Ruby - Bug #19583

Unary minus inconsistency when used with variables and literals

04/06/2023 06:43 AM - hurricup (Alexandr Evstigneev)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN, 3.2: UNKNOWN

Description

This feels a bit inconsistent and I could not find an explanation.

This is fine and - has higher precedence than .

```
-2.upto 0 do |arg|
puts arg
end
```

But this is not working, won't even compile (requires parens):

```
var = 2
-var.upto 0 do |arg|
puts arg
end
```

I presume that in the first example there is no unary minus operation, just negative literal.

Ok, according to this - my assumption is correct. The question is - why?

```
def some
  puts -2
  a = 42
  puts -a
end

puts RubyVM::InstructionSequence.of(method :some).disasm
```

History

#1 - 04/06/2023 06:47 AM - hurricup (Alexandr Evstigneev)

- Description updated

#2 - 04/06/2023 07:23 AM - Hanmac (Hans Mackowiak)

-2 is a literal
- 2 is a function call of -@

and -@ doesn't have preference over function call, see this:

```
a = 2
v = -a.to_s
p v #=> 2
```

#3 - 04/06/2023 08:25 AM - byroot (Jean Boussier)

- Status changed from Open to Closed

Yeah, I don't think this is a bug, and even if we decided another precedence would have been better, changing it would break way too much code.