

1. Write a Java program to take a positive integer  $N$  (where  $N > 0$ ) as user input and print the **first  $N$  prime numbers starting from 2**. Your code should check all the positive integers starting from 2 and determine whether they are prime or not until  $N$  prime numbers are found.

**Sample Input 1:**

5

**Sample Output 1:**

2

3

5

7

11

**Sample Input 2:**

7

**Sample Output 2:**

2

3

5

7

11

13

17

2. Write a Java code of a program that reads the value of  $N$  (where  $N > 0$ ) from the user and calculates the value of  $y$  if the expression of  $y$  is as follows:

$$y = - (1) - (1 + 2) - (1 + 2 + 3) - \dots - (1 + 2 + 3 + \dots + N)$$

**Sample Input:**

The value of  $N$ : 2

**Sample Output:**

The value of  $y$ : -4

**Sample Input:**

The value of  $N$ : 4

**Sample Output:**

The value of  $y$ : -20

3. Write a Java program that will keep taking even positive integer numbers as inputs from the user and print the number of divisors of those numbers until it gets an odd number and then stops.

**Sample Input & Output:** (The purple numbers are input)

Enter Number: 44

44 has 6 divisors

Enter Number: 30

30 has 8 divisors

Enter Number: 8

8 has 4 divisors

Enter Number: 4

4 has 3 divisors

Enter Number: 6

6 has 4 divisors

Enter Number: 20

20 has 6 divisors

Enter Number: 24

24 has 8 divisors

Enter Number: 5

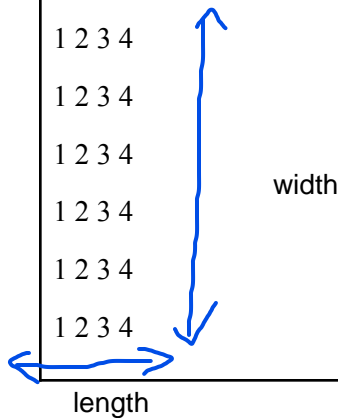
4. Read an integer N that is the number of test cases that follow. Each test case contains two integers X and Y. Print one output line for each test case that the sum of Y odd numbers from X including it if is the case. For example:

For the input 4 5, the output must be 45, that is:  $5 + 7 + 9 + 11 + 13$

For the input 7 4, the output must be 40, that is:  $7 + 9 + 11 + 13$

Sample Input	Sample Output
2 4 3 11 2	21 24
Explanation: Here, the 2 means there are two test cases. For each test case you have to take two inputs (X, Y) and print the sum of Y odd numbers starting from X.	

5. Take the length and width of a **rectangle** from the user and create the rectangle according to the output below. Your output should match the specified output.

Sample Input #1	Sample Input #2
4 6	3 5
<b>Output</b> 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 	<b>Output</b> 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3

6. Take the height of a **right-justified right triangle** from the user and create the triangle according to the output below. Your output should match the specified output.

Sample Input #1	Sample Input #2
4	3
<b>Output</b> 1 1 2 1 2 3 1 2 3 4	<b>Output</b> 1 1 2 1 2 3

7. Take the height of an **isosceles triangle** from the user and create the triangle according to the output below. Your output should match the specified output.

Sample Input #1	Sample Input #2
4	3

Output	Output
1	1
1 2 3	1 2 3
1 2 3 4 5	1 2 3 4 5
1 2 3 4 5 6 7	

8. Write a Java program that will ask for a range (a starting number and an ending number) from the user and print all the Armstrong numbers between that range.

*[Armstrong Number: An Armstrong number is a number whose sum of digits raised to the power the number of digits equals to that number.*

*For example, 371 is an Armstrong number because  $3^3 + 7^3 + 1^3 = 371$ , here the total number of digits in 371 is 3 ]*

**Sample Input 1:**

Start: 300

End: 500

**Sample Output 1:**

Armstrong numbers:

370 ->

371

407

**Sample Input 2:**

Start: 100

End: 200

**Sample Output 2:**

Armstrong numbers:

153

9. Trace the following code, create a tracing table and write the outputs.

1	<code>public class T1{</code>
2	<code>    public static void main(String args[]){</code>
3	<code>        int x = 0, y = 0;</code>
4	<code>        int sum = 0;</code>
5	<code>        while (x &lt; 4){</code>
6	<code>            y = x - 3;</code>
7	<code>            while (y &lt; 3){</code>
8	<code>                sum = (sum % 3) + x - y * 3 ;</code>
9	<code>                System.out.println(sum);</code>
10	<code>                y = y + 1;</code>
11	<code>            }</code>
12	<code>            if (x &gt; 5){</code>
13	<code>                x++;</code>
14	<code>            }</code>
15	<code>            else{</code>
16	<code>                x += 2;</code>
17	<code>            }</code>
18	<code>        }</code>
19	<code>    }</code>
20	<code>}</code>

10. Trace the following code, create a tracing table and write the outputs.

1	<code>public class T2 {</code>
2	<code>    public static void main(String args[]) {</code>
3	<code>        int x = 0, i = 0, sum = 0;</code>
4	<code>        i = 1;</code>
5	<code>        x = 2;</code>
6	<code>        sum = 0;</code>
7	<code>        while (i &lt; 20){</code>
8	<code>            x = x + i;</code>
9	<code>            sum = sum + x + 1;</code>
10	<code>            System.out.println(sum);</code>
11	<code>            if (x &gt; 5){</code>
12	<code>                i += 2;</code>
13	<code>            }</code>
14	<code>            else {</code>
15	<code>                i += 3;</code>
16	<code>            }</code>
17	<code>        }</code>
18	<code>        sum = sum + i;</code>
19	<code>        System.out.println(sum);</code>
20	<code>    }</code>
21	<code>}</code>

11. Trace the following code, create a tracing table and write the outputs.

1	<code>public class T3</code>
2	<code>{</code>
3	<code>    public static void main(String args[])</code>
4	<code>    {</code>
5	<code>        int x = 0, y = 0;</code>
6	<code>        int sum = 0;</code>
7	<code>        while (x &lt; 10){</code>
8	<code>            y = x - 3;</code>
9	<code>            y = 40;</code>
10	<code>            while (y &gt; 22){</code>
11	<code>                if ((sum &gt; 30) &amp;&amp; (sum &lt; 40)){</code>
12	<code>                    sum = sum + x * 2 ;</code>
13	<code>                }</code>
14	<code>                else if ((sum &gt; 40) &amp;&amp; (sum &lt; 50)){</code>
15	<code>                    sum = sum + x * 3;</code>
16	<code>                }</code>
17	<code>                else {</code>
18	<code>                    sum = sum + 23;</code>
19	<code>                }</code>
20	<code>            System.out.println(sum) ;</code>
21	<code>            y = y - 10;</code>
22	<code>        }</code>
23	<code>        x += 2;</code>
24	<code>    }</code>
25	<code>}</code>
26	<code>}</code>

12. Trace the following code, create a tracing table and write the outputs

1	<code>public class T4</code>
2	<code>{</code>
3	<code>    public static void main(String args[])</code>
4	<code>    {</code>
5	<code>        int x = 0, p = 0, sum = 0;</code>
6	<code>        p = 1;</code>
7	<code>        x = 2;</code>
8	<code>        double q;</code>
9	<code>        sum = 0;</code>
10	<code>        while (p &lt; 12){</code>
11	<code>            q = x + p - (sum+7/3)/3.0%2 ;</code>
12	<code>            sum = sum + (++x) + (int)q;</code>
13	<code>            System.out.println(sum++);</code>
14	<code>            if (x &gt; 5){</code>
15	<code>                p += 4/2;</code>
16	<code>            }</code>
17	<code>            else {</code>
18	<code>                p += 3%1;</code>
19	<code>            }</code>
20	<code>        }</code>
21	<code>        sum = sum + p;</code>
22	<code>        System.out.println(sum);</code>
23	<code>    }</code>
24	<code>}</code>