**Day-01**

**27-02-2025**

**==============**

**continue statement:**

**=============**

-> "continue" is a keyword.

-> continue statement is one of the jump statement/transfer statement like break statement.

-> continue can use to skip the selected iteration and continue with remaining iterations.

Syntax:

continue;

Ex:

Mp3 song: 4 min

0 to 239

got a call at 120s

song ==> pause

after the call

song ==> resume

Ex: Movie ==> TV

180 min

0 to 179

120min ==> power cut for 2 min

// A program to print numbers from 1 to 10

class ContinueStatement{

public static void main(String[] args)

{

for(int i = 1; i <= 10; i = i + 1)

{

if(i == 7){

continue;

}

System.out.println(i);

}

}

}

**Q: What is the difference between break and continue?**

**====================================**

1) break is a keyword and continue is also a keyword.

2) break can be used within "switch" and "loops"

whereas the continue can be used within the loops only.

3) break will terminate the loop or block immediately

whereas the continue can stop the particular iteration and continue with remaining.

**// Prime numbers from the given range:**

**===========================**

import java.util.Scanner;

class PrimeNumbersFromRange{

public static void primeNumbers(int x, int y)

{

for(int i = x; i <= y; i++)

{

boolean flag = true;

if(i < 2){

continue;

}

for(int j = 2; j <= Math.sqrt(i);j++)

{

if(i % j == 0)

{

flag = false;

break;

}

}

if(flag == true){

System.out.print(i+"\t");

}

}

}

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

System.out.println("Enter start value:");

int x = s.nextInt();

System.out.println("Enter last value:");

int y = s.nextInt();

primeNumbers(x,y);

}

}

**Patterns:**

**======**

-> Patterns are possible to define with nested loops.

-> There are three different types of patterns:

1) Star Patterns

2) Number Patterns

3) Alphabet Patterns

\* 1 a

\* \* 1 2 a b

\* \* \* 1 2 3 a b c

**Star Patterns:**

**==========**

-> also called as "asterisk patterns".

Problem-1: Square Pattern

------------------------------

// WAP TO TAKE NUMBER OF ROWS AS AN INPUT AND PRINT SQUARE PATTERN USING '\*' BY CONSIDERING NUMBER OF ROWS AS INPUT VALUE.

import java.util.Scanner;

class SquarePattern{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter number of rows:");

int rows = scan.nextInt();

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

for(int c = 1; c <= rows; c++)

{

System.out.print("\*" + " ");

}

System.out.println();

}

}

}

Assignment:

===========

1) WAP TO PRINT ALL PERFECT NUMBERS FROM THE GIVEN RANGE.

2) WAP TO PRINT RECTANGLE PATTERN USING '\*'.

**Day-02**

**28-02-2025**

**===============**

**/\* WAP TO PRINT ALL PERFECT NUMBERS FROM THE GIVEN RANGE**

**6 ==> 1, 2, 3, 6**

**1 + 2 + 3 == 6**

**\*/**

import java.util.Scanner;

class PerfectNumbersFromRange{

public static void main(String args[])

{

Scanner s = new Scanner(System.in);

System.out.println("Enter the lower value of the range:");

int lower = s.nextInt();

System.out.println("Enter the upper value of the range:");

int upper = s.nextInt();

int num = 6;

for(int i = lower; i <= upper; i++)

{

int sum = 0;

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

{

if(i % j == 0){

//sum = sum + j;

sum += j;

}

}

if(sum == i){

System.out.print(i+"\t");

}

}

System.out.println();

}

}

**// Rectangle Pattern**

**==============**

import java.util.Scanner;

class RectanglePattern{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter the number of rows:");

int rows = scan.nextInt();

System.out.println("Enter the number of columns:");

int cols = scan.nextInt();

for(int r = 1; r <= rows; r++)

{

for(int c = 1; c <= cols; c++)

{

System.out.print("\*" + " ");

}

System.out.println();

}

}

}

**/\* WAP TO ACCEPT THE NUMBER OF LINES FOR THE RIGHT-FACED RIGHT-ANGLED TRIANGLE (Right-faced Pyramid).**

**AND PRINT THAT PATTERN\*/**

import java.util.Scanner;

class RightFacedRightAngleTriangle{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter number of lines:");

int lines = scan.nextInt();

for(int r = 1;r <= lines; r++)

{

for(int c = 1; c <= r; c++){

System.out.print("\* ");

}

System.out.println();

}

}

}

**// left Faced Right-angled Triangle**

import java.util.Scanner;

class LeftFacedRightAngledPattern{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter number of lines:");

int lines = scan.nextInt();

for(int r = 1;r <= lines; r++)

{

for(int s = 1; s <= 2 \*(lines - r); s++)

{

System.out.print(" ");

}

for(int c = 1; c <= r; c++)

{

System.out.print("\*"+" ");

}

System.out.println();

}

}

}

Assignment:

===========

Q-1: WAP TO ACCEPT THE NUMBER OF LINES FOR INVERTED RIGHT-FACED RIGHT ANGLED TRIANGLE.

AND PRINT THAT PATTERN WITH STAR.

**Day-03**

**01-03-2025**

**================**

**/\* WAP TO PRINT THE HALLOW SQUARE \*/**

import java.util.Scanner;

class HallowSquare{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter the number of rows:");

int rows = scan.nextInt();

// logic for hallow square

for(int r = 1; r <= rows; r++)

{

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

if(r == 1 || r == rows || c == 1 || c == rows){

System.out.print("\* ");

}

else{

System.out.print(" ");

}

}

System.out.println();

}

}

}

**// WAP TO PRINT THE PATTERN OF LEFT DIAGNOAL OF THE SQUARE.**

import java.util.Scanner;

class LeftDiagonal{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter the total number of rows:");

int rows = scan.nextInt();

for(int r = 1; r <= rows; r++)

{

for(int c = 1; c <= rows; c++)

{

if(r == c){

System.out.print("\* ");

}

else{

System.out.print(" ");

}

}

System.out.println();

}

}

}

**// WAP TO PRINT THE LEFT DIAGONAL FOR RECTANGLE**

class RectangleLeftDiagonal{

    public static void main(String[] args)

    {

        int rows = 5;

        int cols = 10;

        for(int r = 1; r <= rows; r++)

        {

            for(int c = 1; c <= cols; c++)

            {

                if(r == c){

                    System.out.print("\* ");

                }

                else{

                    System.out.print("  ");

                }

            }

            System.out.println();

        }

    }

}

Assignment:

===========

1) WAP TO PRINT THE HALLOW RECTANGULAR PATTERN.

2) WAP TO PRINT THE RIGHT DIAGONAL OF THE SQUARE.

**Day-04**

**03-03-2025**

**==============**

**Pattern-01:**

**--------------**

import java.util.Scanner;

class TrianglePattern{

public static void main(String[] args){

Scanner scan = new Scanner(System.in);

System.out.println("Enter number of rows:");

int n = scan.nextInt();

for(int i = 1; i <= n; i++)

{

for(int j = 1; j <= (n - i); j++)

{

System.out.print(" ");

}

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

System.out.print("\* ");

}

System.out.println();

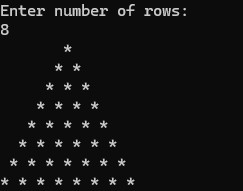
}

}

}

Output:

---------



**Pattern-2:**

**------------**

import java.util.Scanner;

class TriangleWithOddPattern{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter number of rows:");

int n = scan.nextInt();

for(int i = 1; i <= n; i++)

{

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

{

System.out.print(" ");

}

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

{

System.out.print("\* ");

}

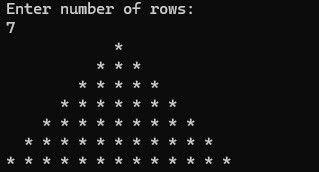
System.out.println();

}

}

}

Output:



**Pattern-3:**

**-------------**

class CrossPattern{

public static void main(String[] args)

{

int n = 6;

for(int i = 1;i <= n; i++)

{

for(int j = 1; j <= n; j++)

{

if(i == j || (i + j) == (n+1)){

System.out.print("\* ");

}

else{

System.out.print(" ");

}

}

System.out.println();

}

}

}

Output:

---------



**Patter-4:**

-----------

class InvertedRightAngledTriangle{

public static void main(String[] args)

{

int n = 10;

for(int i = n; i >= 1; i--)

{

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

{

System.out.print(" ");

}

for(int k = 1; k <= i; k++)

{

System.out.print("\* ");

}

System.out.println();

}

}

}

Output:

----------



**Pattern-5:**

**-------------**

class InvertedTriangle{

public static void main(String[] args)

{

int n = 10;

for(int i = n; i >= 1; i--)

{

for(int j = 1; j <= (n-i); j++)

{

System.out.print(" ");

}

for(int k = 1; k <= i; k++)

{

System.out.print("\* ");

}

System.out.println();

}

}

}

Output:

--------

