30+ Most Asked Pattern Programs in C, C++ and Java

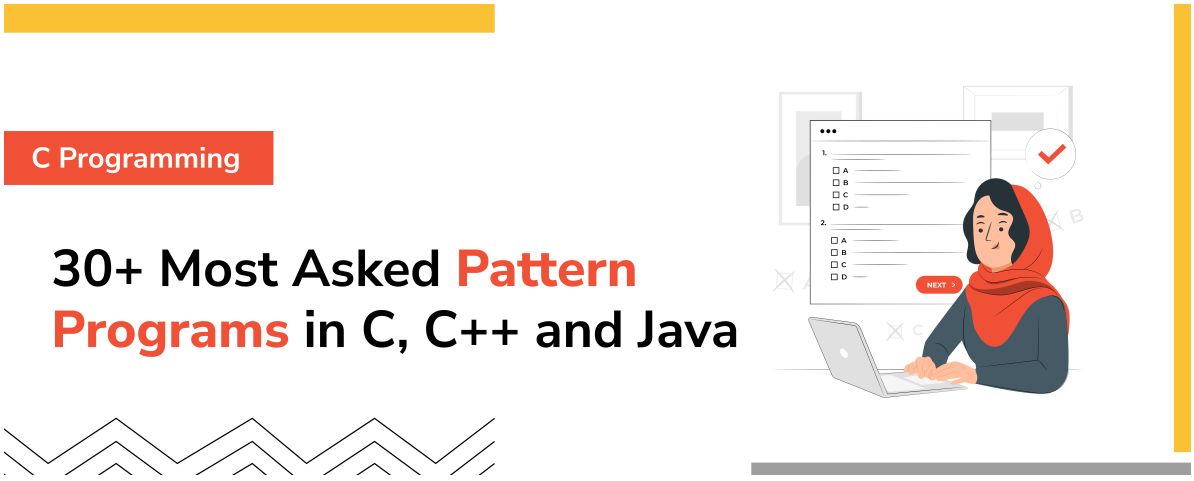
Published on Thu Mar 09 2023

C Programming | C++ Programming | Java Programming | Interview Preparation

Share

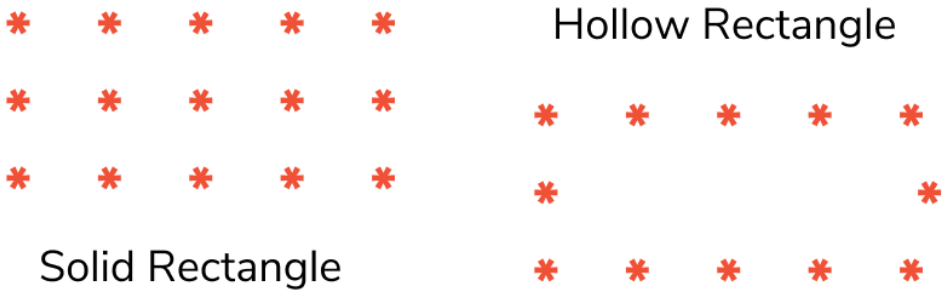
This article discusses a variety of **Pattern programs in C, C++ and Java**. **Pattern programs** are nothing but patterns consisting of numbers, alphabets or symbols in a particular form. **These kinds of pattern programs can be solved easily using loop conditions**.

Read more to see **30 of the most asked pattern programs in C, C++ and Java.**



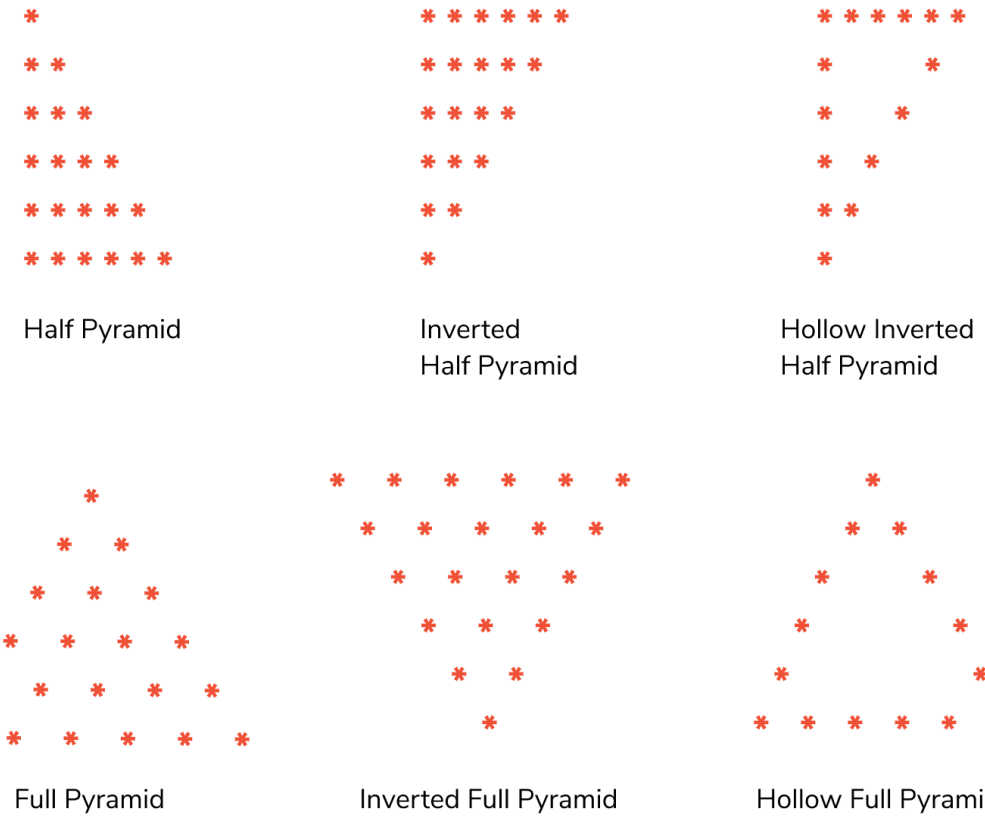
#1 - Rectangle Pattern Programs in C, C++, Java

To print a solid and hollow rectangle using stars Click here.

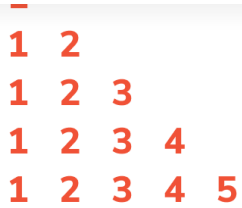


#2 - Pyramid Pattern Programs in C, C++, and Java using stars

To print the Pyramid patterns shown below Click here.

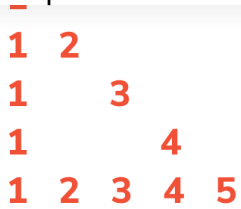


#3 - Pyramid Pattern Programs in C, C++, and Java using

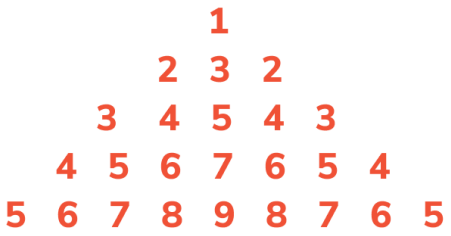


Half Pyramid

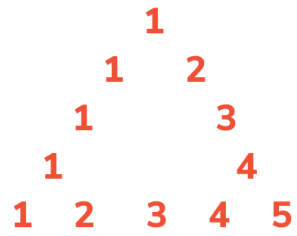
Inverted Half Pyramid



Hollow Half Pyramid



Full Pyramid

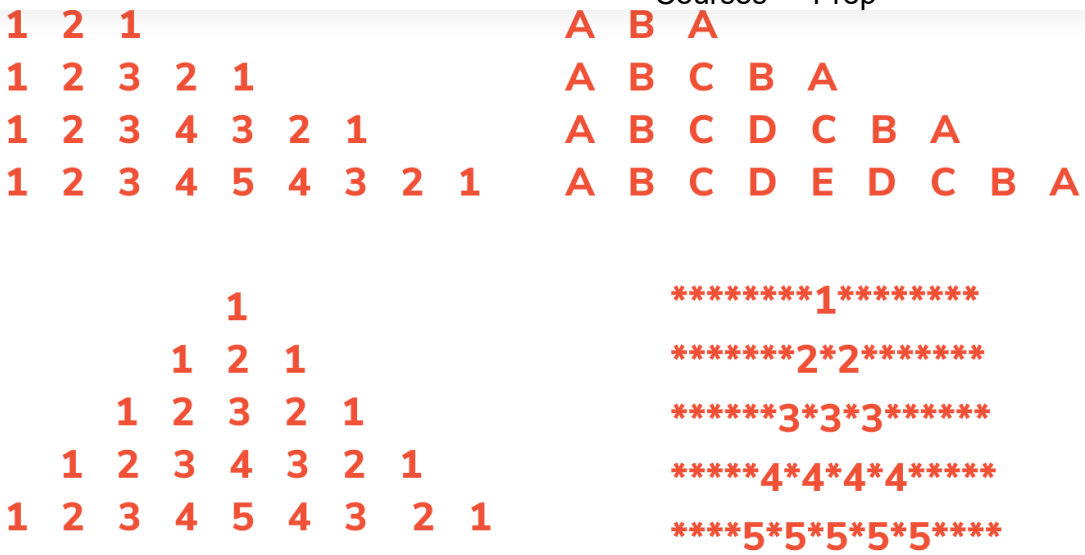


Hollow Full Pyramid

Hollow Inverted Half Pyramid

#4 - Palindrome Pyramid Pattern Programs in C, C++, and Java using numbers & Alphabets

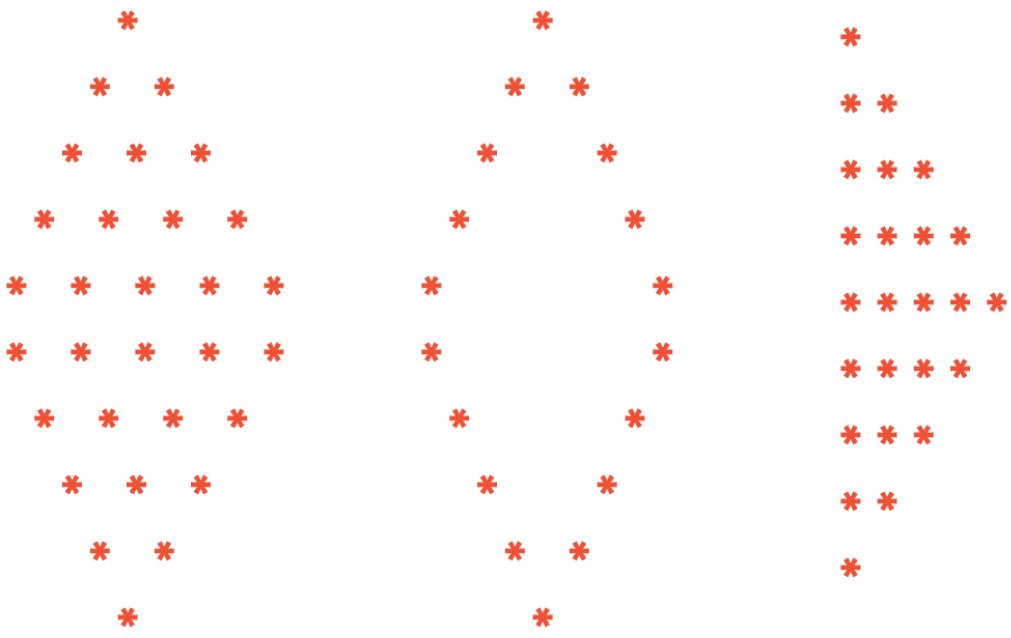
Click here to see the program to print the pattern shown below using numbers and alphabets.



Different types of Palindrome Pyramid Patterns

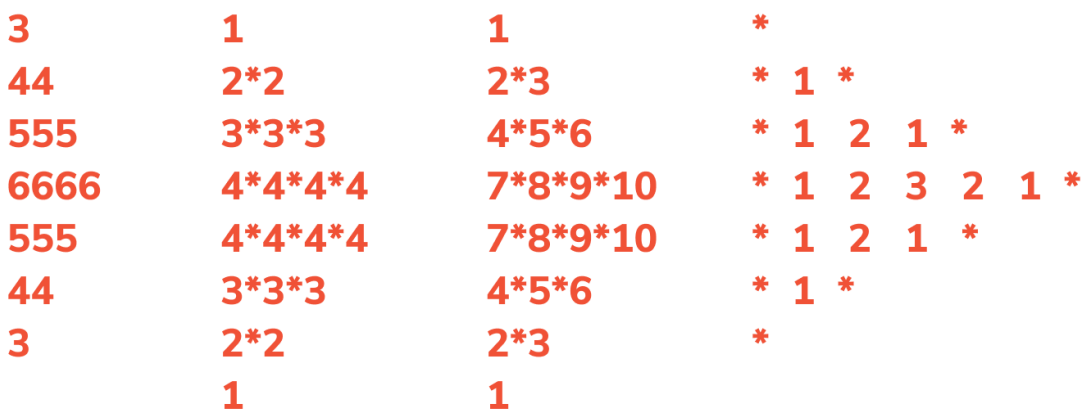
#5 - Diamond Pattern Programs in C, C++, and Java using stars

Click here to get the program to print the Diamond pattern programs using stars.



Solid Diamond | Hollow Diamond | Solid Half Diamond

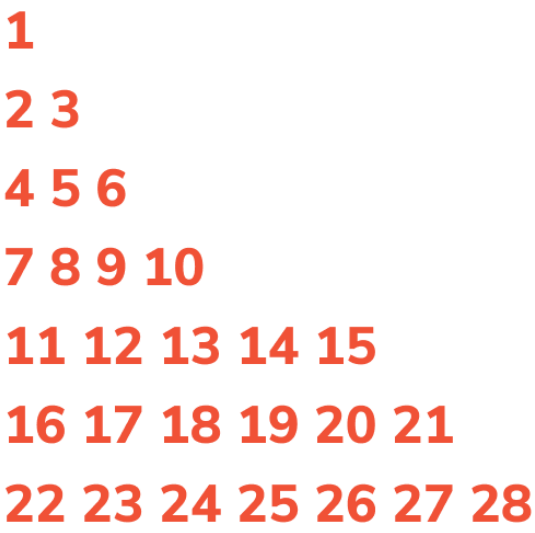
- **Click here to view the solution of the 4th problem.**



Different types of Solid Half Diamonds

#7 - Floyd's Triangle Pattern Program in C, C++, Java, Python

To know the solution to print Floyd's triangle below, click here.



#8 - Pascal's Triangle Pattern Program in C, C++, Java, Python

Click here to know the detailed solution of the below shown

row 2 = (0+1), (1+1), (1+0) = 1, 2, 1

row 3 = (0+1), (1+2), (2+1), (1+0) = 1, 3, 3, 1

row 4 = (0+1), (1+3), (3+3), (3+1), (1+0) = 1, 4, 6, 4, 1

row 5 = (0+1), (1+4), (4+6), (6+4), (4+1),(1+0) = 1, 5, 10, 10, 5, 1

row 6 = (0+1), (1+5), (5+10), (10+10), (10+5), (5+1), (1+0) = 1, 6, 15, 20, 15, 6, 1

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
```

#9 - Hollow Diamond Inscribed in a Rectangle

Input: 5

Output:

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

C

```
#include <stdio.h>
int main()
{
   int i, j, n;
   scanf("%d", &n);
   // upper half of the pattern
   for(i = 0; i < n; i++)
   {
      for(j = 0; j < (2 * n); j++)
      {
         if(i + j <= n - 1)  // upper left triangle
            printf("*");
         else
            printf(" ");
         if((i + n) <= j)  // upper right triangle
            printf("*");
         else
            printf(" ");
      }
      printf("\n");
   }
   // bottom half of the pattern
   for(i = 0; i < n; i++)
   {
      for(j = 0; j < (2 * n); j++)
      {
         if(i >= j)  //bottom left triangle
            printf("*");
         else
            printf(" ");
         if(i >= (2 * n - 1) - j)  // bottom right triangle
            printf("*");
         else
            printf(" ");
      }
      printf("\n");
   }
   return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
   int i, j, n;
   cin >> n;
```

```
        else
            cout << " ";
        if((i + n) <= j)  // upper right triangle
            cout << "*";
        else
            cout << " ";
    }
    cout << "\n";
  }
  // bottom half of the pattern
  for(i = 0; i < n; i++)
  {
    for(j = 0; j < (2 * n); j++)
    {
        if(i >= j)  // bottom left triangle
            cout << "*";
        else
            cout << " ";
        if(i >= (2 * n - 1) - j)  // bottom right triangle
            cout << "*";
        else
            cout << " ";
    }
    cout << "\n";
  }
  return 0;
}
```

Java

```
import java.util.Scanner;
public class Main{
  public static void main(String args[])
  {
    Scanner sc = new Scanner(System.in);
    int i, j;
    int n = sc.nextInt();
    // upper half of the patternfor(i = 0; i < n; i++)
    {
        for(j = 0; j < (2 * n); j++)
        {
            if(i + j <= n - 1)  // upper left triangle
                System.out.print("*");
            else
                System.out.print(" ");
            if((i + n) <= j) // upper right triangle
                System.out.print("*");
            else
                System.out.print(" ");
        }
```

```
            else
               System.out.print(" ");
            if(i >= (2 * n - 1) - j)  // bottom right triangle
               System.out.print("*");
            else
               System.out.print(" ");
         }
         System.out.println();
      }
   }
}
```

#10 - Butterfly Pattern Printing

Input: 5

Output:

Solution for Butterfly Pattern:

C

```
#include <stdio.h>
int main()
{
   int i, j, n;
   scanf("%d", &n);
   // upper half of the pattern
   for(i = 0; i < n; i++)
   {
      for(j = 0; j < (2 * n); j++)
      {
         if(i >= j)  // upper left triangle
            printf("*");
         else
            printf(" ");
```

```
    for(i = 0; i < n; i++)
    {
        for(j = 0; j < (2 * n); j++)
        {
            if(i + j <= n - 1) // bottom left triangle
                printf("*");
            else
                printf(" ");
            if((i + n) <= j) // bottom right triangle
                printf("*");
            else
                printf(" ");
        }
        printf("\n");
    }
    return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
    int i, j, n;
    cin >> n;
    // upper half of the pattern
    for(i = 0; i < n; i++)
    {
        for(j = 0; j < (2 * n); j++)
        {
            if(i >= j) // upper left triangle
                cout << "*";
            else
                cout << " ";
            if(i >= (2 * n - 1) - j)  // upper right triangle
                cout << "*";
            else
                cout << " ";
        }
        cout << "\n";
    }
    // bottom half of the pattern
    for(i = 0; i < n; i++)
    {
        for(j = 0; j < (2 * n); j++)
        {
            if(i + j <= n - 1)  // bottom left triangle
                cout << "*";
            else
```

```
    return 0;
}
```

Java

```
import java.util.Scanner;
public class Main{
   public static void main(String args[])
   {
      Scanner sc = new Scanner(System.in);
      int i, j;
      int n = sc.nextInt();
      // upper half of the patternfor(i = 0; i < n; i++)
      {
         for(j = 0; j < (2 * n); j++)
         {
            if(i >= j)  // upper left triangle
               System.out.print("*");
            else
               System.out.print(" ");
            if(i >= (2 * n - 1) - j)  // upper right triangle
               System.out.print("*");
            else
               System.out.print(" ");
         }
         System.out.println();
      }
      // bottom half of the patternfor(i = 0; i < n; i++)
      {
         for(j = 0; j < (2 * n); j++)
         {
            if(i + j <= n - 1)  // bottom left triangle
               System.out.print("*");
            else
               System.out.print(" ");
            if((i + n) <= j)  // bottom right triangle
               System.out.print("*");
            else
               System.out.print(" ");
         }
         System.out.println();
      }
   }
}
```

#11 - Diagonal & Sides of a Rectangle

Input: 7 (input should be an odd number only, else the desired output will not be obtained)

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

Solution for Printing the diagonal and sides of a rectangle:

C

```
#include <stdio.h>
int main()
{
   int i, j, n;
   scanf("%d", &n);  // 'n' must be odd
   for(i = 0; i < n; i++)
   {
      for(j = 0; j < n; j++)
      {
         // left diagonal, right diagonal, top horizontal, bottom horizontal, left vertical, right verti
         if(i == j || i + j == n - 1 || i == 0 || i == n - 1 || j == 0 || j == n - 1)
            printf("*");
         else
            printf(" ");
      }
      printf("\n");
   }
   return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
int i, j, n;
cin >> n;  // 'n' must be odd
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// left diagonal, right diagonal, top horizontal, bottom horizontal, left vertical, right vertical
if(i == j || i + j == n - 1 || i == 0 || i == n - 1 || j == 0 || j == n - 1)
cout << "*";
else
cout << " ";
```

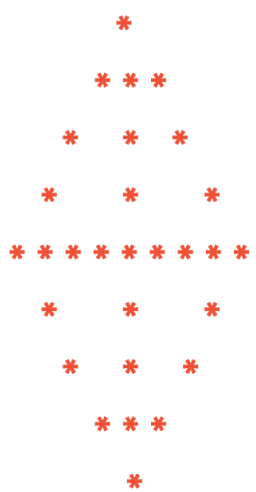
Java

```
import java.util.Scanner;
public class Main
{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
int i, j;
int n = sc.nextInt();  // 'n' must be odd
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// left diagonal, right diagonal, top horizontal, bottom horizontal, left vertical, right vertical
if(i == j || i + j == n - 1 || i == 0 || i == n - 1 || j == 0 || j == n - 1)
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

#12 - Diagonal & Sides of a Rhombus/Diamond

Input: 9 (input should be an odd number only, else the desired output will not be obtained)

Output:



```
int i, j, n;
scanf("%d", &n);  // ‘n’ must be odd
int num1 = n / 2 * 3;
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, center vertical, upper left diagonal, bottom left diagonal, upper right diag
if(i == n / 2 || j == n / 2 || i + j == n / 2 || i - j == n / 2 || j - i == n / 2 || i + j == num1)
printf("*");
else
printf(" ");
}
printf("\n");
}
return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
int i, j, n;
cin >> n;  // ‘n’ must be odd
int num1 = n / 2 * 3;
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, center vertical, upper left diagonal, bottom left diagonal, upper right diag
if(i == n / 2 || j == n / 2 || i + j == n / 2 || i - j == n / 2 || j - i == n / 2 || i + j == num1)
cout << "*";
else
cout << " ";
}
cout << "\n";
}
return 0;
}
```

Java

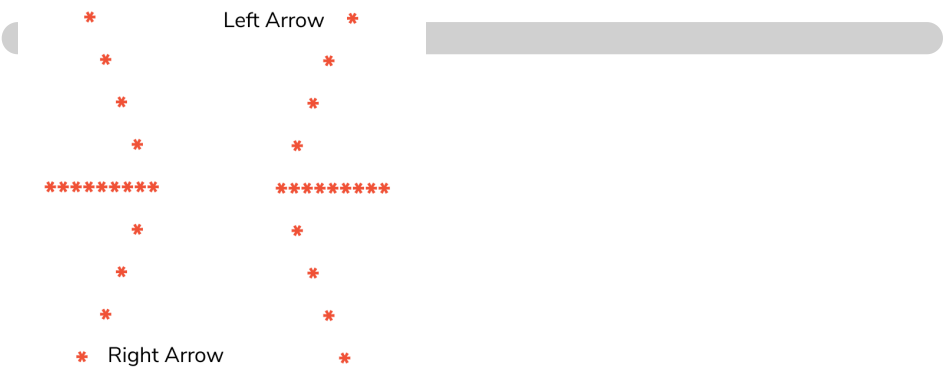
```
import java.util.Scanner;
public class Main
{
```

```
for(j = 0; j < n; j++)
{
// center horizontal, center vertical, upper left diagonal, bottom left diagonal, upper right diag
if(i == n / 2 || j == n / 2 || i + j == n / 2 || i - j == n / 2 || j - i == n / 2 || i + j == num1)
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

#13 - Left and Right Arrows

Input: 7 (Here n is the height and width of pattern to be printed)

Output:



Solution for Printing the left arrow & right arrow:

C

```
#include <stdio.h>
int main()
{
int i, j, n;
scanf("%d", &n);  // 'n' must be odd
int num1 = n / 2 * 3;
// right arrow
printf("Right Arrow\n");
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, upper right diagonal, bottom right diagonal
```

```
printf("Left Arrow\n");
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, bottom left diagonal, upper left diagonal
if(i == n / 2 || i - j == n / 2 || i + j == n / 2)
printf("*");
else
printf(" ");
}
printf("\n");
}
return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
int i, j, n;
cin >> n;  // 'n' must be odd
int num1 = n / 2 * 3;
// right arrow
cout << "Right Arrow" << endl;
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, upper right diagonal, bottom right diagonal
if(i == n / 2 || j - i == n / 2 || i + j == num1)
cout << "*";
else
cout << " ";
}
cout << "\n";
}
// left arrow
cout << "Left Arrow" << endl;
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, bottom left diagonal, upper left diagonal
if(i == n / 2 || i - j == n / 2 || i + j == n / 2)
cout << "*";
else
cout << " ";
```

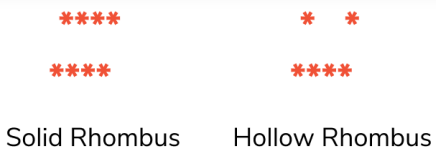
Java

```
import java.util.Scanner;
public class Main{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
int i, j;
int n = sc.nextInt();  // 'n' must be odd
int num1 = n / 2 * 3;
// right arrow
System.out.println("Right Arrow");
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, upper right diagonal, bottom right diagonalif(i == n / 2 || j - i == n / 2 || i +
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
// left arrow
System.out.println("Left Arrow");
for(i = 0; i < n; i++)
{
for(j = 0; j < n; j++)
{
// center horizontal, bottom left diagonal, upper left diagonalif(i == n / 2 || i - j == n / 2 || i + j =
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

#14 - Rhombus Pattern Program in C, C++, Java

Input: 4

Output:



Solid Rhombus Hollow Rhombus

Solution for Printing a solid and hollow Rhombus:

C

```
#include <stdio.h>
int main()
{
int i, j, n;
scanf("%d", &n);
// solid rhombus
printf("Solid Rhombus\n");
for(i = 0; i < n; i++)
{
for(j = 0; j < n - i; j++)
{
printf(" ");  // leading spaces
}
for(j = 0; j < n; j++)
{
printf("*");
}
printf("\n");
}
// hollow rhombus
printf("Hollow Rhombus\n");
for(i = 0; i < n; i++)
{
for(j = 0; j < n - i; j++)
{
printf(" ");  // leading spaces
}
for(j = 0; j < n; j++)
{
// upper horizontal, bottom horizontal, left diagonal, right diagonal
if(i == 0 || i == n - 1 || j == 0 || j == n - 1)
printf("*");
else
printf(" ");
}
printf("\n");
}
return 0;
}
```

```
{
int i, j, n;
cin >> n;
// solid rhombus
cout << "Solid Rhombus" << endl;
for(i = 0; i < n; i++)
{
for(j = 0; j < n - i; j++)
{
cout << " ";  // leading spaces
}
for(j = 0; j < n; j++)
{
cout << "*";
}
cout << "\n";
}
// hollow rhombus
cout << "Hollow Rhombus" << endl;
for(i = 0; i < n; i++)
{
for(j = 0; j < n - i; j++)
{
cout << " ";  // leading spaces
}
for(j = 0; j < n; j++)
{
// upper horizontal, bottom horizontal, left diagonal, right diagonal
if(i == 0 || i == n - 1 || j == 0 || j == n - 1)
cout << "*";
else
cout << " ";
}
cout << "\n";
}
return 0;
}
```

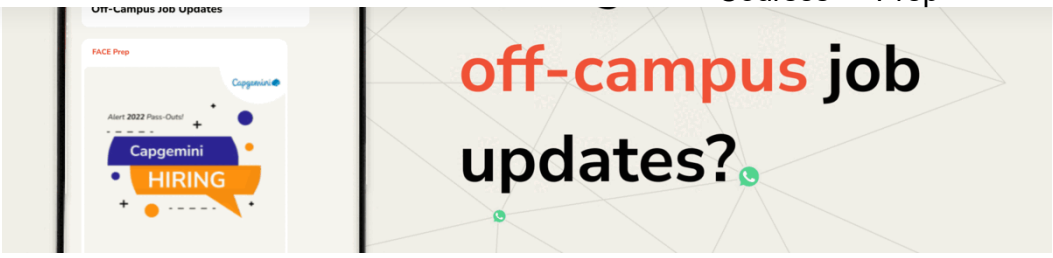
Java

```
import java.util.Scanner;
public class Main{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
int i, j;
int n = sc.nextInt();
// solid rhombus
System.out.println("Solid Rhombus");
for(i = 0; i < n; i++)
```

```
System.out.print("*");
}
System.out.println();
}
// hollow rhombus
System.out.println("Hollow Rhombus");
for(i = 0; i < n; i++)
{
for(j = 0; j < n - i; j++)
{
System.out.print(" ");  // leading spacesfor(j = 0; j < n; j++)
{
// upper horizontal, bottom horizontal, left diagonal, right diagonalif(i == 0 || i == n - 1 || j == 0
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

Check out more C Programming questions

Useful video on Pattern Programming for beginners



FACE Prep

CATEGORIES

- Aptitude
- Programming & DSA
- Recruitment Essentials
- Tech Skills
- Business Skills
- Human Skills

RESOURCES

- Skill Zone
- Company Corner
- Webinars
- Articles
- Tests
- Videos
- FACE Prep Edge
- Company-specific courses

TOPICS

- Quantitative Aptitude
- Logical Reasoning
- Verbal Ability
- Data Structures
- Algorithms
- C programming
- C++ Programming
- Java programming
- Python programming
- Interview Prep

FACE PREP

- Wall of Love
- Contact Us
- Terms and conditions
- Privacy Policy

SOCIAL

- YouTube
- Instagram
- LinkedIn
- Facebook

COMPANIES

- Google
- TCS
- Cognizant
- TATA ELXSI
- Tech Mahindra
- Deloitte
- IBM
- BYJU's
- Deloitte
- Goldman Sachs

Tech Mahindra
IBM
Wipro
BYJU's
Infosys
eLitmus
TCS Ninja
Cocubes
Mindtree