Java ArithmeticException Example

```
public class ArithmeticExceptionExample {

 public static void main(String[] args) {
  try {
        int result = 30 / 0;
// Trying to divide by zero
        }
catch (ArithmeticException e) {

System.err.println("ArithmeticException
caught!");
            //e.printStackTrace();
             e.getMessage();
        }
    }
}
```

ArrayIndexOutOfBoundsException

```
public class ArrayIndexOutOfBounds {
    public static void main(String[] args) {
  int[] nums = new int[] {1,2,3};
try {
   int numFromNegativeIndex = nums[-1];
// Trying to access at negative index
   int numFromGreaterIndex = nums[4];
// Trying to access at greater index
  int numFromLengthIndex = nums[3];
} catch (ArrayIndexOutOfBoundsException e) {
System.err.println("ArrayIndexOutOfBoundsExc
eption caught");
            e.printStackTrace();
   }
```

```
 } }
```

Java ClassCastException Examples

Here is a very simple example, an Integer object cannot be cast to a String object:

```
public class ClassCastExceptionExample {
    public static void main(String[] args) {
        Object obj = new Integer(100);
        System.out.println((String) obj);
    }
}
```

Java ClassNotFoundException Example

Below example demonstrates the common causes of **java.lang.ClassNotFoundException** is using `Class.forName` or `ClassLoader.loadClass` to load a class by passing the string name of a class and it's not found on the classpath.

```
public class ClassNotFoundExceptionExample {

    public static void main(String[] args) {

        try {


Class.forName("com.example.corejava.Demo");


ClassLoader.getSystemClassLoader().loadClass
("com.example.corejava.Demo");
```

```
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

Java ClassNotFoundException Example

Below example demonstrates the common causes of **java.lang.ClassNotFoundException** is using `Class.forName` or `ClassLoader.loadClass` to load a class by passing the string name of a class and it's not found on the classpath

```
public class ClassNotFoundExceptionExample {

    public static void main(String[] args) {

        try {


Class.forName("com.example.corejava.Demo");


ClassLoader.getSystemClassLoader().loadClass
("com.example.corejava.Demo");

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

Java IllegalStateException Example

In this example, the Iterator.remove() method throws an IllegalStateException - if the next method has not yet been called, or the remove method has already been called after the last call to the next method.

```
import java.util.ArrayList;

import java.util.Iterator;
import java.util.List;

public class IllegalStateExceptionExample {

    public static void main(String[] args) {

        List < Integer > intList = new
ArrayList < > ();

        for (int i = 0; i < 10; i++) {
            intList.add(i);
        }

        Iterator < Integer > intListIterator
= intList.iterator(); // Initialized with
index at -1

        try {
            intListIterator.remove(); //
IllegalStateException
        } catch (IllegalStateException e) {

System.err.println("IllegalStateException
caught!");
            e.printStackTrace();
        }
    }
}
```

Java InterruptedException Example

In the below example, note that thread interrupted in main() method throws InterruptedException exception that is handled in the run() method.

```
class ChildThread extends Thread {
    public void run() {

   try {
        Thread.sleep(1000);
     }
   catch (InterruptedException e) {
    System.err.println("InterruptedException
          caught!");
     e.printStackTrace();
        }
    }
}
public class InterruptedExceptionExample {
    public static void main(String[] args)
throws InterruptedException {
ChildThread childThread = new ChildThread();
  childThread.start();
  childThread.interrupt();
    }
}
```

```
Output:
```

```
InterruptedException caught!
```

```
java.lang.InterruptedException: sleep
interrupted
 at java.lang.Thread.sleep(Native Method)
 at
com.javaguides.corejava.ChildThread.run(Inte
rruptedExceptionExample.java:7)
```

Java NullPointerException Example

In below example, the person object is null and we are invoking its fields on null object leads to `NullPointerException`.

```
public class NullPointerExceptionExample {

public static void main(String[] args) {

   Person personObj = null;
  try {
    String name = personObj.personName;
// Accessing the field of a null object
personObj.personName = "Ramesh Fadatare";
// Modifying the field of a null object
  } catch (NullPointerException e) {

System.err.println("NullPointerException
caught!");
  e.printStackTrace();
  }
 }
}

class Person {
    public String personName;

    public String getPersonName() {
        return personName;
    }

    public void setPersonName(String 
personName) {
        this.personName = personName;
    }
}
```

```
Output:
NullPointerException caught!
java.lang.NullPointerException
 at 
com.example.corejava.NullPointerExceptionExa
mple.main(NullPointerExceptionExample.java:9
)
```

Java NumberFormatException Example

In the below example, we are trying to parse "`100ABCD`" string into integer leads to `NumberFormatException`:

```
public class NumberFormatExceptionExample {

    public static void main(String[] args) {

    String str1 = "100ABCD";
    try {
    int x = Integer.parseInt(str1);
// Converting string with inappropriate 
format
```

```
    int y = Integer.valueOf(str1);
  } catch (NumberFormatException e) {

System.err.println("NumberFormatException 
caught!");
     e.printStackTrace();
        }
    }
}

Output:
NumberFormatException caught!
java.lang.NumberFormatException: For input 
string: "100ABCD"
 at 
java.lang.NumberFormatException.forInputStri
ng(NumberFormatException.java:65)
 at 
java.lang.Integer.parseInt(Integer.java:580)
 at 
java.lang.Integer.parseInt(Integer.java:615)
 at 
com.example.corejava.NumberFormatExceptionEx
ample.main(NumberFormatExceptionExample.java
:9)
```

Java ParseException Example

In this example, we use `DateFormat.parse(String source)` method which throws `ParseException` object. This `parse()` method throws `ParseException` - if the beginning of the specified string cannot be parsed.
Here is a complete code to throw `ParseException` exception:

```
import java.text.DateFormat;

import java.text.ParseException;
import java.text.SimpleDateFormat;

public class ParseExceptionExample {
 public static void main(String[] args) {
 DateFormat format = new
SimpleDateFormat("MM, dd, yyyy");
  try {
     format.parse("01, , 2010");
    }
catch (ParseException e) {

System.err.println("ParseException
caught!");
     //e.printStackTrace();
    }
 }
}

Output:

ParseException caught!
java.text.ParseException: Unparseable date:
"01, , 2010"
 at
java.text.DateFormat.parse(DateFormat.java:3
66)
 at
com.javaguides.corejava.ParseExceptionExampl
e.main(ParseExceptionExample.java:15)
```

Java StringIndexOutOfBoundsException Example

In this below example, the exception occurred because the referenced index was not present in the String.

```
public class StringIndexOutOfBounds {

 public static void main(String[] args) {

   String str = "Hello World";
  try {
  char charAtNegativeIndex = str.charAt(-1);
// Trying to access at negative index
 char charAtLengthIndex = str.charAt(11);
// Trying to access at index equal to size
of the string
 } catch (StringIndexOutOfBoundsException e)
{

System.err.println("StringIndexOutOfBoundsEx
ception caught");
   e.printStackTrace();
   }
 }
}
```

```
Output:
```

```
StringIndexOutOfBoundsException caught
java.lang.StringIndexOutOfBoundsException:
String index out of range: -1
 at java.lang.String.charAt(String.java:658)
 at
com.example.corejava.StringIndexOutOfBounds.
main(StringIndexOutOfBounds.java:9)
```

```
GCD of Two numbers:
```

```java
public class FindGCDExample1
{
public static void main(String[] args)
{
//x and y are the numbers to find the GCF
int x = 12, y = 8, gcd = 1;
//running loop form 1 to the smallest of both numbers
for(int i = 1; i <= x && i <= y; i++)
{
//returns true if both conditions are satisfied
if(x%i==0 && y%i==0)
//storing the variable i in the variable gcd
gcd = i;
}
//prints the gcd
System.out.printf("GCD of %d and %d is: %d", x, y, gcd);
}
}
```