Sum of digits:

Add all i/p values

3 + 4 + 5 = 12

```
int n = Sc.nextInt();
int res = 0, rem;
while (n != 0) {
    rem = n % 10;
    res = res + rem;
    n = n / 10;
}
System.out.println(res);
```

O/p: I/p 1 2 3
→ 6

Palindrome

If number is same as rev value then it is palindrome

121 → Palindrome 153 → Not

```
int n = sc.nextInt();
int res = 0, rem, temp = n;
while (n != 0) {
    rem = n % 10;
    res = (res * 10) + rem;
    n = n / 10;
}
if (temp == res) {
    System.out.print("Palindrome");
}
else {
    System.out.print(" Not Palindrome");
}
```

DAY-3

Pattern Printing:

How to approach

→ No. of lines = No of rows = No. of times outer loop will be run.

→ Identify for every row no how many cols are there, types of elements in columns.

→ What do we need to print.

① Half pyramid

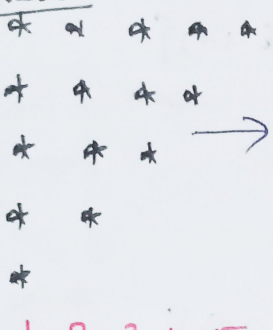
col: 1 / 1 2 / 1 2 3 / 1 2 3 4

```
for (int row = 1; row <= 4; row++) {
    for (int col = 1; col <= row; col++) {
        System.out.print("*");
    }
    System.out.println();
}
```

if → col instead of '*'

row →
```
1
2 2
3 3 3
4 4 4 4
```

② Inverted



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

```
for (int row = 5; row >= 1; row--) {
    for (int col = 1; col <= row; col++) {
        System.out.print("*");
    }
    System.out.println();
}
```

col → row

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

Rows & Columns are equal

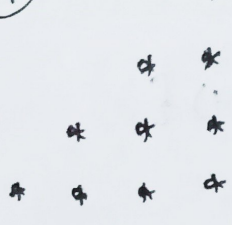
```
for (int row = 1; row <= 4; row++) {
    for (int col = 1; col <= 4; col++) {
        System.out.print("*");
    }
    System.out.println();
}
```

col → row

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

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

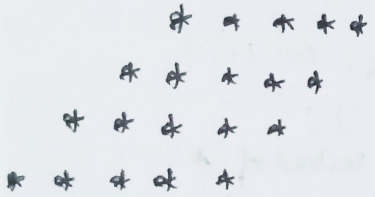
③ Inverted Half Pyramid 180°



We need to add spaces

```
for (int row = 1; row <= 4; row++) {
    for (int space = 1; space <= 4-row; space++) {
        System.out.print(" ");
    }
    for (int col = 1; col <= row; col++) {
        System.out.print("* ");
    }
    System.out.println();
}
```

⑤ Rhombus



In prev col<=row it stop upto row value but Here col upto 5. So in first line, we have upto 5 stars.

In this rows have 4 & col has 5 Rs to prev pattern

```
for(int row = 1; row<= 4; row++){
    for(int space = 1; space<= 4-row ; space++) {
        System.out.print(" ");
    }
    for(int col = 1; col<= 5 ; col++){
        System.out.print("*");
    }
    System.out.println();
}
```

⑥

n=4

```
for (int row=0; row< 2*n ; row++) {
    int totalColInRow = row>n ? 2*n-row : row;
    for(int col = 0; col< totalColInRow ; col++) {
        System.out.print("*");
    }
    System.out.println();
}
```

⑦ Pyramid

```
for(int row = 4; row>0 ; row--) {
    for(int space=1; space<= 5-row ; space++) {
        System.out.print(" ");
    }
    for(int col=1; col<= 2*row-1 ; col++) {
        System.out.print("*");
    }
    System.out.println();
}
```

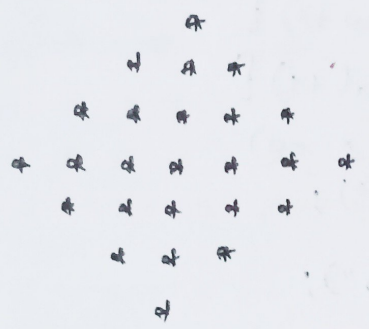
row = 0 ; row <= 5 ; row++

Hollow pyramid pattern

only add if condition in for (col) condition

```
if (col==1 || col==2*row-1 || row==n)
    System.out.print("*");
```

⑧ Diamond Pattern



Similar to prev pattern

Only changes in first line for lower part

```
int n = 4;
for(int row = 1; row <= n; row++){
  for(int space = 0; space <= n-row; space++){
    System.out.print(" ");
  }
  for(int col = 1; col <= 2*row-1; col++){
    System.out.print("* ");
  }
  System.out.println();
}
for(int row = n-1; row >= 1; row--){
  for(int space = 0; space <= n-row; space++){
    System.out.print(" ");
  }
  for(int col = 1; col <= 2*row-1; col++){
    System.out.print("* ");
  }
  System.out.println();
}
```

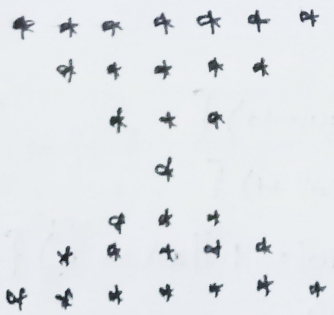
upper half

lower half

We can also solve sandglass pattern using this

Lower half then upper half

→ First decrement then increment



⑨ Floyd's Triangular

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

Each row increase req of no. Outer loop controls the no. of rows, while inner loop handles printing the no. in each row

```
int counter = 1;
for(int row = 1; row <= 4; row++){
  for(int col = 1; col <= row; col++){
    System.out.print(counter + " ");
    counter = counter + 1;
  }
  System.out.println();
}
```

(10) 0 to 1 triangular pattern

```
(1,1) 1
(2,1) 0 1
(3,1) 1 0 1
(4,1) 0 1 0 1
```

Sum of row & col is even then "1" if odd it will be "0".

```
for (int row = 1; row <= 5; row++) {
    for (int col = 1; col <= row; col++) {
        if (~~row+col~~ (row + col) % 2 == 0)
            System.out.print("1");
        else
            System.out.print("0");
    }
    System.out.println();
}
```

(11) Fibonacci Series:

```
0
1 1
2 3 5
8 13 21 34
```

In fibonacci 3rd no. of. will be sum of two prev numbers & after every step first no will be 2nd & 2nd will be third

```
int a = 0, b = 1;
for (int row = 1; row <= 4; row++) {
    for (int col = 1; col <= row; col++) {
        System.out.print(a + " ");
        int sum = a + b;
        a = b;
        b = sum;
    }
    System.out.println();
}
```

(12) Square Pattern

```
(1,1) * * * * (1,n)
*       *
*       *
(n,1) * * * * (n,n)
```

```
int n = 4;
for (int row = 1; row <= n; row++) {
    for (int col = 1; col <= row/n; col++) {
        if (row == 1 || row == n || col == 1 || col == row/n) {
            System.out.print("* ");
        }
        else {
            System.out.print("  ");
        }
    }
    System.out.println();
}
```

Hollow rightangled Triangle

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

(13)

```
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
```

Odd

```
1  3  5  7  9
11 13 15 17 19
21 23 25 27 29
31 33 35 37 39
41 43 45 47 49
```

Even

```
2  4  6  8  10
12 14 16 18 20
22 24 26 28 30
32 34 36 38 40
42 44 46 48 50
```

```
Count = 2
int count = 1, n = 5;
for(int row = 1; row <= n; row++){
  for(int col = 1; col <= n; col++){
    if(count < 10){
      System.out.print("0");
    }
    System.out.print(count + " ");
    Count++; Count += 2;
  } Count += 2;
  System.out.println();
}
```

Only changes in count value for continuous value
Count++ just increment
To skip value count += 2

(14)

```
1  2  3  4  5
2  4  6  8  10
3  6  9  12 15
4  8  12 16 20
5  10 15 20 25
```

```
int n=5, count = 1;
for(int i = 1; i <= n; i++){
  for(int j = 1; j <= n; j++){
    if(count (i*j) < 10){
      System.out.print("0");
    }
    System.out.print((i+j) + " ");
    count++;
  }
  System.out.println();
```

For Hollow triangle just add if condition to print stars("*")

(15) Equilateral Triangle

```
Int n = 4;
for(int row = 1; row <= n; row++){
  for(Int space = 1; space <= (n - row); space++){
    System.out.print(" ");
  }
  for(int col = 1; col <= row; col++){
    if(row == 1 || row == n || col == 1 || col == row)
    System.out.print("* ");
  }
  System.out.println();
}
```

Hollow Equi Triangle