# Java exception handling

**import** java.util.Scanner;

**class** Division {

**public** **static** **void** main(String[] args) {

**int** a, b, result;

Scanner input = **new** Scanner(System.in);

System.out.println("Input two integers");

a = input.nextInt();

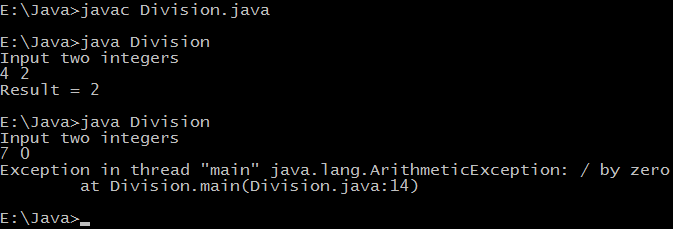
b = input.nextInt();

result = a / b;

System.out.println("Result = " + result);

}

}



## Java exception handling example

**class** Division {

**public** **static** **void** main(String[] args) {

**int** a, b, result;

Scanner input = **new** Scanner(System.in);

System.out.println("Input two integers");

a = input.nextInt();

b = input.nextInt();

*// try block*

**try** {

result = a / b;

System.out.println("Result = " + result);

}

*// catch block*

**catch** (ArithmeticException e) {

System.out.println("Exception caught: Division by zero.");

}

}

}

Whenever an exception is caught corresponding catch block is executed, For example above code catches ArithmeticException only. If some other kind of exception is thrown it will not be caught so it's the programmer work to take care of all exceptions as in our try block we are performing arithmetic so we are capturing only arithmetic exceptions. A simple way to capture any exception is to use an object of Exception class as other classes inherit Exception class, see another example below:

**class** Exceptions {

**public** **static** **void** main(String[] args) {

String languages[] = { "C", "C++", "Java", "Perl", "Python" };

**try** {

**for** (**int** c = 1; c <= 5; c++) {

System.out.println(languages[c]);

}

}

**catch** (Exception e) {

System.out.println(e);

}

}

}

Output of program:

C++

Java

Perl

Python

java.lang.ArrayIndexOutOfBoundsException: 5

**Finally block in Java**

Finally block is always executed whether an exception is thrown or not.

**class** Allocate {

**public** **static** **void** main(String[] args) {

**try** {

**long** data[] = **new** **long**[1000000000];

}

**catch** (Exception e) {

System.out.println(e);

}

**finally** {

System.out.println("finally block will execute always.");

}

}

}

Output of program:

**finally** block will execute always.

Exception in thread "main" java.lang.OutOfMemoryError: Java heap space

at Allocate.main(Allocate.java:5)