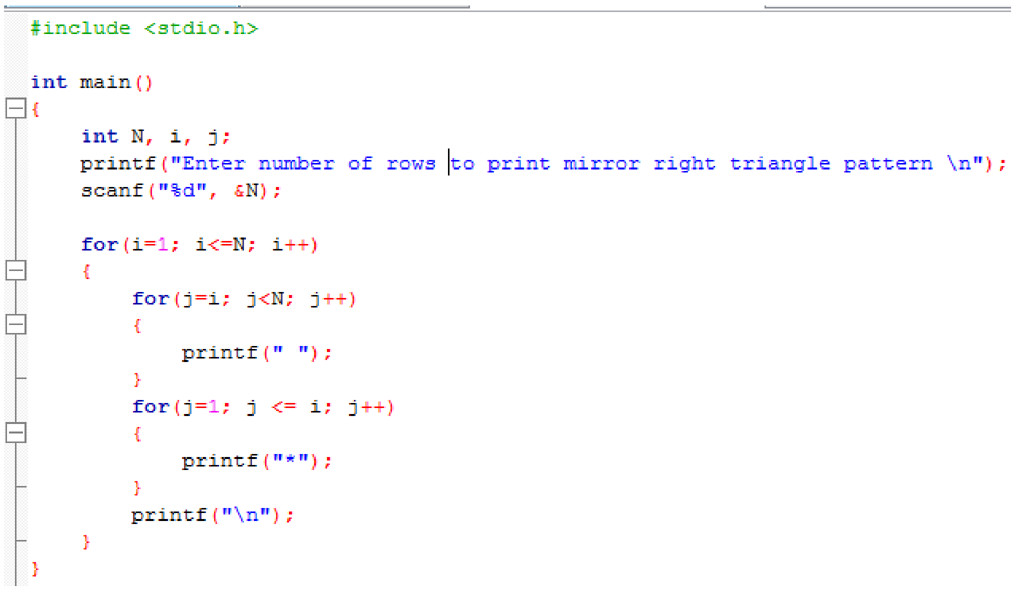
```
    *
   **
  ***
 ****
*****
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces.

Steps or algorithms to follow this program:

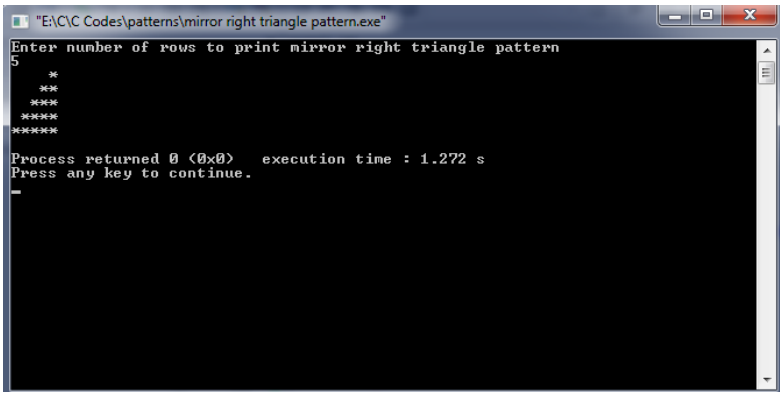
1. Input number of rows or read number of rows from user. Store it in a variable N.
2. To make spaces run an outer loop from i to N
3. To print square run another nested loop from 1 to N
4. After print Square, print a new line. Go to step 2

## Program



```c
#include <stdio.h>

int main()
{
    int N, i, j;
    printf("Enter number of rows to print mirror right triangle pattern \n");
    scanf("%d", &N);

    for(i=1; i<=N; i++)
    {
        for(j=i; j<N; j++)
        {
            printf(" ");
        }
        for(j=1; j <= i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

## Output



"E:\C\C Codes\patterns\mirror right triangle pattern.exe"

```
Enter number of rows to print mirror right triangle pattern
5
    *
   **
  ***
 ****
*****

Process returned 0 (0x0)   execution time : 1.272 s
Press any key to continue.
```

Arrange and Designed by Al-Mustanjid

## Mirror Reversed Right Triangle pattern

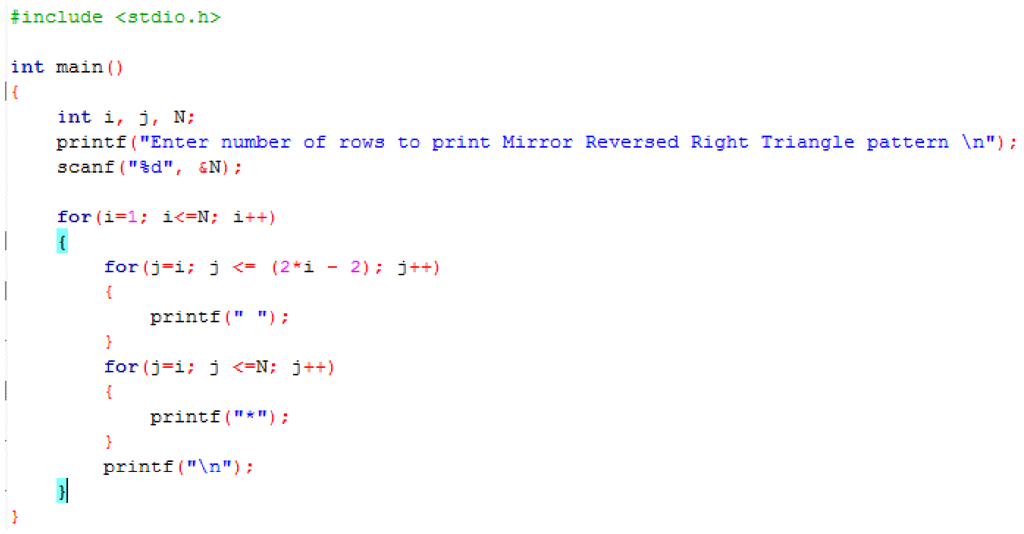
```
*****
 ****
  ***
   **
    *
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces.

Steps or algorithms to follow this program:

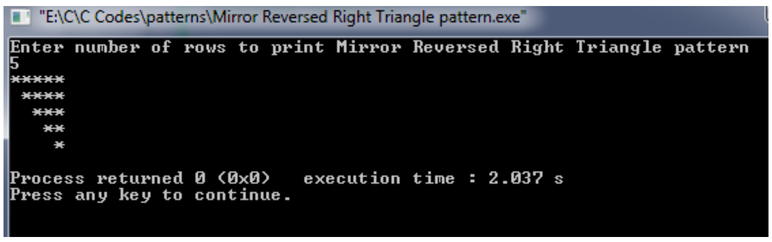
1. Input number of rows or read number of rows from user. Store it in a variable N.
2. To make spaces run an outer loop from i to *2*I – 2, space increasing gradually*
3. To print square run another nested loop from i to N
4. After print Square, print a new line. Go to step 2

## Program



```c
#include <stdio.h>

int main()
{
    int i, j, N;
    printf("Enter number of rows to print Mirror Reversed Right Triangle pattern \n");
    scanf("%d", &N);

    for(i=1; i<=N; i++)
    {
        for(j=i; j <= (2*i - 2); j++)
        {
            printf(" ");
        }
        for(j=i; j <=N; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

## Output



```
"E:\C\C Codes\patterns\Mirror Reversed Right Triangle pattern.exe"
Enter number of rows to print Mirror Reversed Right Triangle pattern
5
*****
 ****
  ***
   **
    *

Process returned 0 (0x0)   execution time : 2.037 s
Press any key to continue.
```

## Pyramid Pattern (Equilateral Triangle)

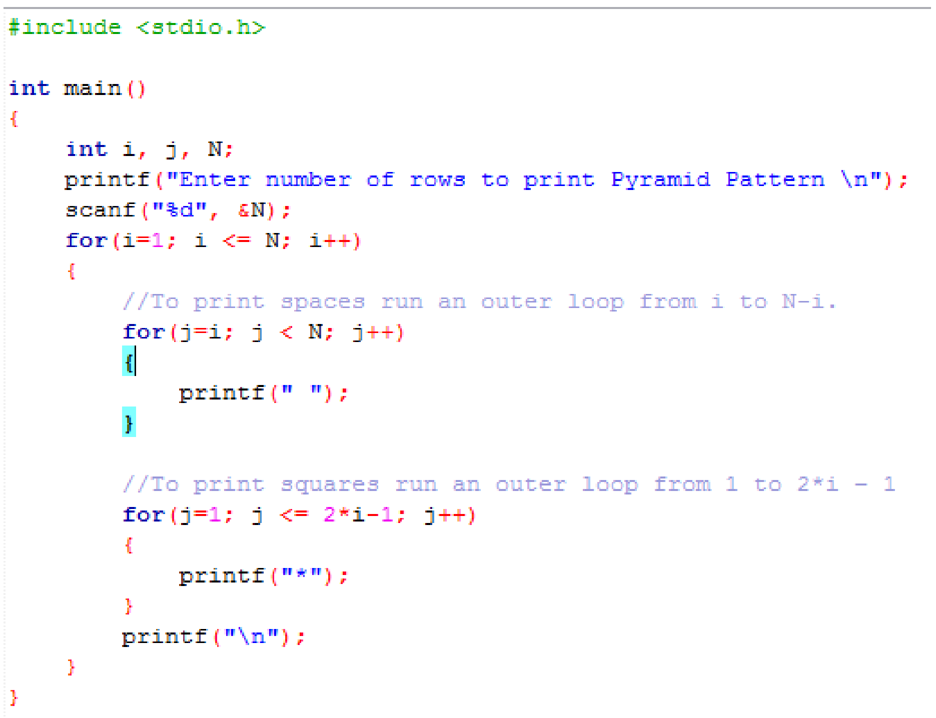
```
    *
   ***
  *****
 *******
*********
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces.

Steps or algorithms to follow this program:

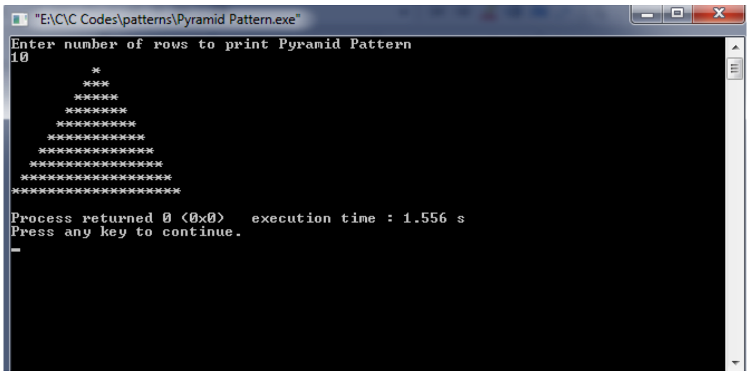
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run an outer loop from 1 to N.
3. To print spaces run an outer loop from i to N.
4. To print squares run an outer loop from 1 to 2*i – 1
5. After print square, print a new line. Go to step 2.

### Program



```c
#include <stdio.h>

int main()
{
    int i, j, N;
    printf("Enter number of rows to print Pyramid Pattern \n");
    scanf("%d", &N);
    for(i=1; i <= N; i++)
    {
        //To print spaces run an outer loop from i to N-i.
        for(j=i; j < N; j++)
        {
            printf(" ");
        }

        //To print squares run an outer loop from 1 to 2*i - 1
        for(j=1; j <= 2*i-1; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

### Output



```
"E:\C\C Codes\patterns\Pyramid Pattern.exe"
Enter number of rows to print Pyramid Pattern
10
         *
        ***
       *****
      *******
     *********
    ***********
   *************
  ***************
 *****************
*******************

Process returned 0 (0x0)   execution time : 1.556 s
Press any key to continue.
```

## Inverted Pyramid Pattern

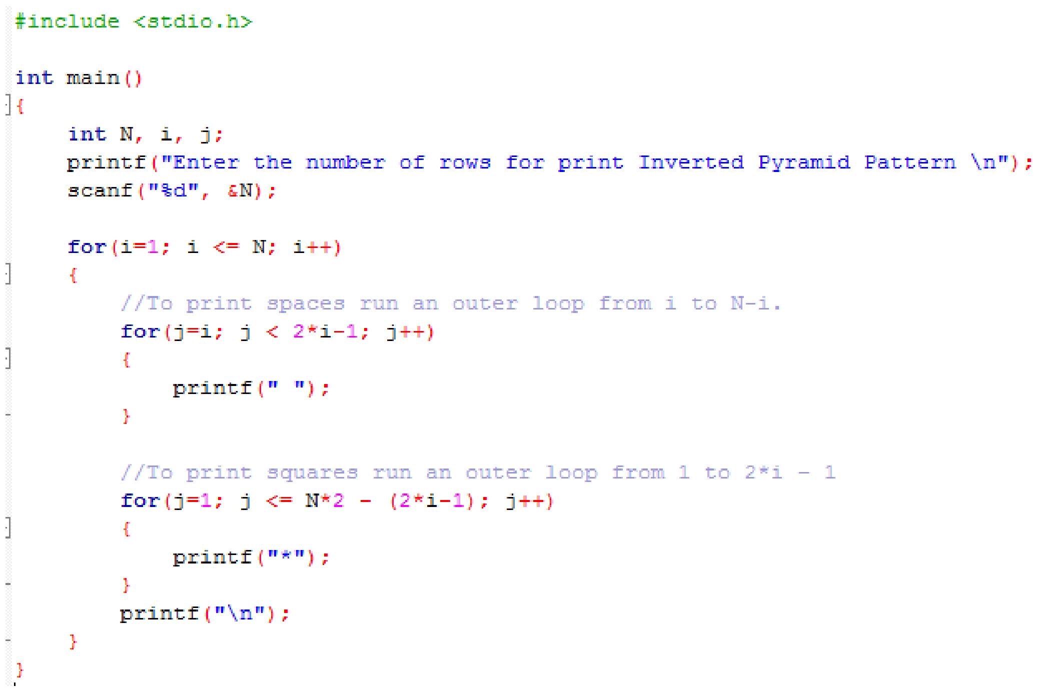
```
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

The pattern looks like the above star section. Here the pattern is a matrix of n rows and columns with spaces. Here number of stars is N*2 – (2*i-1) and space is 2*i-1 for each column.

Steps or algorithms to follow this program:

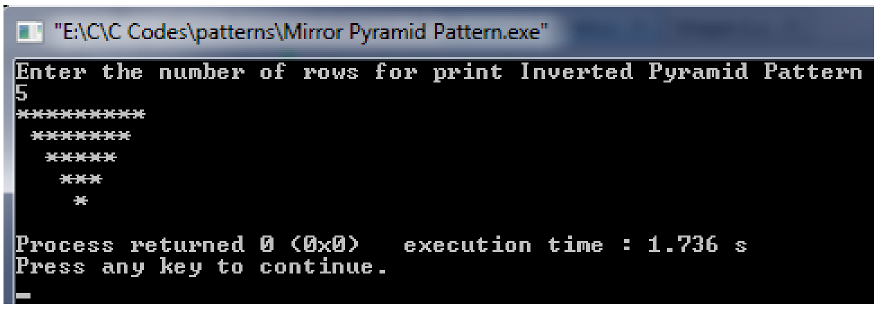
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run an outer loop from 1 to N.
3. To print spaces run an outer loop from 1 to 2*i-1.
4. To print squares run an outer loop from 1 to N*2 – (2*i-1)
5. After print square, print a new line. Go to step 2.

## Program



```c
#include <stdio.h>

int main()
{
    int N, i, j;
    printf("Enter the number of rows for print Inverted Pyramid Pattern \n");
    scanf("%d", &N);

    for(i=1; i <= N; i++)
    {
        //To print spaces run an outer loop from i to N-i.
        for(j=i; j < 2*i-1; j++)
        {
            printf(" ");
        }

        //To print squares run an outer loop from 1 to 2*i - 1
        for(j=1; j <= N*2 - (2*i-1); j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

**Output**



"E:\C\C Codes\patterns\Mirror Pyramid Pattern.exe"

```
Enter the number of rows for print Inverted Pyramid Pattern
5
*********
 *******
  *****
   ***
    *

Process returned 0 (0x0)   execution time : 1.736 s
Press any key to continue.
```

## Both

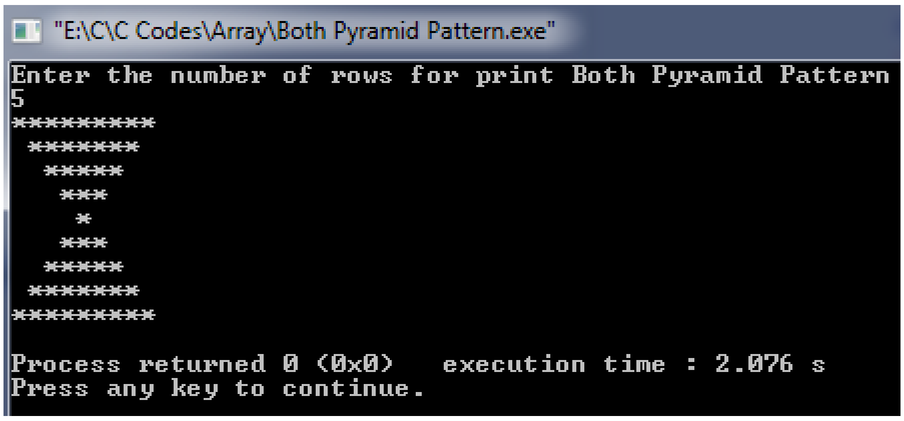
Steps or algorithms to follow this program:

1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run an outer loop from 1 to N.
3. To print spaces run an outer loop from 1 to 2*i-1.
4. To print squares run an outer loop from 1 to N*2 – (2*i-1)
5. After print square, print a new line. Go to step 2.
6. To iterate through another part of rows Run an outer loop from 2 to N.
7. To print spaces run an outer loop from 1 to N
8. To print squares run an outer loop from 1 to 2*i-1.
9. After print square, print a new line. Go to step 6.

**Program**

```c
int main()
{
    int N, i, j;
    printf("Enter the number of rows for print Both Pyramid Pattern \n");
    scanf("%d", &N);

    for(i=1; i <= N; i++)
    {
        //To print spaces run an outer loop from i to N-i.
        for(j=i; j < 2*i-1; j++)
        {
            printf(" ");
        }

        //To print squares run an outer loop from 1 to 2*i - 1
        for(j=1; j <= N*2 - (2*i-1); j++)
        {
            printf("*");
        }
        printf("\n");
    }

    for(i=2; i <= N; i++)
    {
        //To print spaces run an outer loop from i to N-i.
        for(j=i; j < N; j++)
        {
            printf(" ");
        }

        //To print squares run an outer loop from 1 to 2*i - 1
        for(j=1; j <= 2*i-1; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

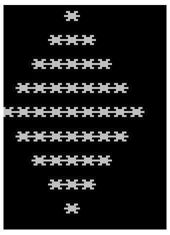
## Output



"E:\C\C Codes\Array\Both Pyramid Pattern.exe"

```
Enter the number of rows for print Both Pyramid Pattern
5
*********
 ********
  ******
   ***
    *
   ***
  ******
 ********
*********

Process returned 0 (0x0)   execution time : 2.076 s
Press any key to continue.
```

## Diamond Star Pattern



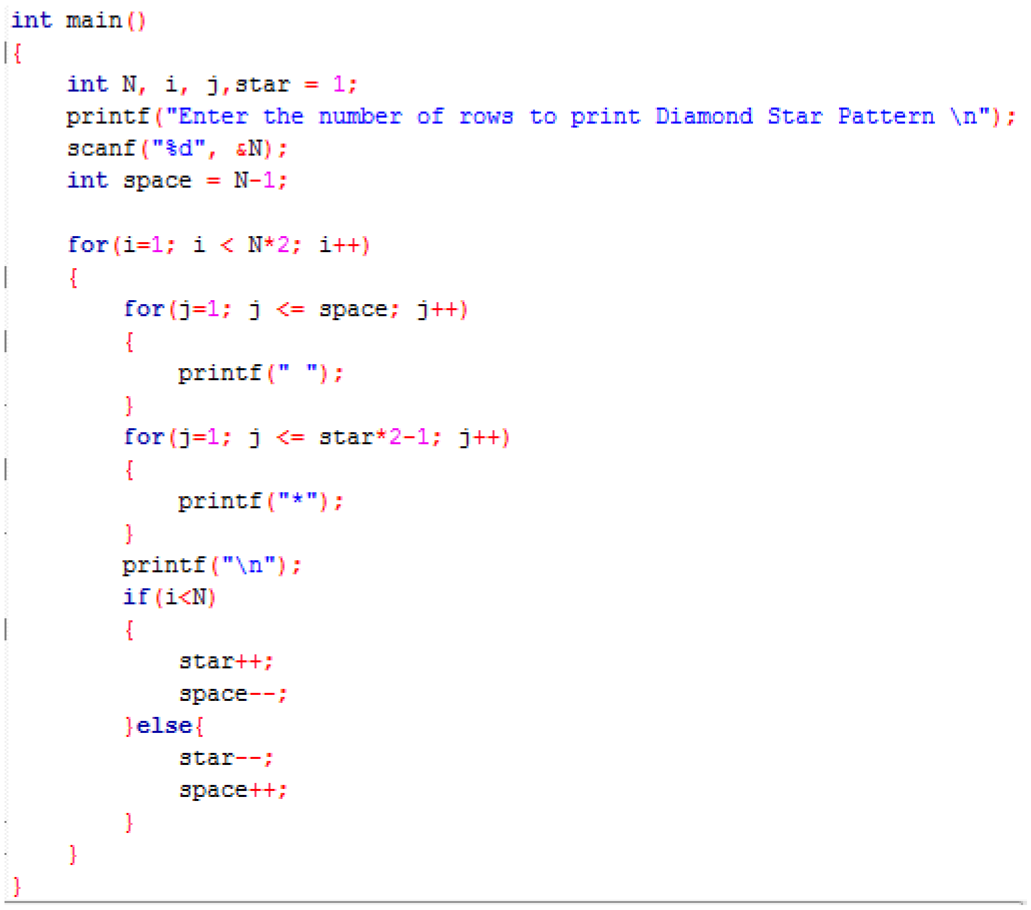
```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. Here row will is 2 * N. Space and columns are both in increasing and decreasing order and total space is N-1

Steps or algorithms to follow this program:

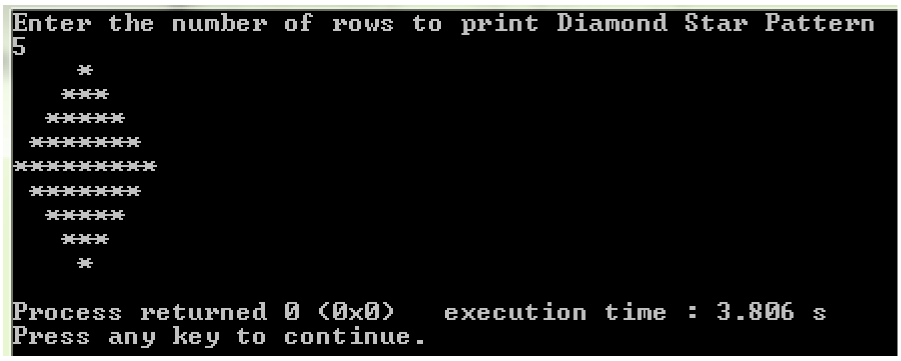
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. Declare a variable for space count, space = N-1
3. Declare another variable for star count, star = 1
4. To iterate through rows Run a loop from 1 to N*2.
5. To create space run a nested loop from 1 to space
6. To print star column wise run another nested loop from 1 to star*2-1
7. After print new line, check if (i < N) then increase star and decrease space otherwise decrease star and increase space.

**Program**



```c
int main()
{
    int N, i, j,star = 1;
    printf("Enter the number of rows to print Diamond Star Pattern \n");
    scanf("%d", &N);
    int space = N-1;

    for(i=1; i < N*2; i++)
    {
        for(j=1; j <= space; j++)
        {
            printf(" ");
        }
        for(j=1; j <= star*2-1; j++)
        {
            printf("*");
        }
        printf("\n");
        if(i<N)
        {
            star++;
            space--;
        }else{
            star--;
            space++;
        }
    }
}
```

**Output**



```
Enter the number of rows to print Diamond Star Pattern
5
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *

Process returned 0 (0x0)   execution time : 3.806 s
Press any key to continue.
```

## Half Diamond Pattern

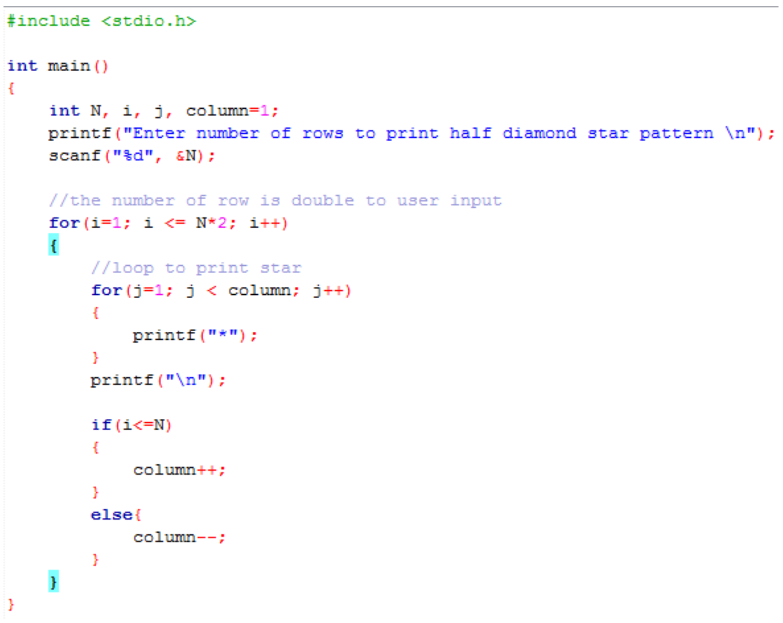
```
*
**
***
****
*****
****
***
**
*
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. Here row will be 2 * N -1. Columns with star first in increasing order then again decreasing order.

Steps or algorithms to follow this program:

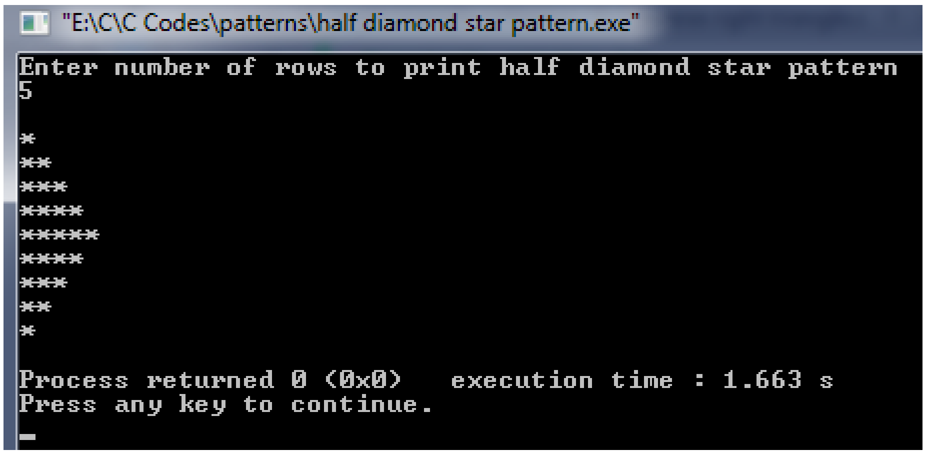
8. Input number of rows or read the number of rows from the user. Store it in a variable N.
9. Declare a variable for loop counter, column = 1
10. To iterate through rows Run an outer loop from 1 to N*2.
11. Run an outer loop from 1 to column, Print Square inside this loop.
12. After print new line, check if (i<=N) then increase column otherwise decrease column.

**Program**



```c
#include <stdio.h>

int main()
{
    int N, i, j, column=1;
    printf("Enter number of rows to print half diamond star pattern \n");
    scanf("%d", &N);

    //the number of row is double to user input
    for(i=1; i <= N*2; i++)
    {
        //loop to print star
        for(j=1; j < column; j++)
        {
            printf("*");
        }
        printf("\n");

        if(i<=N)
        {
            column++;
        }
        else{
            column--;
        }
    }
}
```

**Output**



```
Enter number of rows to print half diamond star pattern
5

*
**
***
****
*****
****
***
**
*

Process returned 0 (0x0)   execution time : 1.663 s
Press any key to continue.
```

### Mirrored Half Diamond Star Pattern

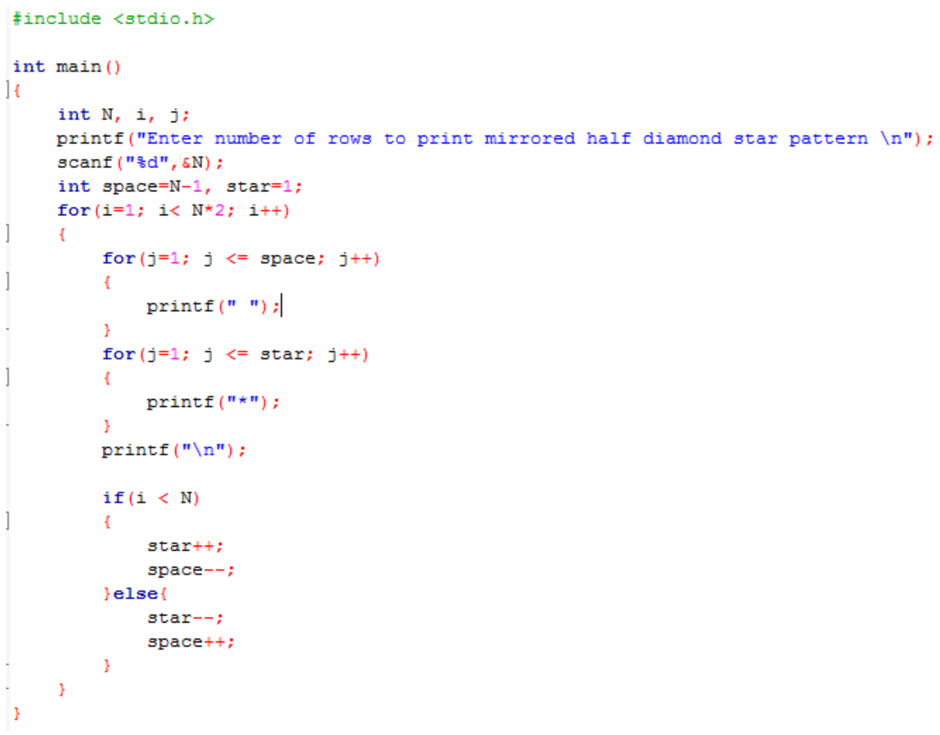
```
    *
   **
  ***
 ****
*****
 ****
  ***
   **
    *
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. Here row will be 2 * N -1. Columns with star first in increasing order then again decreasing order. But there are spaces which we need to count, space number is N-1 in both increasing and decreasing order.

Steps or algorithms to follow this program:

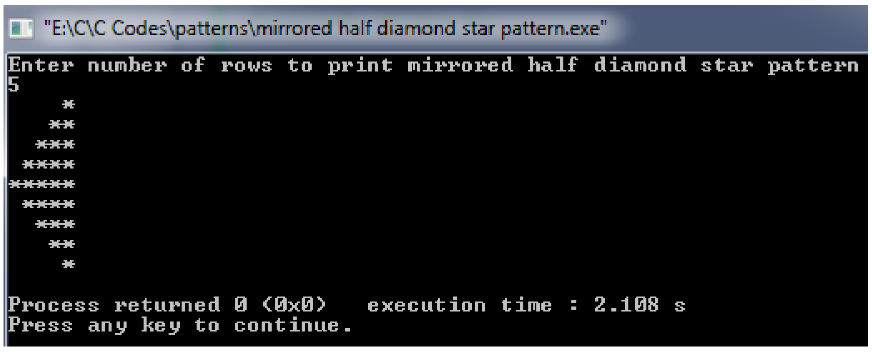
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. Declare two variables for space and star counter, space=N-1 and star=1
3. To iterate through rows Run an outer loop from 1 to N*2.
4. Run an outer loop for print space from 1 to space
5. Run another nested outer loop from 1 to star for print star
6. After print new line, check if (i < N) then increase star and decrease space otherwise decrease star and increase space.

## Program



```
#include <stdio.h>

int main()
{
    int N, i, j;
    printf("Enter number of rows to print mirrored half diamond star pattern \n");
    scanf("%d",&N);
    int space=N-1, star=1;
    for(i=1; i< N*2; i++)
    {
        for(j=1; j <= space; j++)
        {
            printf(" ");
        }
        for(j=1; j <= star; j++)
        {
            printf("*");
        }
        printf("\n");

        if(i < N)
        {
            star++;
            space--;
        }else{
            star--;
            space++;
        }
    }
}
```

## Output



```
"E:\C\C Codes\patterns\mirrored half diamond star pattern.exe"
Enter number of rows to print mirrored half diamond star pattern
5
    *
   **
  ***
 ****
*****
 ****
  ***
   **
    *

Process returned 0 (0x0)   execution time : 2.108 s
Press any key to continue.
```

## K Star Pattern

```
ccccccc
cccccc
ccccc
cccc
ccc
cc
c
cc
ccc
cccc
ccccc
cccccc
ccccccc
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. The whole pattern is divided into two parts, space in upper part decreases whereas bottom part increases.

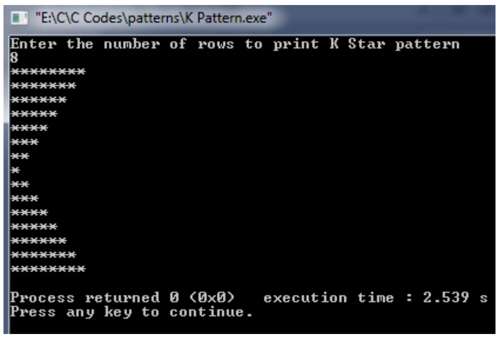
Steps or algorithms to follow this program:

1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run a loop from 1 to N.
3. To print star run a loop from i to N
4. To create spaces run another loop from 1 to N
5. Print a new line, go to step 2
6. To iterate through rows Run a loop from 1 to N.
7. To print star run a loop from 1 to i
8. Print a new line, go to step 6

### Program

```
int main()
{
    int N, i, j;
    printf("Enter the number of rows to print K Star pattern \n");
    scanf("%d", &N);

    for(i=1; i < N; i++)
    {
        for(j=i; j <= N; j++)
        {
            printf("*");
        }
        for(j=1; j < N; j++)
        {
            printf(" ");
        }
        printf("\n");
    }

    for(i=1; i <= N; i++)
    {
        for(j=1; j <= i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

### Output



"E:\C\C Codes\patterns\K Pattern.exe"

```
Enter the number of rows to print K Star pattern
8
********
*******
******
*****
****
***
**
*
**
***
****
*****
******
*******
********

Process returned 0 (0x0)   execution time : 2.539 s
Press any key to continue.
```

**Mirror K Star Pattern**

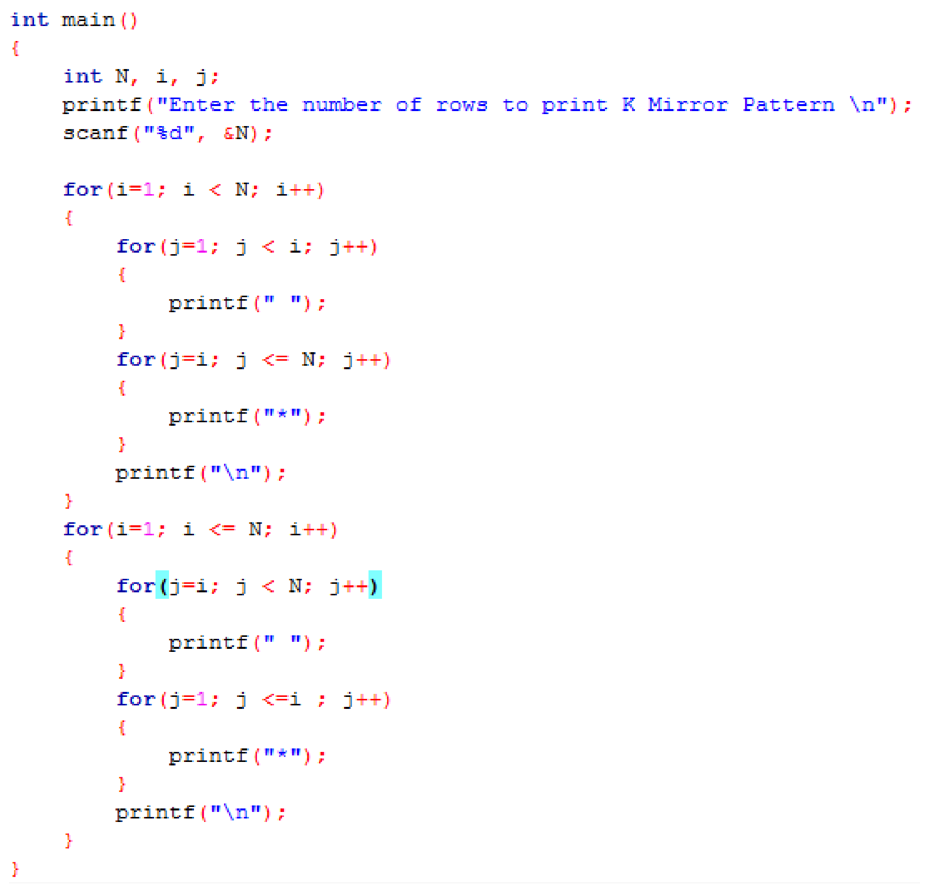
```
ccccccc
 cccccc
  ccccc
   cccc
    ccc
     cc
      c
     cc
    ccc
   cccc
  ccccc
 cccccc
ccccccc
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. The whole pattern is divided into two parts, space in upper part decreases whereas bottom part increases.

Steps or algorithms to follow this program:

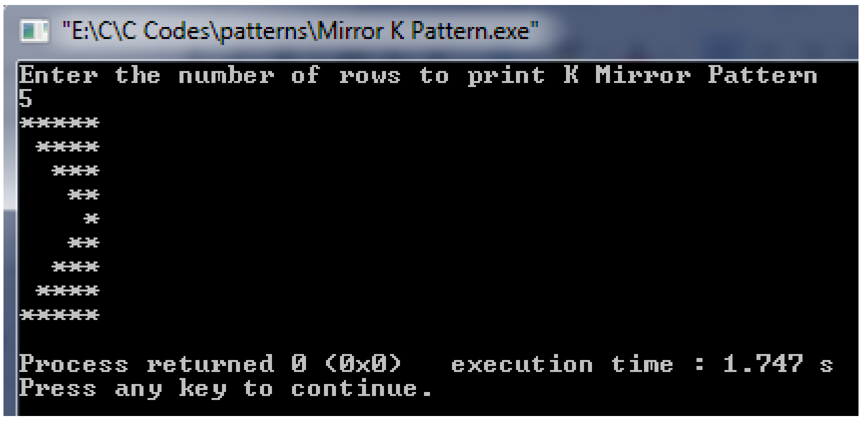
1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. To iterate through rows Run a loop from 1 to N.
3. To create spaces run another loop from 1 to i
4. To print star run a loop from i to N
5. Print a new line, go to step 2
6. To iterate through rows Run a loop from 1 to N.
7. To create spaces run another loop from i to N
8. To print star run a loop from 1 to i
9. Print a new line, go to step 6

**Program**



```c
int main()
{
    int N, i, j;
    printf("Enter the number of rows to print K Mirror Pattern \n");
    scanf("%d", &N);

    for(i=1; i < N; i++)
    {
        for(j=1; j < i; j++)
        {
            printf(" ");
        }
        for(j=i; j <= N; j++)
        {
            printf("*");
        }
        printf("\n");
    }
    for(i=1; i <= N; i++)
    {
        for(j=i; j < N; j++)
        {
            printf(" ");
        }
        for(j=1; j <=i ; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

**Output**



```
"E:\C\C Codes\patterns\Mirror K Pattern.exe"
Enter the number of rows to print K Mirror Pattern
5
*****
 ****
  ***
   **
    *
   **
  ***
 ****
*****

Process returned 0 (0x0)   execution time : 1.747 s
Press any key to continue.
```

## X Star Pattern

```
*       *
 *     *
  *   *
   * *
    *
   * *
  *   *
 *     *
*       *
```

The pattern looks like the above figure. Here the pattern is a matrix of n rows and columns with spaces. Here the row count is N*2-1. Notice that if row and columns are equal then print star otherwise print space.

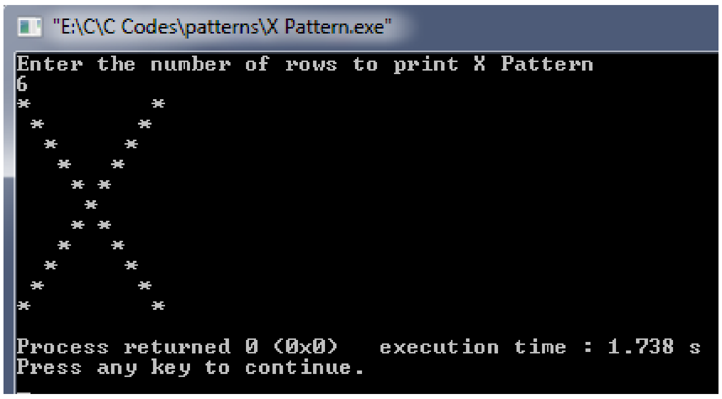
Steps or algorithms to follow this program:

1. Input number of rows or read the number of rows from the user. Store it in a variable N.
2. Declare a variable for row counter, count = N*2-1
3. To iterate through rows Run a loop from 1 to count
4. Run another nested loop from 1 to count
5. Check if rows and column are equal (i==j) for first diagonal and (j== count – i + 1) then print star, otherwise print space
6. Print a new line

**Program**

```c
#include <stdio.h>

int main()
{
    int N, i, j, count;
    printf("Enter the number of rows to print X Pattern \n");
    scanf("%d", &N);
    count = N*2-1;
    for(i=1; i <= count; i++)
    {
        for(j=1; j <= count; j++)
        {
            if((j==i) || j == (count-i+1))
            {
                printf("*");
            }else{
                printf(" ");
            }
        }
        printf("\n");
    }
}
```

**Output**



"E:\C\C Codes\patterns\X Pattern.exe"

```
Enter the number of rows to print X Pattern
6
*         *
 *       *
  *     *
   *   *
    * *
     *
    * *
   *   *
  *     *
 *       *
*         *

Process returned 0 (0x0)   execution time : 1.738 s
Press any key to continue.
```