

```

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

Half Pyramid

```

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

Inverted
Half Pyramid

```

*   *
*   *
*   *
*   *
    
```

Hollow Inverted
Half Pyramid

```

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

Full Pyramid

```

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

Inverted Full Pyramid

```

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

Hollow Full Pyramid

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

[Click here](#) to see the program to print the Pyramid Patterns below using numbers.

Follow
us on:



```

1 2
1 2 3
1 2 3 4
1 2 3 4 5
    
```

Half Pyramid

```

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

Inverted
Half Pyramid

```

1 2
1 3
1 4
1 2 3 4 5
    
```

Hollow
Half Pyramid

```

      1
    2 3 2
  3 4 5 4 3
4 5 6 7 6 5 4
5 6 7 8 9 8 7 6 5
    
```

Full Pyramid

```

      1
    1 2
  1 3
1 4
1 2 3 4 5
    
```

Hollow Full Pyramid

```

1 2 3 4 5
2 5
3 5
4 5
5
    
```

Hollow Inverted
Half Pyramid

Empower your placement journey with our
exclusive self-paced courses

FACE Prep PRO

At the lowest price of **Rs. 1999**/year

[Join Now](#)

**Join Now to get
access to**



6000+ Company specific
questions

Follow
us on:



#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.

```

1 2 1      A B A
1 2 3 2 1  A B C B A
1 2 3 4 3 2 1 A B C D C B A
1 2 3 4 5 4 3 2 1 A B C D E D C B A
    
```

```

      1      *****1*****
    1 2 1    *****2*2*****
  1 2 3 2 1  *****3*3*3*****
1 2 3 4 3 2 1 *****4*4*4*4*****
1 2 3 4 5 4 3 2 1 *****5*5*5*5*5*****
    
```

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.

Follow
us on:



```

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

Solid Diamond

```

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

Hollow Diamond

```

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

Solid Half
Diamond

Program to print the patterns below using numbers and stars:

- [Click here to view the solution of the first 3 problems.](#)
- [Click here to view the solution of the 4th problem.](#)

3	1	1	*
44	2*2	2*3	* 1 *
555	3*3*3	4*5*6	* 1 2 1 *
6666	4*4*4*4	7*8*9*10	* 1 2 3 2 1 *
555	4*4*4*4	7*8*9*10	* 1 2 1 *
44	3*3*3	4*5*6	* 1 *
3	2*2	2*3	*
	1	1	

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](#).

Follow
us on:



1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

16 17 18 19 20 21

22 23 24 25 26 27 28

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

[Click here to know the detailed solution of the below shown Pascal's triangle.](#)

row 0 = 1

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

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

Follow
us on:



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:

Follow
us on:





Follow
us on:



Solution for diamond inscribed inside rectangle:

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("*");
```

```

        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;
    // 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
                cout << "*";
            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

```

Follow
us on:




```

        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 pattern
        for(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(" ");
            }
            System.out.println();
        }
        // bottom half of the pattern
        for(i = 0; i < n; i++)
        {
            for(j = 0; j < (2 * n); j++)
            {
                if(i >= j) // bottom left triangle
                    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();
        }
    }
}

```

Follow
us on:



Everything you need for placement preparation at **One** place



6000+ Company
specific questions



4000+ Aptitude
Questions



Programming basics
in C++/ Python



500+ DSA Questions
for practice



180+ Mock Tests
& Assessments

JUST

Rs. 1999

valid for 1 year*

~~Rs. 3999~~

ENROL NOW

#10 - Butterfly Pattern Printing

Input: 5

Output:

Follow
us on:





Follow
us on:



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("*");
```

```

        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 <= 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

```

Follow
us on:



```

        if((i + n) <= 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 pattern
        for(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 pattern
        for(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();
        }
    }
}

```

Follow
us on:



#11 - Diagonal & Sides of a Rectangle

Output:

```

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

Follow
us on:



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 vertical
            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 << " ";
        }
        cout << "\n";
    }
    return 0;
}
```

Follow
us on:



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:

```

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

Follow
us on:



Solution for Printing the diagonal and sides of a Rhombus/Diamond:

C

```
#include <stdio.h>
int main()
```



```
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 di
        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 di
            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;
}
```

Follow
us on:



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;
```

```
{
// center horizontal, center vertical, upper left diagonal, bottom left diagonal, upper right di
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 the pattern to be printed)

Output:

```

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

Left Arrow

```

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

Right Arrow

Follow
us on:



```
#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
            if(i == n / 2 || j - i == n / 2 || i + j == num1)
                printf("*");
            else
                printf(" ");
        }
        printf("\n");
    }
    // left arrow
    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;
}
```

Follow
us on:



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
```

```
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 << " ";
}
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(); // '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 diagonal
if(i == n / 2 || j - i == n / 2 ||
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 diagonal
if(i == n / 2 || i - j == n / 2 || i +
System.out.print("*");
else
System.out.print(" ");
}
}
```

Follow
us on:



#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++)
  
```

Follow
us on:



```
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;
}
```

C++

```
#include <iostream>
using namespace std;
int main()
{
    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++)
        {
```

Follow
us on:



```
{
// 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++)
{
for(j = 0; j < n - i; j++)
{
System.out.print(" "); // leading spaces
}
for(j = 0; j < n; j++)
{
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 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)
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

Follow
us on:

