Integer Overflow

I have used NetBeans to demonstrate a unique and complete demonstration of risky resource management, integer overflow.

**Integer overflow**-

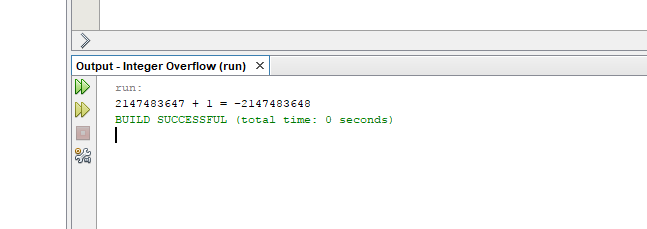
Integer overflow occurs when the result of an arithmetic operation, such as addition or multiplication, exceeds the maximum size of the integer type used to store it.

For example, adding 1 dollar to an account with a balance of 2,147,483,647 could cause an integer underflow and yield a new balance of -2,147,483,647. You can see how this would create a HUGE problem for any application dealing with the smaller primitive data types.

Vulnerability-

For the purpose of this demonstration, we will use two arguments. The arguments will be 1 and Integer.Max\_Value, which gives us the high possible integer value, 2,147,483,647.

The vulnerable application demonstrates a wrap around. Adding the two values will look like this:

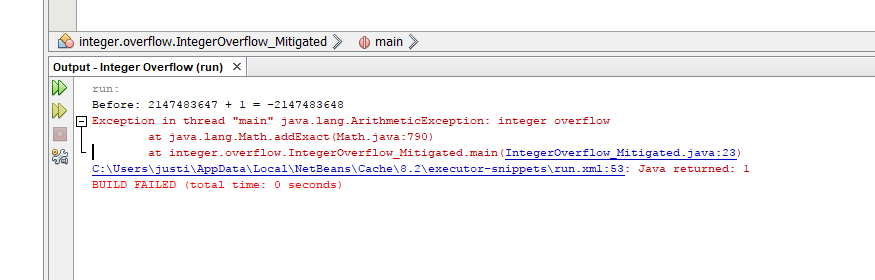


Notice the negative sign. The integer has wrapped back around to its lowest value, much like when a car odometer reaches its highest value and wraps back around to zero.

Mitigation-

There is a method in Java named, “**Math.addExact()**” that we can use to mitigate this issue. The method will throw an *ArithmeticException* if an overflow occurs when it adds its arguments. For this demonstration we will print the result to the screen using the vulnerable method, and using the Math.addExact() method.

This is our result:



The program will not allow this calculation to occur.