How to Print Pattern in Java

Java pattern program enhances the coding skill, logic, and looping concepts. It is mostly asked in **Java interview** to check the logic and thinking of the programmer. We can print a **Java pattern program** in different designs. To learn the pattern program, we must have a deep knowledge of the Java loop, such as **for** loop **do-while** loop. In this section, we will learn **how to print a pattern in Java**.

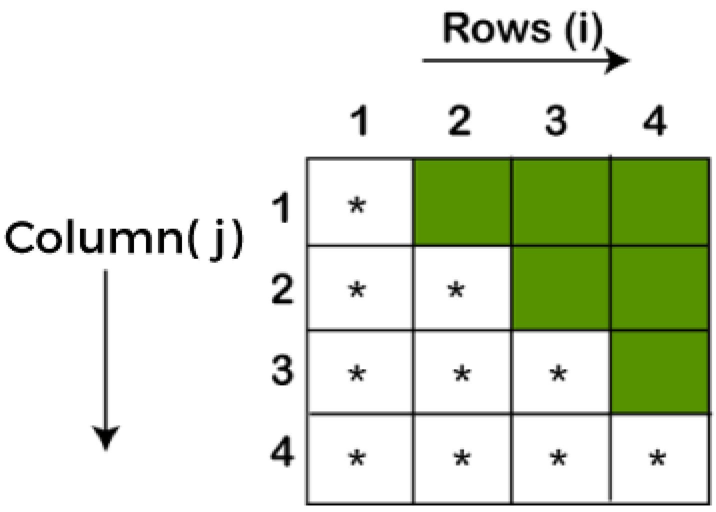
We have classified the **Java pattern program** into three categories:

- **Start Pattern**
- **Number Pattern**
- **Character Pattern**

Before moving to the pattern programs, let's see the approach.

Whenever you design logic for a pattern program, first draw that pattern in the blocks, as we have shown in the following image. The figure presents a clear look of the pattern.

Each pattern program has two or more than two loops. The number of the loop depends on the complexity of pattern or logic. The first for loop works for the row and the second loop works for the column. In the pattern programs, **Java for loop** is widely used.



In the above pattern, the **row** is denoted by **i** and the **column** is denoted by **j**. We see that the first row prints only a star. The second-row prints two stars, and so on. The **colored** blocks print the **spaces**.

Let's create the logic for the pattern, give above. In the following code snippet, we are starting row and column value from 0. We can also start it from 1, it's your choice.

```
for(int i=0; i<row; i++)
{
for(int j=0; j<=i; j++)
{
System.out.print("* ");
}
System.out.println();
```

In the above code snippet, the first for loop is for row and the second for loop for columns.

Let's see the execution of the code step by step, for **n=4** (the number of rows we want to print).

Iteration 1:

```
For i=0, 0<4 (true)
For j=0, j<=0 (true)
```

The first **print** statement prints a star at the first row and the second **println** statement prints the spaces and throws the cursor at the next line.

```
*
```

Now the value of i and j is increased to 1.

Iteration 2:

```
For i=1, 1<4 (true)
For j=1, 1<=1 (true)
```

The first **print** statement prints two stars at the second row and the second **println** statement prints the spaces and throws the cursor at the next line.

```
*
* *
```

Now the value of i and j is increased to 2.

Iteration 3:

```
For i=2, 2<4 (true)
For j=2, 2<=2 (true)
```

The first **print** statement prints three stars at the third row and the second **println** statement prints the spaces and throws the cursor at the next line.

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

Now the value of i and j is increased to 3.

Iteration 4:

```
For i=3, 3<4 (true)
For j=3, 3<=3 (true)
```

The first **print** statement prints four stars at the fourth row and the second **println** statement prints the spaces and throws the cursor at the next line.

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

Now the value of i and j is increased to 4.

```
For i=4, 4<4 (false)
```

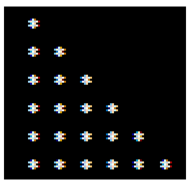
The execution of the program will terminate when the value of i will be equal to the number of rows.

Star Pattern

1. Right Triangle Star Pattern

```java
public class RightTrianglePattern
{
public static void main(String args[])
{
//i for rows and j for columns
//row denotes the number of rows you want to print
int i, j, row=6;
//outer loop for rows
for(i=0; i<row; i++)
{
//inner loop for columns
for(j=0; j<=i; j++)
{
//prints stars
System.out.print("* ");
}
//throws the cursor in a new line after printing each line
System.out.println();
}
}
}
```

Output:

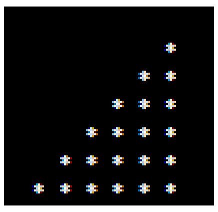


```
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

2. Left Triangle Star Pattern

```
public class LeftTrianglePattern
{
public static void main(String args[])
{
//i for rows and j for columns
//row denotes the number of rows you want to print
int i, j, row = 6;
//Outer loop work for rows
for (i=0; i<row; i++)
{
//inner loop work for space
for (j=2*(row-i); j>=0; j--)
{
//prints space between two stars
System.out.print(" ");
}
//inner loop for columns
for (j=0; j<=i; j++ )
{
//prints star
System.out.print("* ");
}
//throws the cursor in a new line after printing each line
System.out.println();
}
}
}
```

Output:

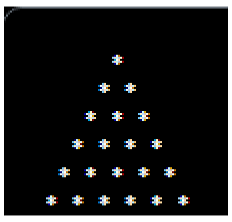


```
          *
        * *
      * * *
    * * * *
  * * * * *
* * * * * *
```

3. Pyramid Star Pattern

```
public class PyramidPattern
{
public static void main(String args[])
{
//i for rows and j for columns
//row denotes the number of rows you want to print
int i, j, row = 6;
//Outer loop work for rows
for (i=0; i<row; i++)
{
//inner loop work for space
for (j=row-i; j>1; j--)
{
//prints space between two stars
System.out.print(" ");
}
//inner loop for columns
for (j=0; j<=i; j++ )
{
//prints star
System.out.print("* ");
}
//throws the cursor in a new line after printing each line
System.out.println();
}
}
}
```

Output:



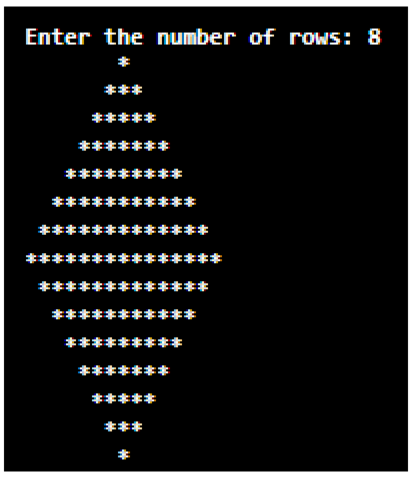
```
     *
    * *
   * * *
  * * * *
 * * * * *
* * * * * *
```

4. Diamond Shape Pattern

```
import java.util.Scanner;
public class DiamondPattern
{
public static void main(String args[])
{
int row, i, j, space = 1;
System.out.print("Enter the number of rows you want to print: ");
Scanner sc = new Scanner(System.in);
row = sc.nextInt();
space = row - 1;
for (j = 1; j<= row; j++)
{
for (i = 1; i<= space; i++)
{
System.out.print(" ");
}
space--;
for (i = 1; i <= 2 * j - 1; i++)
{
System.out.print("*");
}
System.out.println("");
}
space = 1;
for (j = 1; j<= row - 1; j++)
{
for (i = 1; i<= space; i++)
{
```

```
System.out.print(" ");
}
space++;
for (i = 1; i<= 2 * (row - j) - 1; i++)
{
System.out.print("*");
}
System.out.println("");
}
}
}
```

Output:

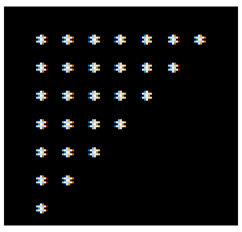


```
Enter the number of rows: 8
       *
      ***
     *****
    *******
   *********
  ***********
 *************
***************
 *************
  ***********
   *********
    *******
     *****
      ***
       *
```

5. Downward Triangle Star Pattern

```
public class DownwardTrianglePattern
{
public static void main(String[] args)
{
int rows=7;
//inner loop
for (int i= rows-1; i>=0 ; i--)
{
//outer loop
for (int j=0; j<=i; j++)
{
//prints star and space
System.out.print("*" + " ");
}
//throws the cursor in the next line after printing each line
System.out.println();
}
}
}
```

Output:



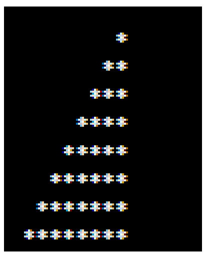
```
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
```

6. Mirrored Right Triangle Star Pattern

```
public class MirroredRightTrianglePattern
{
public static void main(String[] args)
{
```

```
int n=7;
//inner loop
for (int i= 0; i<= n; i++)
{
//outer loop
for (int j=1; j<=n-i; j++)
{
System.out.print(" ");
}
for (int k=0;k<=i;k++)
{
System.out.print("*");
}
System.out.println("");
}
}
}
```

Output:

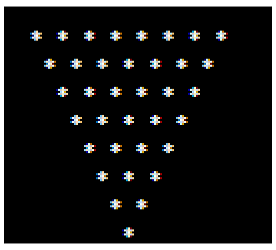


```
       *
      **
     ***
    ****
   *****
  ******
 *******
********
```

7. Reverse Pyramid Star Pattern

```
public class ReversePyramidPattern
{
public static void main(String[] args)
{
int rows=8;
for (int i= 0; i<= rows-1; i++)
{
for (int j=0; j<=i; j++)
{
System.out.print(" ");
}
for (int k=0; k<=rows-1-i; k++)
{
System.out.print("*" + " ");
}
System.out.println();
}
}
}
```

Output:



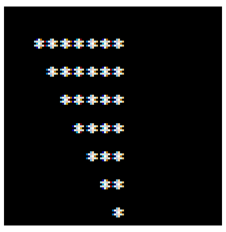
```
 * * * * * * * *
  * * * * * * *
   * * * * * *
    * * * * *
     * * * *
      * * *
       * *
        *
```

8. Right Down Mirror Star Pattern

```
public class RightDownMirrorPattern
{
```

```
public static void main(String args[])
{
int row=7;
for (int i= row; i>= 1; i--)
{
for (int j=row; j>i;j--)
{
System.out.print(" ");
}
for (int k=1;k<=i;k++)
{
System.out.print("*");
}
System.out.println("");
}
}
}
```

Output:

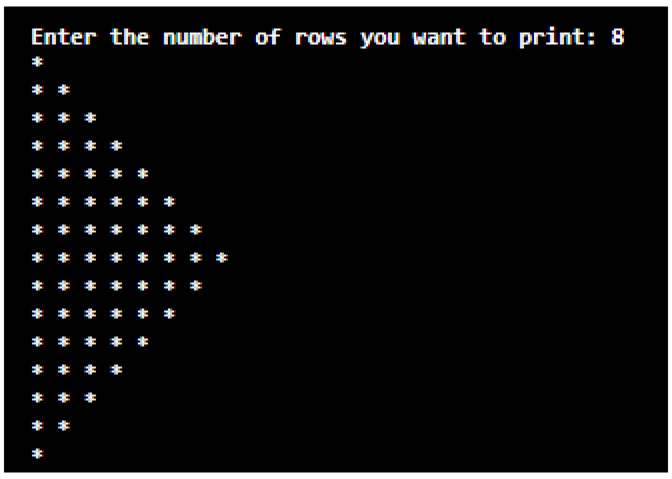


```
*******
 ******
  *****
   ****
    ***
     **
      *
```

9. Right Pascal's Triangle

```
import java.util.Scanner;
public class RightPascalTrianglePattern
{
public static void main(String[] args)
{
int i, j, rows;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows you want to print: ");
rows = sc.nextInt();
for (i= 0; i<= rows-1; i++)
{
for (j=0; j<=i; j++)
{
System.out.print("*"+ " ");
}
System.out.println("");
}
for (i=rows-1; i>=0; i--)
{
for(j=0; j <= i-1;j++)
{
System.out.print("*"+ " ");
}
System.out.println("");
}
}
}
```

Output:

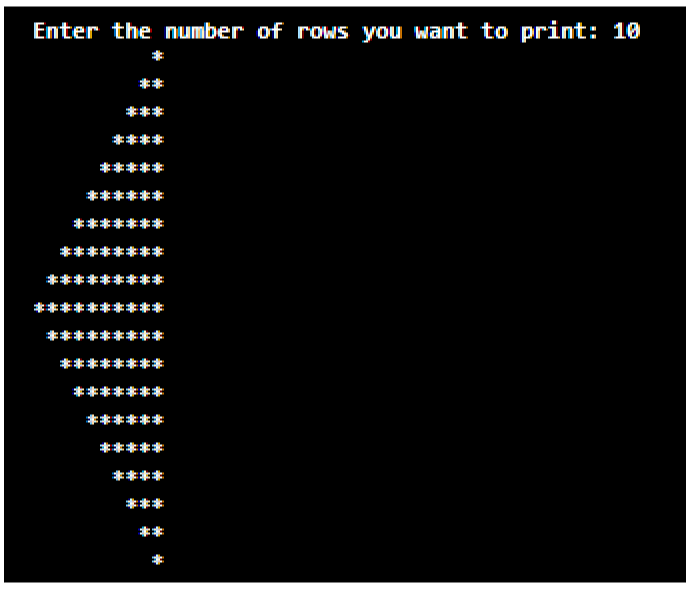


```
Enter the number of rows you want to print: 8
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
```

10. Left Pascal's Triangle

```java
import java.util.Scanner;
public class LeftPascalTrianglePattern
{
public static void main(String[] args)
{
int i, j, k, rows;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows you want to print: ");
rows = sc.nextInt();
for (i= 1; i<= rows ; i++)
{
for (j=i; j <rows ;j++)
{
System.out.print(" ");
}
for (k=1; k<=i;k++)
{
System.out.print("*");
}
System.out.println("");
}
for (i=rows; i>=1; i--)
{
for(j=i; j<=rows;j++)
{
System.out.print(" ");
}
for(k=1; k<i ;k++)
{
System.out.print("*");
}
System.out.println("");
}
sc.close();
}
}
```

Output:

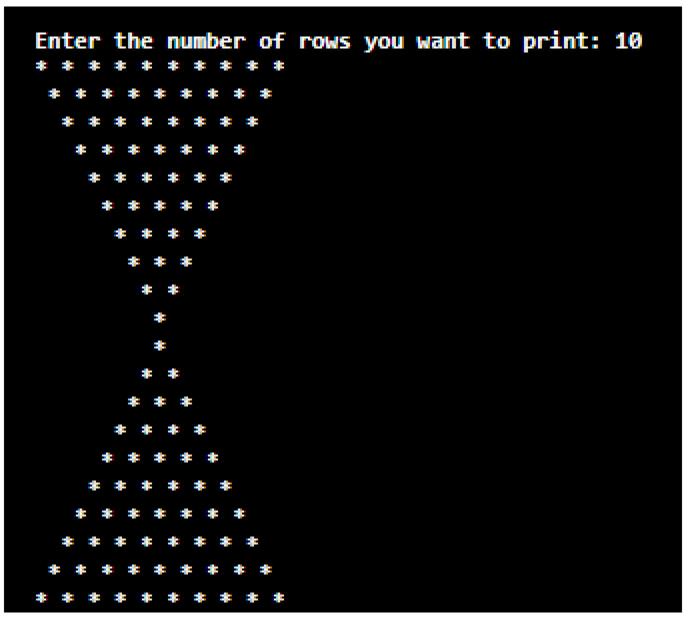


```
Enter the number of rows you want to print: 10
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
**********
 *********
  ********
   *******
    ******
     *****
      ****
       ***
        **
         *
```

11. Sandglass Star Pattern

```java
import java.util.Scanner;
public class SandglassPattern
{
public static void main(String[] args)
{
int i, j, k, n;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows you want to print: ");
n = sc.nextInt();
for (i= 0; i<= n-1 ; i++)
{
for (j=0; j<i; j++)
{
System.out.print(" ");
}
for (k=i; k<=n-1; k++)
{
System.out.print("*" + " ");
}
System.out.println("");
}
for (i= n-1; i>= 0; i--)
{
for (j=0; j<i; j++)
{
System.out.print(" ");
}
for (k=i; k<=n-1; k++)
{
System.out.print("*" + " ");
}
System.out.println("");
}
sc.close();
}
}
```

Output:

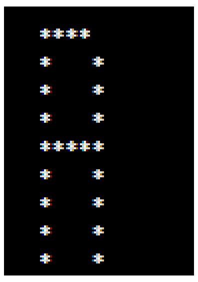


```
Enter the number of rows you want to print: 10
* * * * * * * * * *
 * * * * * * * * *
  * * * * * * * *
   * * * * * * *
    * * * * * *
     * * * * *
      * * * *
       * * *
        * *
         *
         *
        * *
       * * *
      * * * *
     * * * * *
    * * * * * *
   * * * * * * *
  * * * * * * * *
 * * * * * * * * *
* * * * * * * * * *
```

12. Alphabet Star Pattern

```java
import java.util.*;
public class AlphabetPattern
{
public static void main(String[] args)
{
int i, j, n=8;
// Outer for loop for number of lines
for (i = 0; i<=n; i++)
{
// Inner for loop for logic execution
for (j = 0; j<= n / 2; j++)
{
// prints two vertical lines
if ((j == 0 || j == n / 2) && i != 0 ||
// print first line of alphabet
i == 0  && j != n / 2 ||
// prints middle line
i == n / 2)
System.out.print("*");
else
System.out.print(" ");
}
System.out.println();
}
}
}
```

Output:

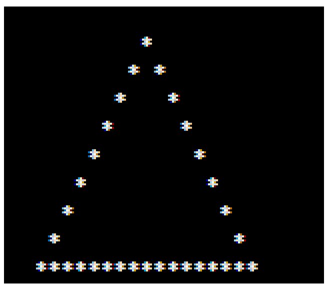


```
****
*   *
*   *
*   *
*****
*   *
*   *
*   *
*   *
```

13. Triangle Star Pattern

```java
import java.util.Scanner;
public class TrianglePattern
{
public static void main(String[] args)
{
int i, j, k, rows=9;
for (i=1; i<= rows ; i++)
{
for (j = i; j < rows ; j++)
{
System.out.print(" ");
}
for (k = 1; k <= (2*i -1) ;k++)
{
if(k==1 || i == rows || k==(2*i-1))
{
System.out.print("*");
}
else
{
System.out.print(" ");
}
}
System.out.println("");
}
}
}
```

Output:

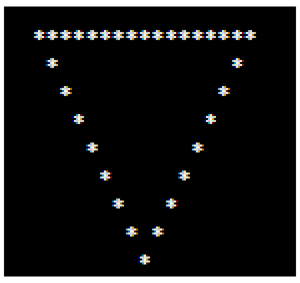


```
        *
       * *
      *   *
     *     *
    *       *
   *         *
  *           *
 *             *
*****************
```

14. Down Triangle Pattern

```
import java.util.Scanner;
public class DownTrianglePattern
{
public static void main(String[] args)
{
int i, j, k, rows=9;
for (i=rows; i>= 1 ; i--)
{
for (j = i; j<rows ; j++)
{
System.out.print(" ");
}
for (k = 1; k <= (2*i -1) ;k++)
{
if( k==1 || i == rows || k==(2*i-1))
{
System.out.print("*");
}
else
{
System.out.print(" ");
}
}
System.out.println("");
}
}
}
```

Output:

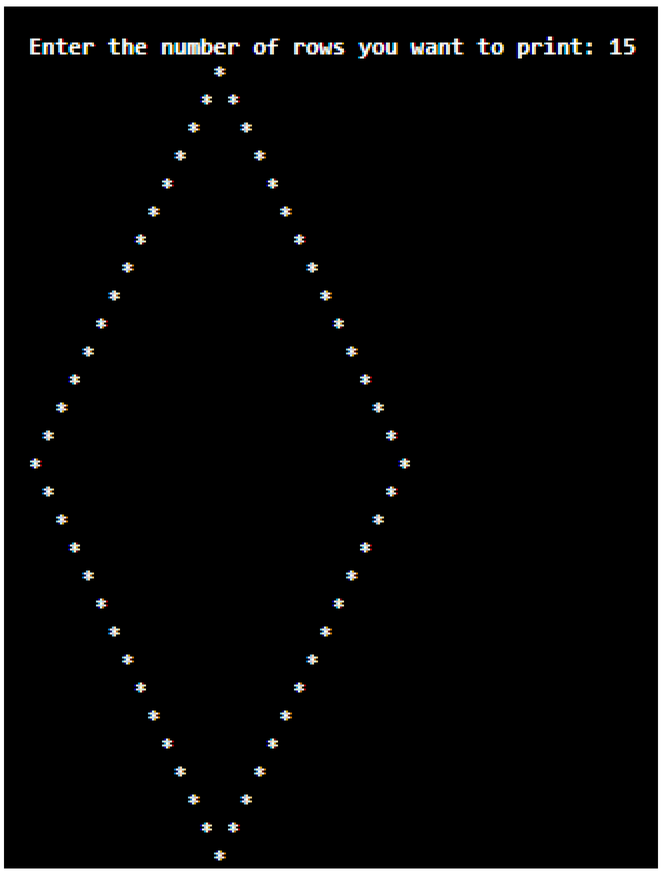


```
*****************
 *             *
  *           *
   *         *
    *       *
     *     *
      *   *
       * *
        *
```

15. Diamond Star Pattern

```
import java.util.*;
public class DiamondPattern
{
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
System.out.println("Enter the number of rows you want to print: ");
int rows = sc.nextInt();
for (i=1; i<= rows ; i++)
{
for (j = rows; j > i ; j--)
{
System.out.print(" ");
}
System.out.print("*");
for (k = 1; k < 2*(i -1) ;k++)
{
System.out.print(" ");
}
if( i==1)
{
System.out.println("");
}
else
{
System.out.println("*");
}
}
for (i=rows-1; i>= 1 ; i--)
{
for (int j = rows; j > i ; j--)
{
System.out.print(" ");
}
System.out.print("*");
for (int k = 1; k < 2*(i -1) ;k++)
{
System.out.print(" ");
}
if(i==1)
System.out.println("");
else
System.out.println("*");
}
}
}
```

Output:



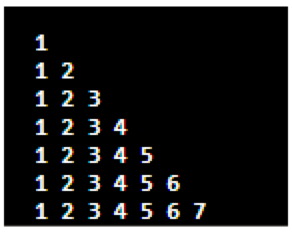
```
Enter the number of rows you want to print: 15
              *
             * *
            *   *
           *     *
          *       *
         *         *
        *           *
       *             *
      *               *
     *                 *
    *                   *
   *                     *
  *                       *
 *                         *
*                           *
 *                         *
  *                       *
   *                     *
    *                   *
     *                 *
      *               *
       *             *
        *           *
         *         *
          *       *
           *     *
            *   *
             * *
              *
```

Number Pattern

1. Pattern-1

```
public class Pattern1
{
public static void main(String args[])
{
int i, j,number, n=7;
//loop for rows
for(i=0; i<n; i++)
{
number=1;
//loop for columns
for(j=0; j<=i; j++)
{
//prints num
System.out.print(number+ " ");
//incrementing the value of number
number++;
}
//throws the cursor at the next line after printing each row
System.out.println();
}
}
}
```

Output:



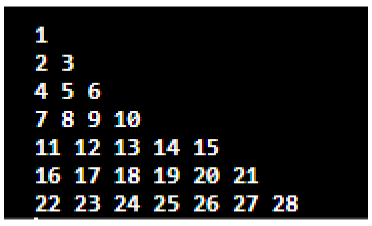
```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
```

2. Pattern-2

```
public class Pattern2
{
public static void main(String[] args)
{
```

```
int i, j, k = 1;
//inner loop
for (i = 1; i <= 7; i++)
{
//outer loop
for (j = 1; j< i + 1; j++)
{
//prints the value of k
System.out.print(k++ + " ");
}
//throws the cursor at the next line
System.out.println();
}
}
}
```

Output:

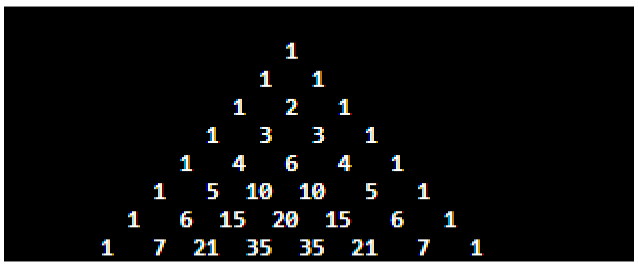


```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28
```

3. Pattern-3

```
public class Pattern3
{
public static void main(String[] args)
{
int n = 8;    //n is the number of rows you want to print
for (int i = 0; i < n; i++)
{
int number = 1;
System.out.printf("%" + (n - i) * 2 + "s", "");
for (int j = 0; j <= i; j++)
{
System.out.printf("%4d", number);
number = number * (i - j) / (j + 1);
}
System.out.println();
}
}
}
```

Output:

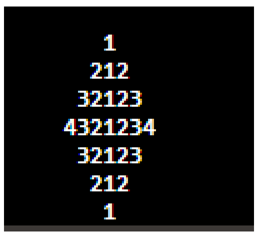


```
                   1
                 1   1
               1   2   1
             1   3   3   1
           1   4   6   4   1
         1   5  10  10   5   1
       1   6  15  20  15   6   1
     1   7  21  35  35  21   7   1
```

4. Pattern-4

```
public class Pattern4
{
public static void main(String[] args)
{
for (int i = 1; i <= 4; i++)
{
int n = 8;
for (int j = 1; j<= n - i; j++)
{
System.out.print(" ");
}
for (int k = i; k >= 1; k--)
{
System.out.print(k);
}
for (int l = 2; l <= i; l++)
{
System.out.print(l);
}
System.out.println();
}
for (int i = 3; i >= 1; i--)
{
int n = 10;
for (int j = 0; j<= n - i; j++)
{
System.out.print(" ");
}
for (int k = i; k >= 1; k--)
{
System.out.print(k);
}
for (int l = 2; l <= i; l++)
{
System.out.print(l);
}
System.out.println();
}
}
}
```

Output:



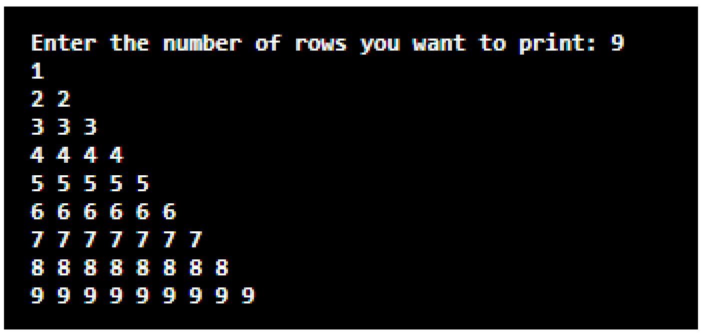
```
   1
  212
 32123
4321234
 32123
  212
   1
```

5. Pattern-5

```
import java.util.*;
public class Pattern5
{
public static void main(String[] args)
{
int i, j, rows;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows you want to print: ");
 rows = sc.nextInt();
for (i = 1; i <= rows; i++)
{
for (j = 1; j <= i; j++)
{
```

```
System.out.print(i+" ");
}
System.out.println();
}
}
}
```

Output:



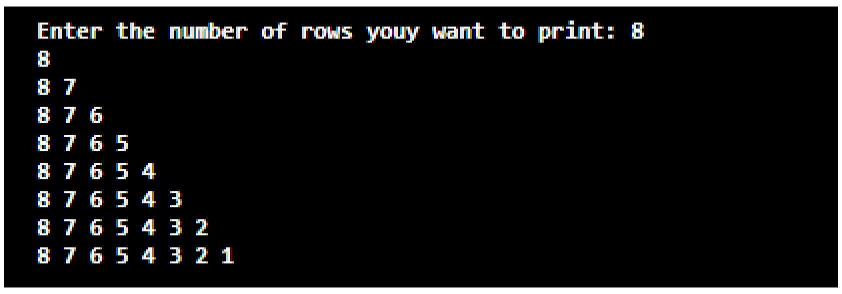
```
Enter the number of rows you want to print: 9
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
7 7 7 7 7 7 7
8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9
```

6. Pattern-6

```
import java.util.*;
public class Pattern6
{
public static void main(String[] args)
{
int i, j, rows;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows youy want to print: ");
rows = sc.nextInt();
for (i = rows; i >= 1; i--)
{
for (j = rows; j >= i; j--)
{
System.out.print(j+" ");
}

System.out.println();
}
}
}
```

Output:

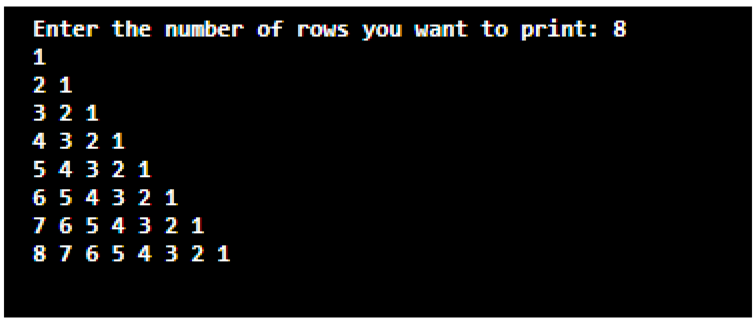


```
Enter the number of rows youy want to print: 8
8
8 7
8 7 6
8 7 6 5
8 7 6 5 4
8 7 6 5 4 3
8 7 6 5 4 3 2
8 7 6 5 4 3 2 1
```

7. Pattern-7

```
import java.util.Scanner;
public class Pattern7
{
public static void main(String[] args)
{
int i, j, n;
Scanner sc = new Scanner(System.in);
System.out.print("Enter the number of rows you want to print: ");
n = sc.nextInt();
for (i = 1; i <= n; i++)
{
for (j = i; j >= 1; j--)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

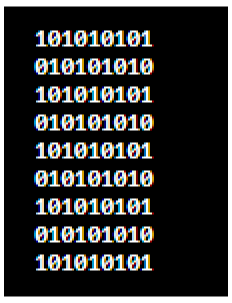


```
Enter the number of rows you want to print: 8
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1
7 6 5 4 3 2 1
8 7 6 5 4 3 2 1
```

8. Pattern-8

```
public class Pattern8
{
public static void main(String[] args)
{
int rows=9;   //number of rows to print
for (int i = 1; i <= rows; i++)
{
int num;
if(i%2 == 0)
{
num = 0;
for (int j = 1; j <= rows; j++)
{
System.out.print(num);
num = (num == 0)? 1 : 0;
}
}
else
{
num = 1;
for (int j = 1; j <= rows; j++)
{
System.out.print(num);
num = (num == 0)? 1 : 0;
}
}
System.out.println();
}
}
}
```

Output:

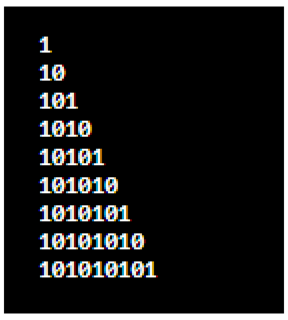


```
101010101
010101010
101010101
010101010
101010101
010101010
101010101
010101010
101010101
```

9. Pattern-9

```
import java.util.Scanner;
public class Pattern9
{
public static void main(String[] args)
{
int i, j, rows=9;
for (i = 1; i <= rows; i++)
{
for (j = 1; j <= i; j++)
{
if(j%2 == 0)
{
System.out.print(0);
}
else
{
System.out.print(1);
}
}
System.out.println();
}
}
}
```

Output:

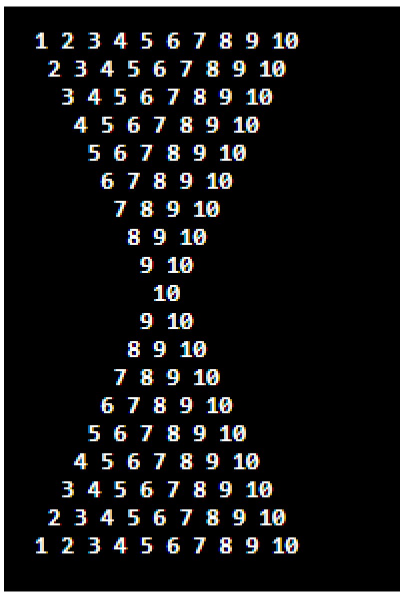


```
1
10
101
1010
10101
101010
1010101
10101010
101010101
```

10. Pattern-10

```
public class Pattern10
{
public static void main(String[] args)
{
int n = 10;
for (int i = 1; i <= n; i++)
{
for (int j = 1; j < i; j++)
{
System.out.print(" ");
}
for (int k = i; k <= n; k++)
{
System.out.print(k+" ");
}
System.out.println();
}
for (int i = n-1; i >= 1; i--)
{
for (int j = 1; j < i; j++)
{
System.out.print(" ");
}
for (int k = i; k <= n; k++)
{
System.out.print(k+" ");
}
System.out.println();
}
}
}
```

Output:

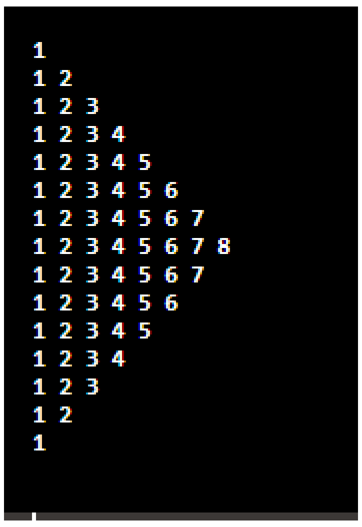


```
1 2 3 4 5 6 7 8 9 10
 2 3 4 5 6 7 8 9 10
  3 4 5 6 7 8 9 10
   4 5 6 7 8 9 10
    5 6 7 8 9 10
     6 7 8 9 10
      7 8 9 10
       8 9 10
        9 10
         10
        9 10
       8 9 10
      7 8 9 10
     6 7 8 9 10
    5 6 7 8 9 10
   4 5 6 7 8 9 10
  3 4 5 6 7 8 9 10
 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
```

11. Pattern-11

```
public class Pattern11
{
public static void main(String[] args)
{
int rows=8;
//Prints upper half pattern
for (int i = 1; i <= rows; i++)
{
for (int j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
System.out.println();
}
//prints lower half pattern
for (int i = rows-1; i >= 1; i--)
{
for (int j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

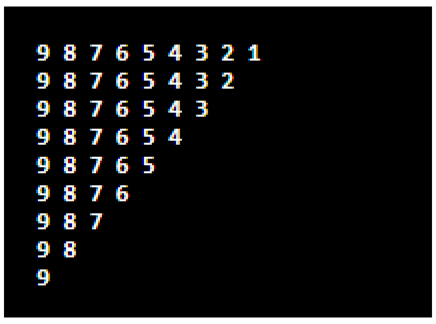


```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

12. Pattern-12

```
public class Pattern12
{
public static void main(String[] args)
{
int rows=9;
for (int i = 1; i <= rows; i++)
{
for (int j = rows; j >= i; j--)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

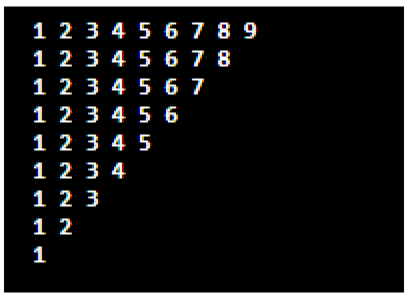


```
9 8 7 6 5 4 3 2 1
9 8 7 6 5 4 3 2
9 8 7 6 5 4 3
9 8 7 6 5 4
9 8 7 6 5
9 8 7 6
9 8 7
9 8
9
```

13. Pattern-13

```
public class Pattern14
{
public static void main(String[] args)
{
int i, j, rows=9;
for (i = rows; i >= 1; i--)
{
for (j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

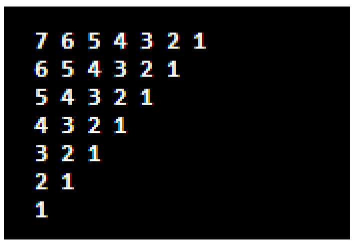


```
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

14. Pattern-14

```
public class Pattern14
{
public static void main(String[] args)
{
int rows=7, i, j;
for (i = rows; i >= 1; i--)
{
for (j = i; j >= 1; j--)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

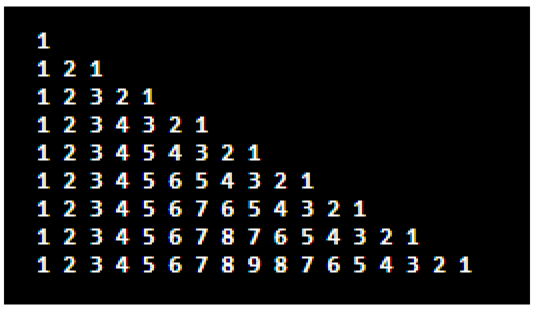


```
7 6 5 4 3 2 1
6 5 4 3 2 1
5 4 3 2 1
4 3 2 1
3 2 1
2 1
1
```

15. Pattern-15

```
public class Pattern15
{
public static void main(String[] args)
{
int i, j, rows=9;
for (i = 1; i <= rows; i++)
{
//Prints first half of the row
for (j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
//Prints second half of the row
for (j = i-1; j >= 1; j--)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:

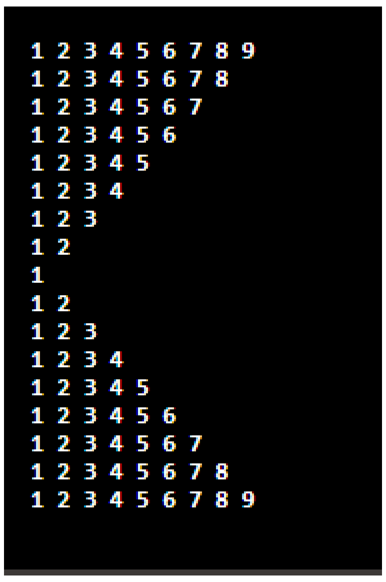


```
1
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1
1 2 3 4 5 4 3 2 1
1 2 3 4 5 6 5 4 3 2 1
1 2 3 4 5 6 7 6 5 4 3 2 1
1 2 3 4 5 6 7 8 7 6 5 4 3 2 1
1 2 3 4 5 6 7 8 9 8 7 6 5 4 3 2 1
```

16. Pattern-16

```java
public class Pattern16
{
public static void main(String[] args)
{
int i, j, rows=9;
//Prints upper half  pattern
for (i = rows; i >= 1; i--)
{
for (j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
System.out.println();
}
//Prints lower half  pattern
for (i = 2; i <= rows; i++)
{
for (j = 1; j <= i; j++)
{
System.out.print(j+" ");
}
System.out.println();
}
}
}
```

Output:



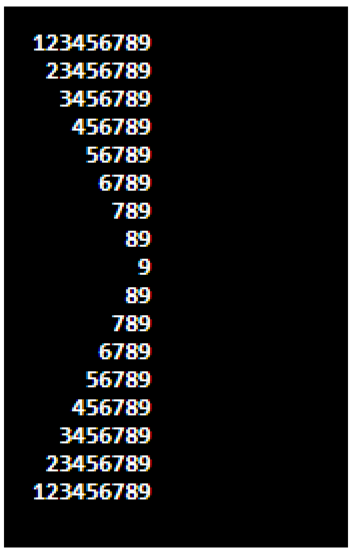
```
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
```

17. Pattern-17

```java
public class Pattern17
{
public static void main(String[] args)
{
int rows=9;
//Prints upper half pattern
for (int i = 1; i <= rows; i++)
{
```

```
//Prints i spaces at the beginning of each row
for (int j = 1; j < i; j++)
{
System.out.print(" ");
}
//Prints i to rows value at the end of each row
for (int j = i; j <= rows; j++)
{
System.out.print(j);
}
System.out.println();
}
//Prints lower half  pattern
for (int i = rows-1; i >= 1; i--)
{
//Prints i spaces at the beginning of each row
for (int j = 1; j < i; j++)
{
System.out.print(" ");
}
//Prints i to rows value at the end of each row
for (int j = i; j <= rows; j++)
{
System.out.print(j);
}
System.out.println();
}
}
}
```

Output:



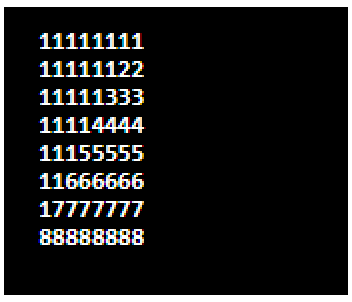
```
123456789
 23456789
  3456789
   456789
    56789
     6789
      789
       89
        9
       89
      789
     6789
    56789
   456789
  3456789
 23456789
123456789
```

18. Pattern-18

```
public class Pattern18
{
public static void main(String[] args)
{
int rows=8;
for (int i = 1; i <= rows; i++)
{
for (int j = 1; j <= rows-i; j++)
{
System.out.print(1);
}
for (int j = 1; j <= i; j++)
{
```

```
System.out.print(i);
}
System.out.println();
}
}
}
```

Output:

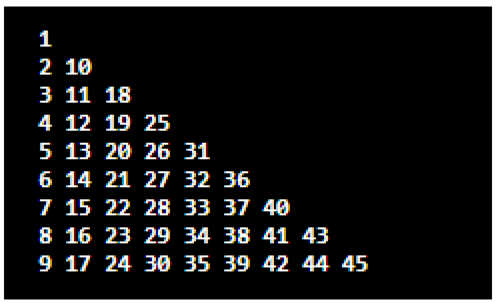


```
11111111
11111122
11111333
11114444
11155555
11666666
17777777
88888888
```

19. Pattern-19

```
public class Pattern19
{
public static void main(String args[])
{
int rows=9;
for (int i = 1; i <= rows; i++)
{
int num = i;
for (int j = 1; j <= i; j++)
{
System.out.print(num+" ");
num = num+rows-j;
}
System.out.println();
}
}
}
```

Output:



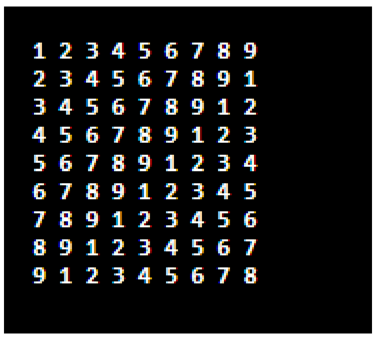
```
1
2 10
3 11 18
4 12 19 25
5 13 20 26 31
6 14 21 27 32 36
7 15 22 28 33 37 40
8 16 23 29 34 38 41 43
9 17 24 30 35 39 42 44 45
```

20. Pattern-20

```
public class Pattern20
{
public static void main(String[] args)
{
int i, j, k, rows=9;
for(i=1;i< rows+1 ;i++)
{
for(j=i; j < rows+1 ;j++)
{
System.out.print(j + " ");
}
for(k=1; k < i ;k++)
{
```

```
System.out.print(k + " ");
}
System.out.println();
}
}
}
```

Output:

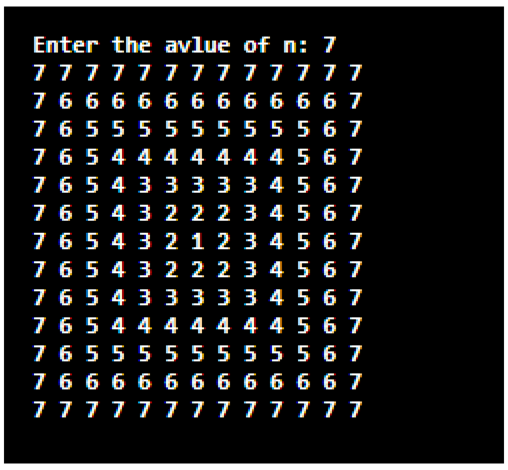


```
1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 9 1
3 4 5 6 7 8 9 1 2
4 5 6 7 8 9 1 2 3
5 6 7 8 9 1 2 3 4
6 7 8 9 1 2 3 4 5
7 8 9 1 2 3 4 5 6
8 9 1 2 3 4 5 6 7
9 1 2 3 4 5 6 7 8
```

21. Pattern-21

```
import java.util.Scanner;
public class Pattern21
{
public static void main(String[] args)
{
int i, j, min, n; //n is the number up to you want to print
System.out.print("Enter the value of n: ");
Scanner sc= new Scanner(System.in);
n=sc.nextInt();
//loo for upper left part
for (i = 1; i <= n; i++)
{
for (j = 1; j <= n; j++)
{
min = i < j ? i : j;
System.out.print(n - min + 1+ " ");
}
//loop for upper right part
for (j = n - 1; j >= 1; j--)
{
min = i < j ? i : j;
System.out.print(n - min + 1+ " ");
}
System.out.println();
}
//loop for lower left part
for (i = n - 1; i >= 1; i--)
{
for (j = 1; j <= n; j++)
{
min = i < j ? i : j;
System.out.print(n - min + 1+ " ");
}
//loop for lower right part
for (j = n - 1; j >= 1; j--)
{
min = i < j ? i : j;
System.out.print(n - min + 1+ " ");
}
System.out.println();
}
}
}
```

Output:



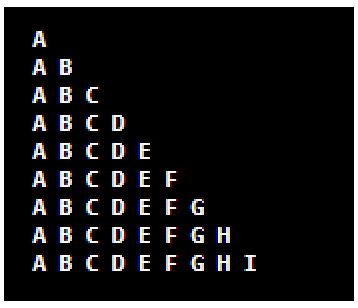
```
Enter the avlue of n: 7
7 7 7 7 7 7 7 7 7 7 7 7 7
7 6 6 6 6 6 6 6 6 6 6 6 7
7 6 5 5 5 5 5 5 5 5 5 6 7
7 6 5 4 4 4 4 4 4 4 5 6 7
7 6 5 4 3 3 3 3 3 4 5 6 7
7 6 5 4 3 2 2 2 3 4 5 6 7
7 6 5 4 3 2 1 2 3 4 5 6 7
7 6 5 4 3 2 2 2 3 4 5 6 7
7 6 5 4 3 3 3 3 3 4 5 6 7
7 6 5 4 4 4 4 4 4 4 5 6 7
7 6 5 5 5 5 5 5 5 5 5 6 7
7 6 6 6 6 6 6 6 6 6 6 6 7
7 7 7 7 7 7 7 7 7 7 7 7 7
```

Character Pattern

1. Right Triangle Alphabetic Pattern

```
public class RightAlphabaticPattern
{
public static void main(String[] args)
{
int alphabet = 65; //ASCII value of capital A is 65
//inner loop for rows
for (int i = 0; i <= 8; i++)
{
//outer loop for columns
for (int j = 0; j <= i; j++)
{
//adds the value of j in the ASCII value of A and prints the corresponding alphabet
System.out.print((char) (alphabet + j) + " ");
}
System.out.println();
}
}
}
```

Output:

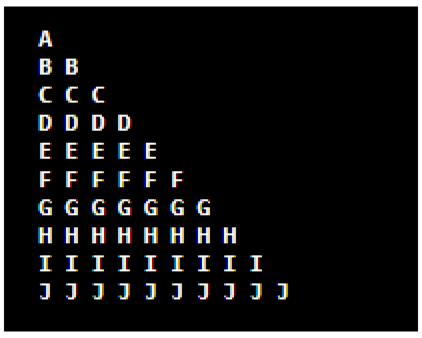


```
A
A B
A B C
A B C D
A B C D E
A B C D E F
A B C D E F G
A B C D E F G H
A B C D E F G H I
```

2. Repeating Alphabet Pattern

```
public class RepeatingPattern
{
public static void main(String[] args)
{
int letter = 65; //ASCII value of capital A is 65
//inner loop for rwos
for (int i = 0; i<= 9; i++)
{
//outer loop for columns
for (int j = 0; j <= i; j++)
{
//prints the character
System.out.print((char) letter + " ");
}
letter++;
System.out.println();
}
}
}
```

Output:

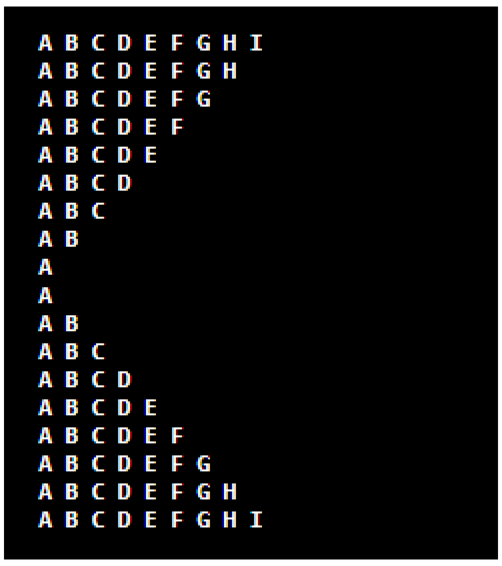


```
A
B B
C C C
D D D D
E E E E E
F F F F F F
G G G G G G G
H H H H H H H H
I I I I I I I I I
J J J J J J J J J J
```

3. K-shape Alphabet Pattern

```
public class KshapePattern
{
public static void main(String[] args)
{
for (int i = 8; i >= 0; i--)
{
int alphabet = 65;
for (int j = 0; j <= i; j++)
{
System.out.print((char) (alphabet + j) + " ");
}
System.out.println();
}
for (int i = 0; i<= 8; i++)
{
int alphabet = 65;
for (int j = 0; j <= i; j++)
{
System.out.print((char) (alphabet + j) + " ");
}
System.out.println();
}
}
}
```

Output:

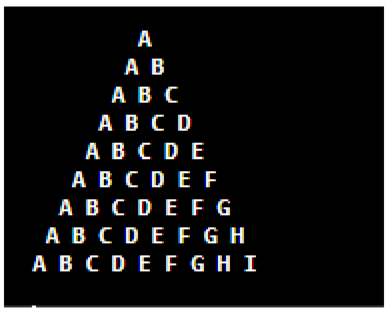


```
A B C D E F G H I
A B C D E F G H
A B C D E F G
A B C D E F
A B C D E
A B C D
A B C
A B
A
A
A B
A B C
A B C D
A B C D E
A B C D E F
A B C D E F G
A B C D E F G H
A B C D E F G H I
```

4. Triangle Character Pattern

```
public class TriangleCharacterPattern
{
public static void main(String[] args)
{
for (int i = 0; i <= 8; i++)
{
int alphabet = 65;
for (int j = 8; j > i; j--)
{
System.out.print(" ");
}
for (int k = 0; k <= i; k++)
{
System.out.print((char) (alphabet + k) + " ");
}
System.out.println();
}
}
}
```

Output:

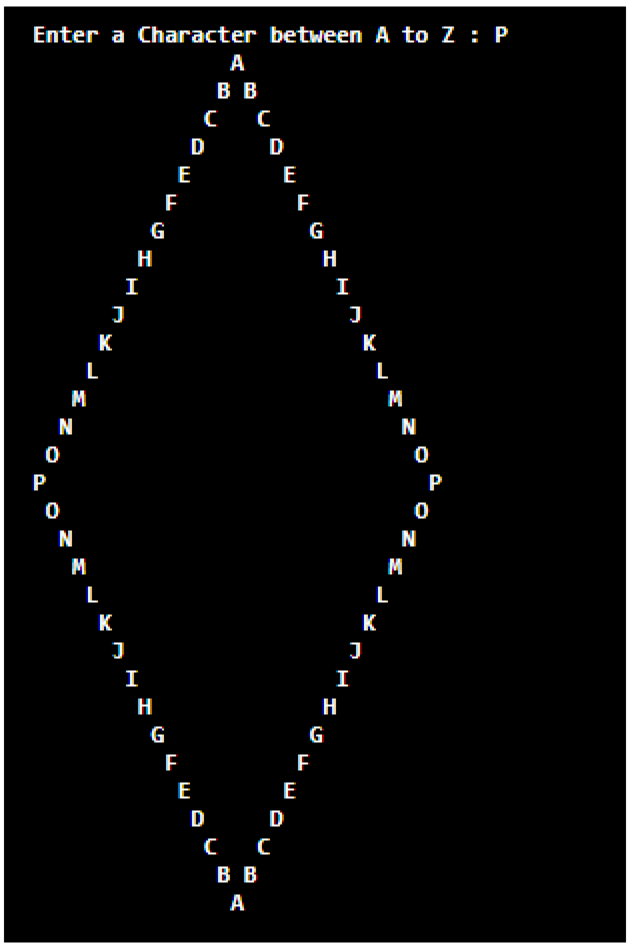


```
        A
       A B
      A B C
     A B C D
    A B C D E
   A B C D E F
  A B C D E F G
 A B C D E F G H
A B C D E F G H I
```

5. Diamond Character Pattern

```
import java.util.Scanner;
public class DiamondCharacterPattern
{
public static void main(String[] args)
{
char[] alphabet = { 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z' };
int alphabet _number = 0;
String[] diamond = new String[26]; // array of strings
System.out.print("Enter a Character between A to Z : ");
Scanner reader = new Scanner(System.in);
try
{
char user_ alphabet = reader.next("[A-Z]").charAt(0);
// search for letter number in the array letter
for (int i = 0; i < alphabet.length; i++)
{
if (letter[i] == user_ alphabet)
{
alphabet _number = i;
break;
}
}
//construct diamond
for (int i = 0; i <= alphabet _number; i++)
{
diamond[i] = "";
//add initial spaces
for (int j = 0; j < alphabet _number - i; j++)
{
diamond[i] += " ";
}
// add alphabet
diamond[i] += alphabet
//add space between letters
if (alphabet[i] != 'A')
{
for (int j = 0; j < 2 * i - 1; j++)
{
diamond[i] += " ";
}
// add alphabet
diamond[i] += alphabet[i];
}
// Draw the first part of the diamond
System.out.println(diamond[i]);
}
for (int i = alphabet _number - 1; i >= 0; i--)
{
// Draw the second part of the diamond
// prints the diamondArray in the reverse order
System.out.println(diamond[i]);
}
}
catch (Exception e)
{
e.printStackTrace();
}
finally
{
reader.close();
}
}
}
```

Output:



```
Enter a Character between A to Z : P
               A
              B B
             C   C
            D     D
           E       E
          F         F
         G           G
        H             H
       I               I
      J                 J
     K                   K
    L                     L
   M                       M
  N                         N
 O                           O
P                             P
 O                           O
  N                         N
   M                       M
    L                     L
     K                   K
      J                 J
       I               I
        H             H
         G           G
          F         F
           E       E
            D     D
             C   C
              B B
               A
```