

CommonRuby - Feature #12508

Integer#mod_pow

06/20/2016 01:48 PM - metanest (Makoto Kishimoto)

Status:	Closed	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		
Description		
A new method Integer#mod_pow, power with modulo. a.mod_pow(b, m) #=> (a**b) % m Sometimes a**b becomes very large number, then naive implementation may be unefficient. Fast implementation is useful. (with USE_GMP symbol, this implement uses mpz_powm()) (see https://github.com/ruby/ruby/pull/1320)		
Related issues:		
Has duplicate Ruby trunk - Feature #11003: Fast modular exponentiation		Closed

History

#1 - 08/09/2016 08:04 AM - matz (Yukihiro Matsumoto)

- Status changed from Open to Feedback

Instead, I propose pow(a) and pow(a,b) where the latter works as mod_pow() here.

Matz.

#2 - 12/30/2016 09:14 AM - metanest (Makoto Kishimoto)

Updated as Integer#pow, with such API.

#3 - 01/22/2017 03:20 AM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

- Status changed from Feedback to Assigned

#4 - 02/22/2017 07:40 AM - matz (Yukihiro Matsumoto)

Go ahead and add pow(a,b).

Matz.

#5 - 12/01/2017 08:33 AM - mrkn (Kenta Murata)

- Has duplicate Feature #11003: Fast modular exponentiation added

#6 - 12/04/2017 02:35 AM - mrkn (Kenta Murata)

- Status changed from Assigned to Closed

Applied in changeset [ruby-trunk:trunk|r61003](#).

bignum.c, numeric.c: add Integer#pow(b, m)

This commit is based on the pull-request [#1320](#) created by Makoto Kishimoto.
[Feature [#12508](#)] [Feature [#11003](#)] [close GH-1320]

- bignum.c (rb_int_powm): Added for Integer#pow(b, m).
- internal.h (rb_int_powm): Declared to refer in numeric.c.

- `bignum.c` (`bary_powm_gmp`): Added for `Integer#pow(b, m)` using GMP.
- `bignum.c` (`int_pow_tmp1`): Added for implementing `Integer#pow(b, m)`.
- `bignum.c` (`int_pow_tmp2`, `int_pow_tmp3`): ditto.
- `internal.h` (`rb_num_positive_int_p`): Moved from `numeric.c` for sharing the definition with `bignum.c`.
- `internal.h` (`rb_num_negative_int_p`, `rb_num_compare_with_zero`): ditto.
- `numeric.c` (`negative_int_p`): Moved to `internal.h` for sharing the definition with `bignum.c`.
- `numeric.c` (`positive_int_p`, `compare_with_zero`): ditto.
- `numeric.c` (`rb_int_odd_p`): Exported (renamed from `int_odd_p`).
- `internal.h` (`rb_int_odd_p`): ditto.
- `internal.h` (`HALF_LONG_MSB`): Added.
- `numeric.c` (`SQRT_LONG_MAX`): Redefined by using `HALF_LONG_MSB`.
- `test/ruby/test_numeric.rb` (`test_pow`): Added for `Integer#pow(b, m)`.