Ruby trunk - Bug #5713

Fixnum#** returns Infinity for 0 ** negative Bignum

12/06/2011 12:36 PM - john_firebaugh (John Firebaugh)

Status:	Closed		
Priority:	Normal		
Assignee:	marcandre (Marc-Andre Lafortune)		
Target version:	2.0.0		
ruby -v:	ruby 1.9.3p0 (2011-10-30 revision 33570) [x86_64-darwin10.8.0]	**Backport:**	

Description

=begin
Instead it should raise ZeroDivisionError, the same as negative Fixnums.

wordsize = 8 * 1.size
fixnum_min = -2 ** (wordsize - 2)

def zero_power(exp)
0 ** exp
rescue ZeroDivisionError
"ZeroDivisionError"
end

[-1, fixnum_min, (fixnum_min-1)].each {|i| puts zero_power(i)}
=end

Related issues:

Related to Ruby trunk - Bug #5715: +/-1 ** Bignum returns different results t...	**Closed**	**12/06/2011**

Associated revisions

Revision 8797ebd6 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@39063 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision 39063 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]

Revision d22ce4a5 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@39064 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

Revision 39064 - 02/05/2013 05:39 AM - marcandre (Marc-Andre Lafortune)

- numeric.c (fix_pow): Handle special cases when base is 0, -1 or +1 [Bug #5713] [Bug #5715]

History

#1 - 12/09/2011 02:29 AM - marcandre (Marc-Andre Lafortune)

- Assignee set to marcandre (Marc-Andre Lafortune)

#2 - 12/10/2011 01:36 AM - john_firebaugh (John Firebaugh)

The execution path of 0 ** -Bignum goes to Rational(0) ** -Bignum, so I think the issue is there. I.e. Rational(0) ** -Bignum should raise ZeroDivisionError, the same as Rational(0) ** -Fixnum.

#3 - 12/10/2011 03:10 AM - marcandre (Marc-Andre Lafortune)

- Category set to core

- Target version set to 2.0.0

John Firebaugh wrote:

> The execution path of 0 ** -Bignum goes to Rational(0) ** -Bignum, so I think the issue is there. I.e. Rational(0) ** -Bignum should raise ZeroDivisionError, the same as Rational(0) ** -Fixnum.

Yes, unless there is objection, Rational#** should treat 0 and 1 as special cases before resorting to conversion to float, i.e.

```
Rational(0) ** (-2**100) # => Infinity, should raise an Error
Rational(0) ** (2**100) # => 0.0, should be Rational(0)
Rational(1) ** (2**100) # => 1.0, should be Rational(1)
```

#4 - 12/10/2011 08:42 AM - john_firebaugh (John Firebaugh)

FYI, I've been doing RubySpec work on this in Rubinius: https://github.com/rubinius/rubinius/commits/master/spec/ruby/shared/rational/exponent.rb

#5 - 03/18/2012 06:46 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#6 - 02/05/2013 02:39 PM - marcandre (Marc-Andre Lafortune)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r39063.
John, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- rational.c (nurat_expt): Deal with special cases for rationals 0, ±1 [bug #5713] [bug #5715]