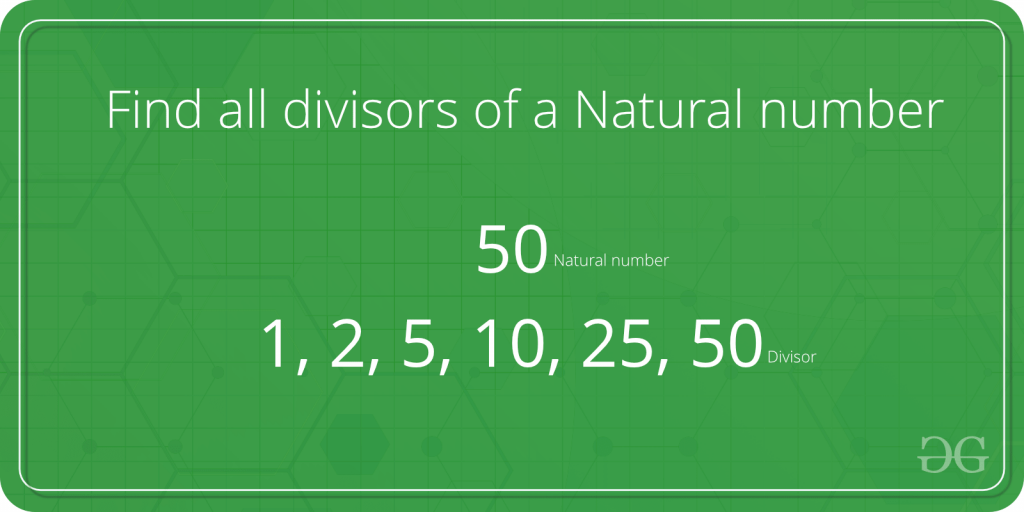
***All Divisors of a Number***

Given a natural number n, print all distinct divisors of it.



**Examples:**

**Input :** n = 10

**Output:** 1 2 5 10

**Input:**  n = 100

**Output:** 1 2 4 5 10 20 25 50 100

**Input:**  n = 125

**Output:** 1 5 25 125

A **Naive Solution** would be to iterate all the numbers from 1 to n, checking if that number divides n and printing it. Below is a program for the same:

C++Java

// Java implementation of Naive method to print all

// divisors

class Test

{

// method to print the divisors

static void printDivisors(int n)

{

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

if (n%i==0)

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

}

// Driver method

public static void main(String args[])

{

System.out.println("The divisors of 100 are: ");

printDivisors(100);;

}

}

**Output:**

The divisors of 100 are:

1 2 4 5 10 20 25 50 100

**Time Complexity :** O(n)   
**Auxiliary Space :** O(1)

**Can we improve the above solution?**   
If we look carefully, all the divisors are present in pairs. For example if n = 100, then the various pairs of divisors are: (1,100), (2,50), (4,25), (5,20), (10,10)  
Using this fact we could speed up our program significantly.   
We, however, have to be careful if there are two equal divisors as in the case of (10, 10). In such case, we’d print only one of them.

Below is an implementation for the same:

C++Java

// A Better (than Naive) Solution to find all divisors

class Test

{

// method to print the divisors

static void printDivisors(int n)

{

// Note that this loop runs till square root

for (int i=1; i<=Math.sqrt(n); i++)

{

if (n%i==0)

{

// If divisors are equal, print only one

if (n/i == i)

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

else // Otherwise print both

System.out.print(i+" " + n/i + " " );

}

}

}

// Driver method

public static void main(String args[])

{

System.out.println("The divisors of 100 are: ");

printDivisors(100);;

}

}

**Output:**

The divisors of 100 are:

1 100 2 50 4 25 5 20 10

Time Complexity: O(sqrt(n))   
Auxiliary Space : O(1)

**Printing all the divisors in sorted order:**

C++Java

import java.util.Vector;

class Test {

// method to print the divisors

static void printDivisors(int n)

{

// Vector to store half of the divisors

Vector<Integer> v = new Vector<>();

for (int i = 1; i <= Math.sqrt(n); i++) {

if (n % i == 0) {

// check if divisors are equal

if (n / i == i)

System.out.printf("%d ", i);

else {

System.out.printf("%d ", i);

// push the second divisor in the vector

v.add(n / i);

}

}

}

// The vector will be printed in reverse

for (int i = v.size() - 1; i >= 0; i--)

System.out.printf("%d ", v.get(i));

}

// Driver method

public static void main(String args[])

{

System.out.println("The divisors of 100 are: ");

printDivisors(100);

}

}

**Output:**

1 2 4 5 10 20 25 50 100

Time Complexity: O(sqrt(n))   
Auxiliary Space : O(1)