

- [Java Swap two Strings](#)
- [Java Check Anagram or Not](#)
- [Java Check Balance Parentheses](#)
- [Java Check Password Strength](#)
- [Java File Programs](#)
- [Java Read File](#)
- [Java Write to File](#)
- [Read & Display File Content](#)
- [Java Copy File](#)
- [Java Append Text to File](#)
- [Java Merge two File](#)
- [List files in Directory](#)
- [Java Delete File](#)
- [Java Miscellaneous Programs](#)
- [Generate Random Numbers](#)
- [Java Print Time & Date](#)
- [Java Get IP Address](#)
- [Java Shutdown Computer](#)
- [Java Programming Tutorial](#)
- [Java Tutorial](#)

Java Program to Get Input from User

This article is created to cover multiple programs in Java that are based on receiving inputs from user. Here are the list of programs included in this article:

- [Get integer input in Java](#)
- [Continue receiving inputs until user enters 0](#)
- [How to handle with invalid inputs in Java ?](#)
- [Get character input in Java](#)
- [Get string input in Java](#)

Get Integer Input in Java

The question is, write a Java program to ask the user to enter an integer value and print the entered value back on the output screen. The program given below is its answer. This program basically shows, how to read an integer value in Java using **Scanner** and **nextInt()**

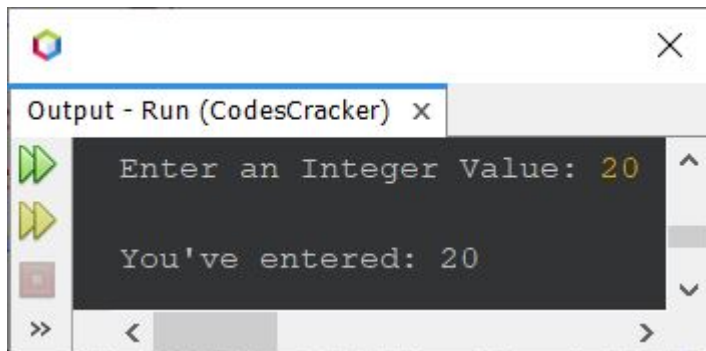
```
import java.util.Scanner;

public class CodesCracker
{
    public static void main(String[] args)
    {
        int num;
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter an Integer Value: ");
        num = scan.nextInt();

        System.out.println("\nYou've entered: " + num);
    }
}
```

The snapshot given below shows the sample run of above program, with user input **20**:



You can use following methods, to scan values of other types:

- **nextDouble()** - to read value of **double** data type
- **nextFloat()** - to read value of **float** type

- **nextLong()** - to read value of **long** type
- **nextShort()** - to read value of **short** type
- **nextByte()** - to read value of **byte** type

Continue Receiving Integer Input until User enters 0 in Java

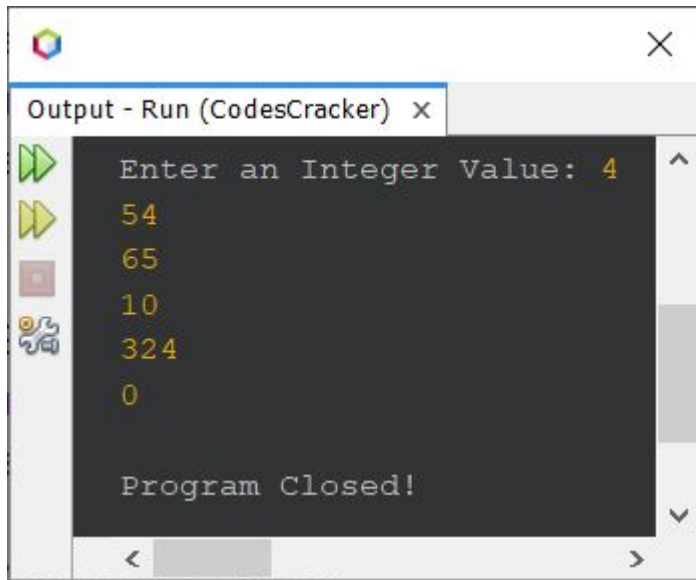
This program is created in a way, to continue receiving the inputs from user, until enters 0. You can modify this program, to use in a way, like to continue receiving inputs from user, until user enters a character 'x' or whatever you want.

```
import java.util.Scanner;

public class CodesCracker
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter an Integer Value: ");
        int num = scan.nextInt();
        while(num!=0)
            num = scan.nextInt();
        System.out.println("\nProgram Closed!");
    }
}
```

Here is its sample run with some user inputs:

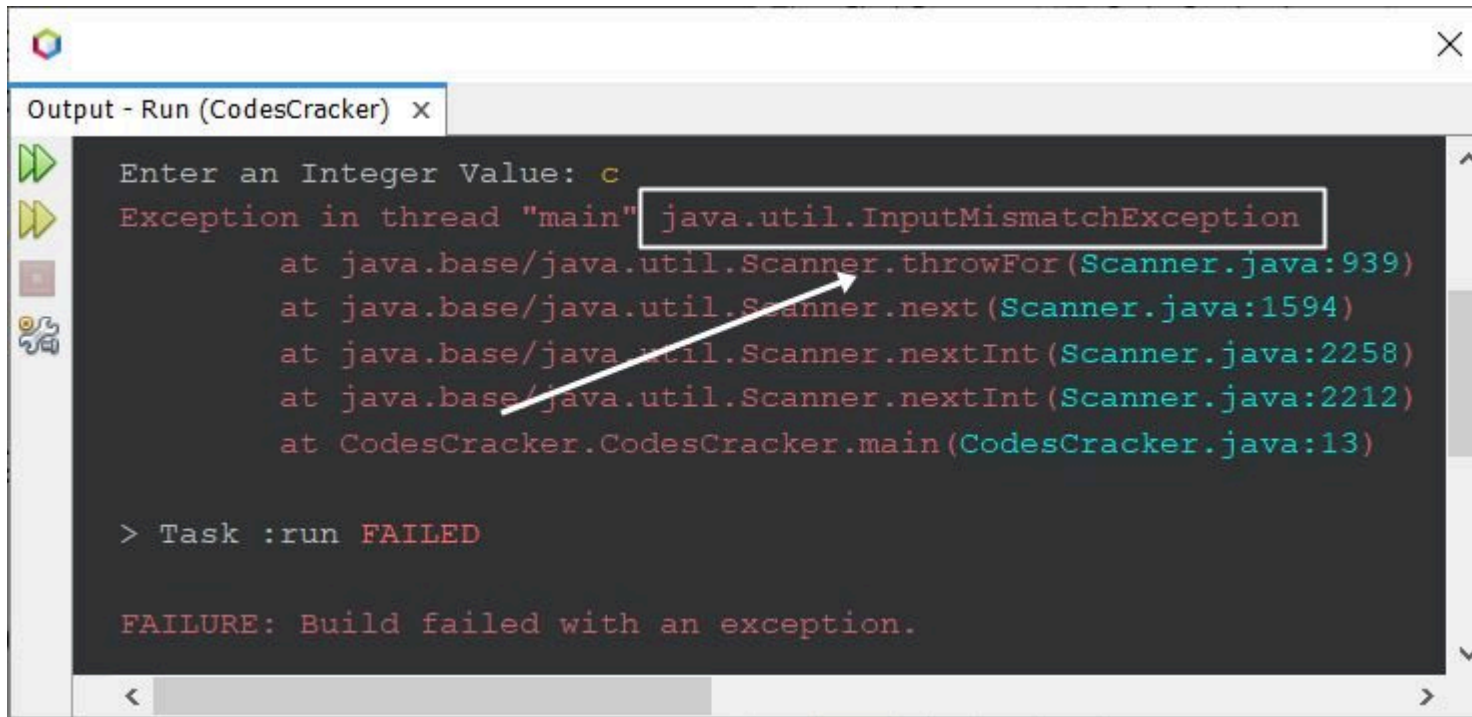


```
Output - Run (CodesCracker) x
Enter an Integer Value: 4
54
65
10
324
0
Program Closed!
```

How to Handle with Invalid Inputs in Java ?

Now the question is, what if user enters an invalid input ?

Like when we need to get integer input, but user enters some other type of value such as a floating-point input, character input, or string input. Let's check it out with first program of this article, using another sample run, but with non-integer value say **c**, a character input, this time:



```
Output - Run (CodesCracker) x

Enter an Integer Value: c
Exception in thread "main" java.util.InputMismatchException
    at java.base/java.util.Scanner.throwFor(Scanner.java:939)
    at java.base/java.util.Scanner.next(Scanner.java:1594)
    at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
    at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
    at CodesCracker.CodesCracker.main(CodesCracker.java:13)

> Task :run FAILED

FAILURE: Build failed with an exception.
```

Now we need to put the scanner statement inside the **try** block, so that, we can catch that exception using the catch block. Here is the complete version of the code, created after modifying the first program of this article. This program handles with invalid input:

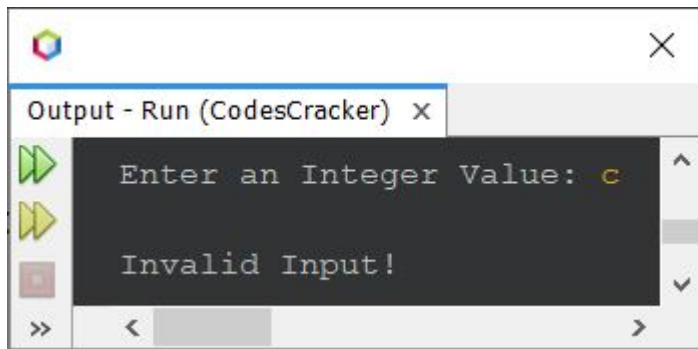
```
import java.util.Scanner;
import java.util.InputMismatchException;

public class CodesCracker
{
    public static void main(String[] args)
    {
        int num;
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter an Integer Value: ");
        try
        {
            num = scan.nextInt();
        }
    }
}
```

```
        System.out.println("\nYou've entered: " + num);
    }
    catch (InputMismatchException ime)
    {
        System.out.println("\nInvalid Input!");
    }
}
```

Here is its sample run with same user input as of previous sample run, that is **c**:



In above program, the following two statements:

```
import java.util.Scanner;
import java.util.InputMismatchException;
```

can also be replaced with a single statement given below:

```
import java.util.*;
```

Take Character Input in Java

This program is created to show you, how the character input can be received from user at run-time of the program.

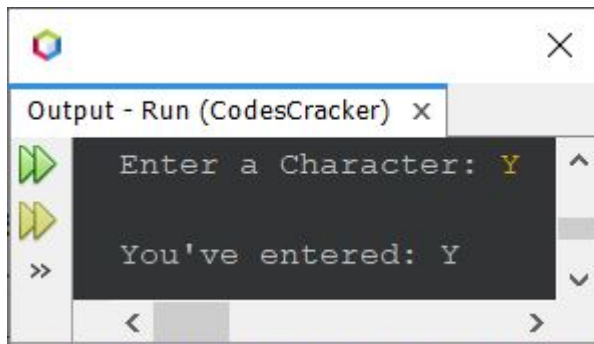
```
import java.util.Scanner;

public class CodesCracker
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);

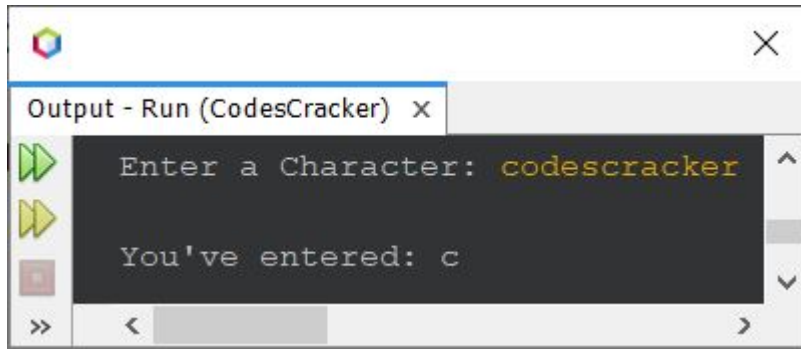
        System.out.print("Enter a Character: ");
        char ch = scan.next().charAt(0);

        System.out.println("\nYou've entered: " +ch);
    }
}
```

The sample run with user input **Y** is shown in the snapshot given below:



In above program, the **next()** method is used to receive string input, whereas the **charAt()** method is used to get the character available at any particular index specified using its parameter. Therefore, **charAt(0)** scans the very first character or the character available at 0th index of the string. Therefore, if you enter any string like **codescracker** on the sample run of above program, then the first character, that is **c** will get initialized to **ch**. Here is its sample run with user input **codescracker**:



Get String Input from User in Java

To get string input from user in Java, we have following two methods:

- `next()`
- `nextLine()`

The **`next()`** method is used to scan a single word, name, or any string without space(s). Whereas the **`nextLine()`** method is used, when we need to scan and receive the whole string with or without spaces, typed before pressing the `ENTER` key. Let's create the program for both the methods.

Get String Input in Java - Without Space

This program uses **`next()`** method to scan a word or string without spaces.

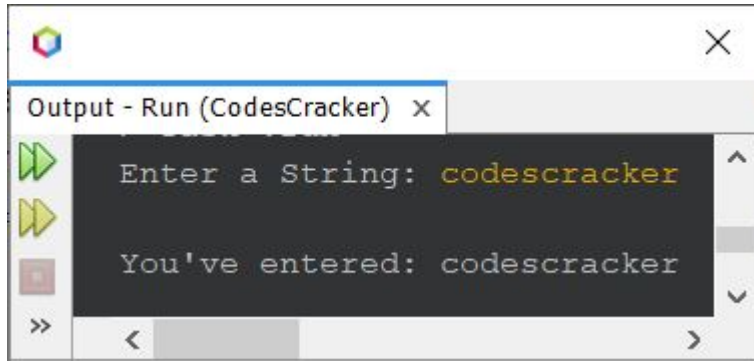
```
import java.util.Scanner;

public class CodesCracker
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
```



```
System.out.print("Enter a String: ");  
String str = scan.next();  
  
System.out.println("\nYou've entered: " +str);  
}  
}
```

Here is its sample run with user input **codescracker**:



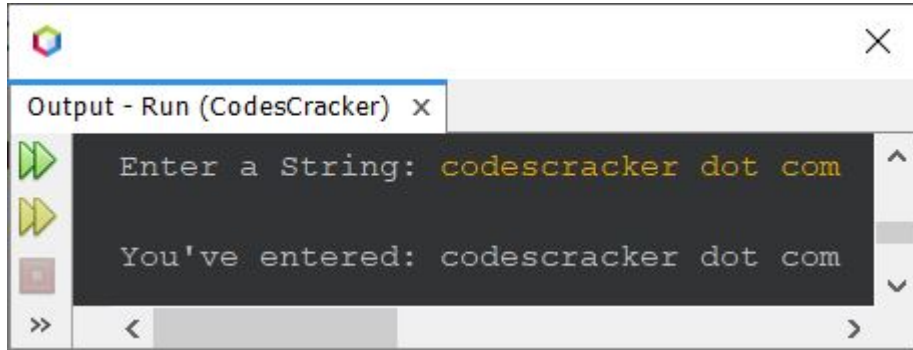
Here is another sample run with user input **codescracker dot com**:



Get String Input in Java - With Spaces

To receive the complete string with spaces, typed in a line, use **nextLine()** instead of **next()**. Rest of all the codes remains same as of previous program.

Here is its sample run, when you use **nextLine()**, while receiving the string input from user:



```
Output - Run (CodesCracker) x
Enter a String: codescracker dot com
You've entered: codescracker dot com
```

Get Multiple Inputs from User in Java

This is the last program of this article, created to show you, how you can get multiple inputs in Java language.

```
import java.util.Scanner;

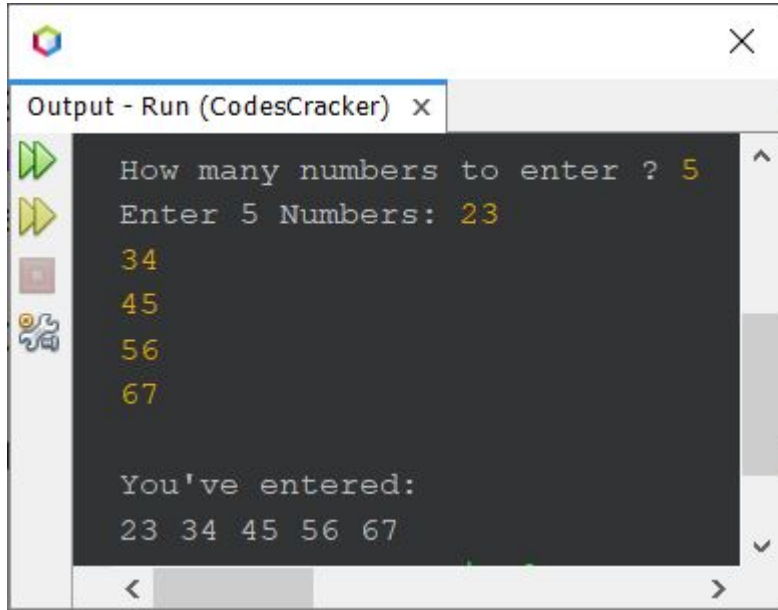
public class CodesCracker
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);

        System.out.print("How many numbers to enter ? ");
        int n = scan.nextInt();
        int[] arr = new int[n];

        System.out.print("Enter " +n+ " Numbers: ");
        for(int i=0; i<n; i++)
            arr[i] = scan.nextInt();
    }
}
```

```
        System.out.println("\nYou've entered: ");
        for(int i=0; i<n; i++)
            System.out.print(arr[i]+ " ");
    }
}
```

Here is its sample run with user input **5** as size, and **23, 34, 45, 56, 67** as five numbers:



```
Output - Run (CodesCracker) x
>> How many numbers to enter ? 5
>> Enter 5 Numbers: 23
34
45
56
67

You've entered:
23 34 45 56 67
```

Same Program in Other Languages

- [C Get Input from User](#)
- [C++ Get Input from User](#)
- [Python Get Input from User](#)

[Java Online Test](#)

[« Previous Program](#)

[Next Program »](#)