1. **N - S - C**

Take two numbers from user A and B respectively.

And print

- n
- sqaure of n (n2)
- cube of n (n3)
- where A < n < B

In other words **print the NUMBER, SQUARE OF NUMBER and CUBE OF NUMBER for EVERY NUMBER from A to B (Both Exclusive)**. See the sample inputs for a better understanding.

- It is guaranteed that A < B and B−A ≥ 2

Input Format:
Asingle contains two space separated integers.

Output Format:
Print the output according to description.

Sample I/O:

Input1:
2 7
Output1:
3 9 27
4 16 64
5 25 125
6 36 216

Input2:
13 21
Output2:
14 196 2744
15 225 3375
16 256 4096
17 289 4913
18 324 5832
19 361 6859
20 400 8000

Input3:
79 81
Output3:
80 6400 512000

2. **Prime**

Write a program to check the given number is Prime or not?

Input Format:
A single line input containing one integer.

Output Format:
Print output according to the discription.

Sample I/O:

Input 1:
5

Output 1:
Prime

Input 2:
6

Output 2:
Not Prime

3. Sum of First and Last digits

Write a program to find the Sum First and Last Digits of a given number

Input Format:
A single line contains an integer N.

Output Format:
Display the Result of Sum of First and Last digits

Sample I/O:

Input 1:
123

Output 1:
4

Input 2:
1234

Output 2:
5

4. N to 1

You will be given a number N. Print all the numbers from N to 1 (Both inclusive).

Input Format:
A single line contains an integer N.

Output Format:
Display the all the numbers from N to 1

Sample I/O:

Input 1:
20

Output 1:
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Input 2:
147

Output 2:
147 146 145 144 143 142 141 140 139 138 137 136 135 134 133 132 131 130 129 128 127 126 125 124 123 122 121 120 119 118 117 116 115 114 113 112 111 110 109 108 107 106 105 104 103 102 101 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

5. Multiplication Table - 3

You'll be given three numbers **N, A and B.** Print the multiplication table of number N from A TO B (Both inclusive). Refer to the sample I/O for more clarity.

- It is guaranteed that $A<B$

Input Format:
A single line contains an three integers N, A, B.

Output Format:
Print the multiplication table.

Sample I/O:

Input 1:
7 14 21

Output 1:
7 x 14 = 98
7 x 15 = 105
7 x 16 = 112
7 x 17 = 119
7 x 18 = 126
7 x 19 = 133
7 x 20 = 140
7 x 21 = 147

Input 2:
291 421 435

Output 2:
291 x 421 = 122511
291 x 422 = 122802
291 x 423 = 123093

291 x 424 = 123384
291 x 425 = 123675
291 x 426 = 123966
291 x 427 = 124257
291 x 428 = 124548
291 x 429 = 124839
291 x 430 = 125130
291 x 431 = 125421
291 x 432 = 125712
291 x 433 = 126003
291 x 434 = 126294
291 x 435 = 126585

Input 3:
16 71 77

Output 3:
16 x 71 = 1136
16 x 72 = 1152
16 x 73 = 1168
16 x 74 = 1184
16 x 75 = 1200
16 x 76 = 1216
16 x 77 = 1232

Input 4:
19 12 15

Output 4:
19 x 12 = 228
19 x 13 = 247
19 x 14 = 266
19 x 15 = 285

6. A to B

You will be given two values A and B respectively. Print the numbers from A to B (Both inclusive)

- It is guaranteed that A≤B

Input Format:
A single line input containing two space-separated integers.

Output Format:
Display output according to the discription.

Sample I/O

Input 1:
10 20

Output 1:
10 11 12 13 14 15 16 17 18 19 20

Input 2:
23 31

Output 2:
23 24 25 26 27 28 29 30 31

Input 3:
1236 1347

Output 3:
1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252
1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266 1267 1268 1269
1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286
1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300 1301 1302 1303
1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315 1316 1317 1318 1319 1320 1321
1322 1323 1324 1325 1326 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338
1339 1340 1341 1342 1343 1344 1345 1346 1347

Input 4:
231 231

Output 4:
231

1 N-S-C

```java
import java.util.*;
public class NSC {
    public static void main(String[] args) {
        Scanner read = new Scanner(System.in);
        /**
         * The idea is to generate all the numbers from A + 1 to B
- 1
         * And print the number, square and cube while doing so
         */
        int A = read.nextInt();
        int B = read.nextInt();
        for (int i = A + 1; i < B; i++) {
            System.out.println(i + " " + i * i + " " + i * i * i);
        }
    }
}
```

2 Prime

```java
import java.util.*;
public class Prime {
    public static void main(String[] args) {
        Scanner read = new Scanner(System.in);
        /**
         * We know that we can call a number a Prime, if and only
if
         * it contains exactly 2 factors (divisors)
         * So we will count the given numbers divisors to find
         * if it's a prime or not
         */
        int n = read.nextInt();
        int factorCount = 0;
        for (int i = 1; i <= n; i++) {
            if (n % i == 0) factorCount++;
        }
        // Checking if factorCount is exactly 2
        if (factorCount == 2) System.out.print("Prime");
        else System.out.print("Not Prime");
    }
}
```

3 Sum of first and last digits

```java
import java.util.Scanner;
public class SumOfFirstandLastDigits {
    public static void main(String[] args) {
        /**
         * We can get the last digit of any number easily
         * by performing %10 on that number
         * for instance 1234%10 -> 4
         * To get the first digit either we can write
         * code or take help of log10() method of Math class
         * log10(x) -> is the nth power of 10 that
         * we can define x in.
         * For example
         * log10(100) is exactly 2. As 100 = 10^2
         * log10(10000) is exactly 4. As 10000 = 10^4
         * For instance if we have a number 987
         * log10(987) is 2.999... which truncates to 2 (if we stor
e it in integer form)
         * Now we can apply 987/pow(10, 2) = 987/100 to get 9
         * which is the first digit of 987
         */
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int p = (int)Math.log10(n);
        int lastDigit = n % 10;
        int firstDigit = n / (int)Math.pow(10, p);
        // Printing their sum
        System.out.print(firstDigit + lastDigit);
    }
}
```

4 N to 1

```java
import java.util.Scanner;
public class Nto1 {
    public static void main(String[] args) {
        // Just using a for loop to go from N to 1
        // by decrementing every time
        Scanner read = new Scanner(System.in);
        int n = read.nextInt();
        for (int i = n; i >= 1; i--) {
            System.out.print(i + " ");
        }
    }
}
```

5 Multiplication Table 3

```java
import java.util.*;
public class MultiplicationTable3 {
    public static void main(String[] args) {
        Scanner read = new Scanner(System.in);
        int number, start, stop;
        number = read.nextInt();
        start = read.nextInt();
        stop = read.nextInt();
        /**
         * The idea is to generate all the numbers from start to s
top
         * and multiply those numbers while generating with
         * give number
         */
        for (int i = start; i <= stop; i++) {
            System.out.println(number + " x " + i + " = " + (numbe
r * i));
        }
    }
}
```