Ruby master - Bug #1317

Creating a range with strings

03/27/2009 06:56 AM - ian (Ian Bailey)

Status:	Rejected		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 1.8.6 (2007-09-24 patchlevel 111) [i486-linux]	**Backport:**	

Description

=begin
irb(main):027:0> Range.new("1", "10").to_a
=> ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
irb(main):028:0> Range.new("2", "10").to_a
=> []

The second result happens when the end of the range is greater numerically but the first character of the start is higher than the first character of the end.

It seems like these two return values are inconsistent. If the range is able to work a list of entries between two integers as strings, it should do it regardless of their first character.
=end

History

#1 - 03/27/2009 07:36 AM - mike (Michael Selig)

=begin
This is a (possibly confusing) consequence of the behaviour of String#next.
For most sortable objects, the following is true:

obj.next > obj

but not for strings. For example:

"9".next => "10"
but
"10" > "9" => false

So if you think of a range as a potential "for" loop in a C-style syntax it would be:

for (obj = start; obj < end; obj = next(obj)) ....

then you will see that the loop will stop immediately when start is "2" and end is "10".

Obviously the behaviour of string comparison can't be changed.

Cheers
Mike.
=end

#2 - 03/27/2009 07:59 AM - jredville (Jim Deville)

=begin

> -----Original Message-----
> From: Michael Selig [mailto:redmine@ruby-lang.org]
> Sent: Thursday, March 26, 2009 3:34 PM
> To: ruby-core@ruby-lang.org
> Subject: [ruby-core:23027] [Bug #1317] Creating a range with strings
>
> Issue #1317 has been updated by Michael Selig.
>
> This is a (possibly confusing) consequence of the behaviour of
> String#next.
> For most sortable objects, the following is true:

> obj.next > obj
>
> but not for strings. For example:
>
> "9".next => "10"
> but
> "10" > "9" => false

Why doesn't Ruby have a collation concept to allow this to be changed?

JD

=end

#3 - 03/27/2009 10:18 AM - phasis68 (Heesob Park)

=begin
2009/3/27 Michael Selig redmine@ruby-lang.org:

> Issue #1317 has been updated by Michael Selig.
>
> This is a (possibly confusing) consequence of the behaviour of String#next.
> For most sortable objects, the following is true:
>
> obj.next > obj
>
> but not for strings. For example:
>
> "9".next => "10"
> but
> "10" > "9" => false
>
> So if you think of a range as a potential "for" loop in a C-style syntax it would be:
>
> for (obj = start; obj < end; obj = next(obj)) ....
>
> then you will see that the loop will stop immediately when start is "2" and end is "10".
>
> Obviously the behaviour of string comparison can't be changed.
>
> I'm not sure the range use String#next.

On ruby 1.9.1 and 1.8.6, "Z".next is "AA".

But the result is different from each other.

[sidns@ns httpd]$ ruby -v -e "p(('A'..'z').to_a)"
ruby 1.8.6 (2007-09-24 patchlevel 111) [i686-linux]
["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N",
"O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"]

[siweb@localhost ~]$ ruby -v -e "p(('A'..'z').to_a)"
ruby 1.9.1p0 (2009-01-30 revision 21907) [i686-linux]
["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N",
"O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z", "[", "\",
"]", "^", "_", "`", "a", "b", "c", "d", "e", "f", "g", "h", "i", "j",
"k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x",
"y", "z"]

Regards,

Park Heesob

=end

#4 - 07/16/2009 06:10 PM - yugui (Yuki Sonoda)

- Status changed from Open to Rejected

=begin

=end