#include <stdio.h>

int main() {

int rows = 5; // Number of rows and columns in the square

// Loop to iterate over rows

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

// Loop to iterate over columns

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

printf("\*");

}

printf("\n"); // Move to the next line after printing each row

}

return 0;

}

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print asterisk

for (int j = 1; j <= i; j++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print numbers from 1 to i

for (int j = 1; j <= i; j++) {

printf("%d", j);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

1

12

123

1234

12345

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print the number i, 'i' times

for (int j = 1; j <= i; j++) {

printf("%d", i);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

1

22

333

4444

55555

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

int spaces = 0; // Number of spaces to print at the beginning of each row

// Outer loop to iterate over rows

for (int i = rows; i >= 1; i--) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increment spaces for the next row

}

return 0;

}

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

int spaces = 4; // Number of spaces to print at the beginning of each row

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces--; // Decrease spaces for the next row

}

return 0;

}

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in each pattern

int spaces = 0; // Number of spaces to print at the beginning of each row

// First pattern: increasing asterisks, decreasing spaces

for (int i = 1; i <= rows; i++) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

spaces = rows - 1; // Reset spaces for the second pattern

// Second pattern: decreasing asterisks, increasing spaces

for (int i = rows; i >= 1; i--) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces--; // Decrease spaces for the next row

}

return 0;

}

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in each pattern

int spaces = rows - 1; // Number of spaces to print at the beginning of each row for the first pattern

// Second pattern: decreasing asterisks, increasing spaces

for (int i = rows; i >= 1; i--) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

spaces = 0; // Reset spaces for the first pattern

// First pattern: increasing asterisks, decreasing spaces

for (int i = 1; i <= rows; i++) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces--; // Decrease spaces for the next row

}

return 0;

}

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in each pattern

int spaces = rows - 1; // Number of spaces to print at the beginning of each row for the first pattern

// Second pattern: decreasing asterisks, increasing spaces

for (int i = rows; i >= 1; i--) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

spaces = 1; // Reset spaces for the first pattern

// First pattern: increasing asterisks, decreasing spaces

for (int i = 1; i <= rows; i++) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

for (int j = 1; j <= 2 \* (rows - i) + 1; j++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

return 0;

}

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in each pattern

int spaces = rows - 1; // Number of spaces to print at the beginning of each row for the first pattern

// Second pattern: decreasing asterisks, increasing spaces

for (int i = rows; i >= 1; i--) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

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

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

spaces = 1; // Reset spaces for the first pattern

// First pattern: increasing asterisks, decreasing spaces

for (int i = 1; i <= rows; i++) {

// Print leading spaces

for (int s = 1; s <= spaces; s++) {

printf(" ");

}

// Print asterisks

for (int j = 1; j <= 2 \* (rows - i) + 1; j++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

spaces++; // Increase spaces for the next row

}

return 0;

}

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Print increasing asterisks

for (int i = 1; i <= rows; i++) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

// Print decreasing asterisks

for (int i = rows - 1; i >= 1; i--) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

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

// Inner loop to print numbers

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

printf("%d", (j % 2));

}

printf("\n"); // Move to the next line after each row

}

// Print 1 0 1 0 1

for (int i = 0; i < 5; i++) {

printf("%d ", i % 2);

}

return 0;

}

1

01

101

0101

1 0 1 0 1

#include <stdio.h>

int main() {

int rows = 4; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Print numbers in increasing order

for (int j = 1; j <= i; j++) {

printf("%d", j);

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print numbers in decreasing order

for (int l = i; l >= 1; l--) {

printf("%d", l);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

1 1

12 21

123 321

12344321

#include <stdio.h>

int main() {

int rows = 4; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Print numbers in increasing order

for (int j = 1; j <= i; j++) {

printf("%d", j);

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print numbers in decreasing order

for (int l = i; l >= 1; l--) {

printf("%d", l);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

1 1

12 21

123 321

12344321

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

int num = 1; // Initialize the starting number

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print numbers

for (int j = 1; j <= i; j++) {

printf("%d ", num); // Print the current number

num++; // Increment the number for the next iteration

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print characters

char ch = 'A'; // Initialize character 'A'

for (int j = 1; j <= i; j++) {

printf("%c", ch); // Print the current character

ch++; // Increment the character for the next iteration

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

A

AB

ABC

ABCD

ABCDE

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = rows; i >= 1; i--) {

// Inner loop to print characters

char ch = 'A'; // Initialize character 'A'

for (int j = 1; j <= i; j++) {

printf("%c", ch); // Print the current character

ch++; // Increment the character for the next iteration

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

ABCDE

ABCD

ABC

AB

A

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

for (int i = 1; i <= rows; i++) {

// Inner loop to print characters

char ch = 'A' + i - 1; // Calculate the character based on the row number

for (int j = 1; j <= i; j++) {

printf("%c", ch); // Print the current character

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

A

BB

CCC

DDDD

EEEEE

#include <stdio.h>

int main() {

int rows = 4; // Number of rows in the pattern

// Outer loop to iterate over rows

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

// Print leading spaces

for (int space = 0; space < rows - i - 1; space++) {

printf(" ");

}

// Print characters in ascending order

for (char ch = 'A'; ch < 'A' + i + 1; ch++) {

printf("%c", ch);

}

// Print characters in descending order

for (char ch = 'A' + i - 1; ch >= 'A'; ch--) {

printf("%c", ch);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

A

ABA

ABCBA

ABCDCBA

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Outer loop to iterate over rows

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

// Print leading spaces

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

printf(" ");

}

// Print characters in ascending order

for (char ch = 'E' - i; ch <= 'E'; ch++) {

printf("%c", ch);

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

E

DE

CDE

BCDE

ABCDE

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Upper part

for (int i = rows; i >= 1; i--) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print asterisks

for (int l = 1; l <= i; l++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

// Lower part

for (int i = 1; i <= rows; i++) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print asterisks

for (int l = 1; l <= i; l++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

\*\*\*\*\*\*\*\*\*\*

\*\*\*\* \*\*\*\*

\*\*\* \*\*\*

\*\* \*\*

\* \*

\* \*

\*\* \*\*

\*\*\* \*\*\*

\*\*\*\* \*\*\*\*

\*\*\*\*\*\*\*\*\*\*

#include <stdio.h>

int main() {

int rows = 5; // Number of rows in the pattern

// Upper part

for (int i = 1; i <= rows; i++) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print asterisks

for (int l = 1; l <= i; l++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

// Lower part

for (int i = rows; i >= 1; i--) {

// Print asterisks

for (int j = 1; j <= i; j++) {

printf("\*");

}

// Print spaces

for (int k = 1; k <= 2 \* (rows - i); k++) {

printf(" ");

}

// Print asterisks

for (int l = 1; l <= i; l++) {

printf("\*");

}

printf("\n"); // Move to the next line after each row

}

return 0;

}

\* \*

\*\* \*\*

\*\*\* \*\*\*

\*\*\*\* \*\*\*\*

\*\*\*\*\*\*\*\*\*\*

\*\*\*\* \*\*\*\*

\*\*\* \*\*\*

\*\* \*\*

\* \*

#include <stdio.h>

int main() {

int rows = 4; // Number of rows in the pattern

// Upper part

for (int i = 1; i <= rows; i++) {

// Print the first '\*'

printf("\*");

// Print the spaces or '\*' characters in between

if (i == 1 || i == rows) {

// For the first and last rows, print all '\*'

for (int j = 1; j <= rows - 2; j++) {

printf("\*");

}

} else {

// For other rows, print one '\*' and spaces in between

for (int j = 1; j <= rows - 2; j++) {

printf(" ");

}

printf("\*");

}

// Print the last '\*'

printf("\*\n");

}

return 0;

}

\*\*\*\*

\* \*

\* \*

\*\*\*\*

#include <stdio.h>

int main() {

int n = 4; // Number of rows in the pattern

int size = n \* 2 - 1; // Size of the pattern

// Upper part

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

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

int val = n - (i < n ? i : size - i - 1);

printf("%d", n - (j < val ? j : size - j - 1));

}

printf("\n");

}

return 0;

}

4444444

4333334

4322234

4321234

4322234

4333334

4444444