

## Re: Sign conventions for remainder

**Source:** <http://sci.tech-archive.net/Archive/sci.math/2004-06/3979.html>

---

**From:** Sean O'Leathlobhair (jwlawler\_at\_yahoo.com)

**Date:** 06/18/04

Date: 18 Jun 2004 04:29:13 -0700

David Eppstein <eppstein@ics.uci.edu> wrote in message  
news:<eppstein-C64F96.13302917062004@news.service.uci.edu>...  
> In article <200406171732.i5HHWRG37280@mickey.empros.com>,  
> mstemper@siemens-emis.com (Michael Stemper) wrote:  
>  
>> *Just for fun, I've been implementing a package to perform arithmetic on  
>> integers of arbitrary size. Addition, subtraction, and multiplication  
>> were all pretty straight-forward. (Subtraction was the easiest!) But,  
>> when I prepared to implement division, I realized that I'm not aware of  
>> the conventions for remainders.*  
>>  
>> *If the divisor and the dividend are both positive, I know that my result  
>> should be  $d = n*q+r$ , with  $0 \leq r < n$ . What if one of them is negative? Should  
>> the remainder still be in that range? Should it be in the negative of that  
>> range? What about if both the divisor and the dividend are negative?*  
>>  
>> *Yes, I am aware that there is no "right" answer to this, there are only  
>> conventions. But, that's exactly what I'm looking for. Conventions are  
>> usually established because they're useful.*  
>  
> *If the divisor is positive, the remainder should still be in the range  
>  $0 \leq r < n$ , and  $d=nq+r$  should still be true. If the divisor is negative,  
> it's less clear to me what the right convention is -- the choice taken  
> by Python is to use the same sign as the divisor, which seems reasonable  
> enough to me.*  
>  
> *The C-C++-Java convention of remainder having the same sign as dividend  
> is wrong and should be avoided.*

Are you talking about the Java primitive numeric types or the classes  
BigInteger and BigDecimal?

The documentation for BigInteger does not seem to be very clear on the  
exact details of the remainder and I have never had the need to check.  
I may knock a simple test program and post the results.

The class BigDecimal approaches the problem a different way. The  
divide method has a plethora of options on how to round divisions. So

sci.math: Re: Sign conventions for remainder

it dumps the problem onto the user. You could copy this idea and have options in your package for the several possible sensible behaviours.

One behaviour that is common but I don't like is round towards zero. Non-mathematicians seem to like it since they seem to expect that  $-5 / 4$  should be similar to  $5 / 4$  except for the sign i.e.  $\pm 1 \text{ rem } 1$ . The reason that I don't like it is that the graph of  $x \text{ rem } n$  has an anomaly at  $x = 0$  which sometimes messes up functions built on top of it.

Opinions vary, give your users the choice.

Here is a link to the Java API docs:

<http://java.sun.com/j2se/1.3/docs/api/>

Seán O'Leathlóbhair