

Ruby master - Feature #12244

Add a way to `integer - integer % num`

04/02/2016 01:45 PM - naruse (Yui NARUSE)

Status:	Open
Priority:	Normal
Assignee:	matz (Yukihiro Matsumoto)
Target version:	
Description	
We sometimes calculates integer - integer % num.	
For example time series events into time partitions, we write code like	
<pre>event # {time: 1459580435, name: "hoge", text: "Rawr!"} partition = event[:time] - event[:time] % num chunk = get_chunk(partition) chunk.write event</pre>	
https://twitter.com/tagomoris/status/715814050534461440 https://twitter.com/tagomoris/status/715814961260457985	
The name is always issue. There are some suggestions likes Integer#adjust](https://twitter.com/cocoatomo/status/716088708655489024).	
kosaki says Excel's FLOOR() function is FLOOR(number, significance) . Therefore Ruby should be Integer#floor(digits=1, significance: nil) . kosaki agrees this .	
I checked the speed of a half baked implementation..., but it's 10x slow...	
<pre>% time ./miniruby -e'i=10000000;while i>0;i-=1;1459497599.floor(significance: 3600);end' ./miniruby 6.58s user 0.02s system 99% cpu 6.596 total #3 1459604131 22:35:31 naruse@windy:~/obj/ruby % time ./miniruby -e'i=10000000;while i>0;i-=1;t=1459497599;t-t%3600;end' ./miniruby -e'i=10000000;while i>0;i-=1;t=1459497599;t-t%3600;end' 0.52s user 0.00s system 99% cp u 0.520 total</pre>	
<pre>diff --git a/numeric.c b/numeric.c index 37217a1..f6acfe3 100644 --- a/numeric.c +++ b/numeric.c @@ -4123,6 +4123,42 @@ int_dotimes(VALUE num) /* * call-seq: + * int.floor([ndigits]) -> integer or float + * + * Rounds +int+ to a given precision in decimal digits (default 0 digits). + * + * Precision may be negative. Returns a floating point number when +ndigits+ + * is positive, +self+ for zero, and round down for negative. + * + * 1.round #=> 1 + * 1.round(2) #=> 1.0 + * 15.round(-1) #=> 20 + */ + +static VALUE +int_floor(int argc, VALUE* argv, VALUE num) +{ + static ID keyword_ids[1]; + VALUE kwargs[1], ndigits, opt; + if (!keyword_ids[0]) {</pre>	

```

+     CONST_ID(keyword_ids[0], "significance");
+ }
+
+ rb_scan_args(argc, argv, "01:", &ndigits, &opt);
+ if (!NIL_P(opt)) {
+     VALUE factor;
+     long a, b;
+     rb_get_kwargs(opt, keyword_ids, 0, 1, kwargs);
+     factor = kwargs[0];
+     a = FIX2LONG(num);
+     b = FIX2LONG(factor);
+     return LONG2FIX(a - a % b);
+ }
+ return Qnil;
+}
+
+/*
+ * call-seq:
+ *   int.round([ndigits]) -> integer or float
+ *
+ * Rounds +int+ to a given precision in decimal digits (default 0 digits).
@@ -4321,7 +4357,7 @@ Init_Numeric(void)
+     rb_define_method(rb_cInteger, "to_i", int_to_i, 0);
+     rb_define_method(rb_cInteger, "to_int", int_to_i, 0);
+     rb_define_method(rb_cInteger, "to_f", int_to_f, 0);
-     rb_define_method(rb_cInteger, "floor", int_to_i, 0);
+     rb_define_method(rb_cInteger, "floor", int_floor, -1);
+     rb_define_method(rb_cInteger, "ceil", int_to_i, 0);
+     rb_define_method(rb_cInteger, "truncate", int_to_i, 0);
+     rb_define_method(rb_cInteger, "round", int_round, -1);

```

History

#1 - 04/02/2016 06:37 PM - naruse (Yui NARUSE)

- Description updated

#2 - 04/02/2016 06:41 PM - naruse (Yui NARUSE)

Why this new method is too slow seems because of keyword argument.

#3 - 05/17/2016 06:05 AM - matz (Yukihiko Matsumoto)

The proposed behavior seems reasonable for me.

The only concern is performance. We need measurement.

Matz.