Ruby master - Bug #6120

Float and BigDecimal bug in remainder in corner cases

03/07/2012 02:51 PM - marcandre (Marc-Andre Lafortune)

Status:	Assigned		
Priority:	Normal		
Assignee:	marcandre (Marc-Andre Lafortune)		
Target version:			
ruby -v:	r34927	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

Currently:

```
4.2.remainder(+Float::INFINITY) # => 4.2, ok
4.2.remainder(-Float::INFINITY) # => NaN, should be 4.2
# (same with all signs reversed)
```

Reasons the remainder should be 4.2 and not NaN:
1) foo.remainder(bar) == foo.remainder(-bar)
2) foo.remainder(bar) == foo when bar.abs > foo.abs

Similarly:
require 'bigdecimal'
bd = BigDecimal.new("4.2")
bd.remainder(BigDecimal.new("+Infinity")) # => NaN, should be bd
bd.remainder(BigDecimal.new("-Infinity")) # => NaN, should be bd
(same with all signs reverse)

Reasons: same as float.

Finally:
bd = BigDecimal.new("4.2")
bd.modulo(BigDecimal.new("0")) # => ZeroDivisionError, probably ok?
bd.remainder(BigDecimal.new("0")) # => NaN, should be probably raise a ZeroDivisionError?

Like in #6044, this could be decided either way, as long as there is consistency. Anyone prefer NaN to raising a ZeroDivisionError?

History

#1 - 03/07/2012 10:37 PM - naruse (Yui NARUSE)

Ruby's math should portably follow SuS. [ruby-core:28206]
You can fix it for Float/Math if it is obviously wrong and the right implementation is clear.

BigDecimal is little another world and it is up to mrkn.

#2 - 03/18/2012 06:46 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#3 - 02/18/2013 09:24 PM - mame (Yusuke Endoh)

- Target version changed from 2.0.0 to 2.6

Should it be assigned to mrkn?

--
Yusuke Endoh mame@tsg.ne.jp

#4 - 02/19/2013 01:44 AM - marcandre (Marc-Andre Lafortune)

Didn't get around fixing it for 2.0.0. Will fix and then assign to mrkn for BigDecimal.

#5 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- *Target version deleted (2.6)*