Print all factors of a number

n = 20 , factors :- 1, 2, 4, 5, 10, 20

all no.'s from 1 to n which are able to divide n.

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    printAllFactors(n);
}

public static void printAllFactors(int n) {
    for (int i = 1; i <= n; i++) {
        if ( n % i == 0 ) {
            System.out.println(i);
        }
    }
}
```

$\boxed{n = 8}$

i=1, (8 % 1 == 0) ✓

i=2, (8 % 2 == 0) ✓

i=3, (8 % 3 == 0) ✗

i=4, (8 % 4 == 0) ✓

i=5, (8 % 5 == 0) ✗

i=6, (8 % 6 == 0) ✗

i=7, (8 % 7 == 0) ✗

i=8, (8 % 8 == 0) ✓

1
2
4
8

Print all unique prime factors

n = 45

↳ factors :- 1, 3, 5, 9, 15, 45

↳ prime factors :- 3, 5,

↳ unique prime factors :- 3, 5

```
for ( int i=1; i<=n; i++) {
    if( n % i == 0) {
        isPrime(i);
    }
}
```

prime factors

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    primeFactors(n); //45
}

public static void primeFactors(int n) {
    for (int i = 2; i <= n; i++) {
        if ( n % i == 0 ) {
            // now i is a factor of n
            // now check if i is prime or not
            boolean check = isPrime(i);
            if (check == true) {
                System.out.println(i);
            }
        }
    }
}
public static boolean isPrime(int n) {

    for (int i = 2; i < n; i++) {
        if ( n % i == 0 ) {
            return false;
        }
    }
    return true;

}
```

n=45

i=2, (45%2==0) ✗

i=3, (45%3==0) ✓

i=4, ✗

i=5, (45%5==0) ✓

i=6, ✗

i=7, ✗

i=8, ✗

i=9, (45%9==0) ✓

i=45

o/p

3
5

Divide n by 2 3 5 and tell steps

n = 2472

steps = 0

steps = ~~0~~
~~2~~
~~4~~
~~6~~
9

2	2472
2	1236
2	618
3	309
	103

$\boxed{n}$

```
while ( n % 2 == 0 ) {
        n /= 2;
        steps += 2;
}
while ( n % 3 == 0 ) {
        n /= 3;
        steps += 3;
}
while ( n % 5 == 0 ) {
        n /= 5;
        steps += 5;
}
```

code

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int steps = scn.nextInt();

    divideBy235(n, steps);
}
public static void divideBy235(int n, int steps) {
    while ( n % 2 == 0 ) {
        n /= 2;
        steps += 2;
    }
    while ( n % 3 == 0 ) {
        n /= 3;
        steps += 3;
    }
    while ( n % 5 == 0 ) {
        n /= 5;
        steps += 5;
    }
    System.out.println(steps);
    System.out.println(n);
}
```

n = 1260, steps = 0

steps = ~~2~~
~~4~~
~~7~~
~~10~~
15

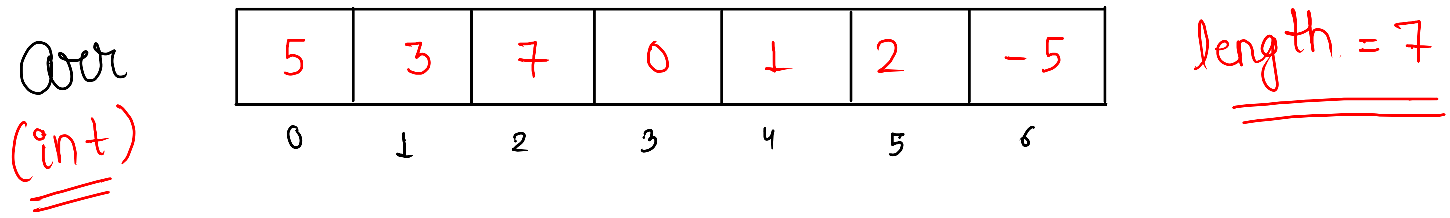
2	1260
2	630
3	315
3	105
5	35
	7

O/P

15
7

⇒ **Arrays** (M.M. Imp)

↳ Collection of similar type of data type.

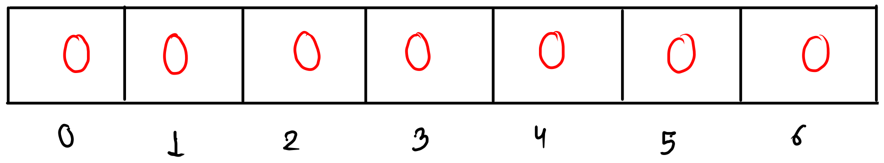


Syntax

data_type [ ] arr = new data_type[size] ;

Ex:-

int[] arr = new int[7];



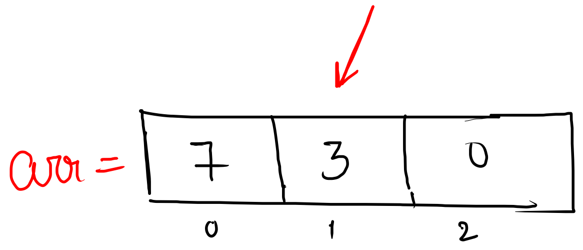
Note:- default value in an int array in Java is always zero.

```
int[] arr = new int[3];

arr[1] = 5;

arr[0] = 7;
arr[1] = 3;              // upgradation

arr[5] = 7;              // exception:- array index
                                     out of bound
```



Note:- str.length() , arr.length // 3

↳ array is static in nature

(size of array can't be changed once defined)

→ how to access any value in array

arr :-

0	1	2	3	4
5	0	5	1	3

```
int a = arr[3];

int b = arr[7];    error
```