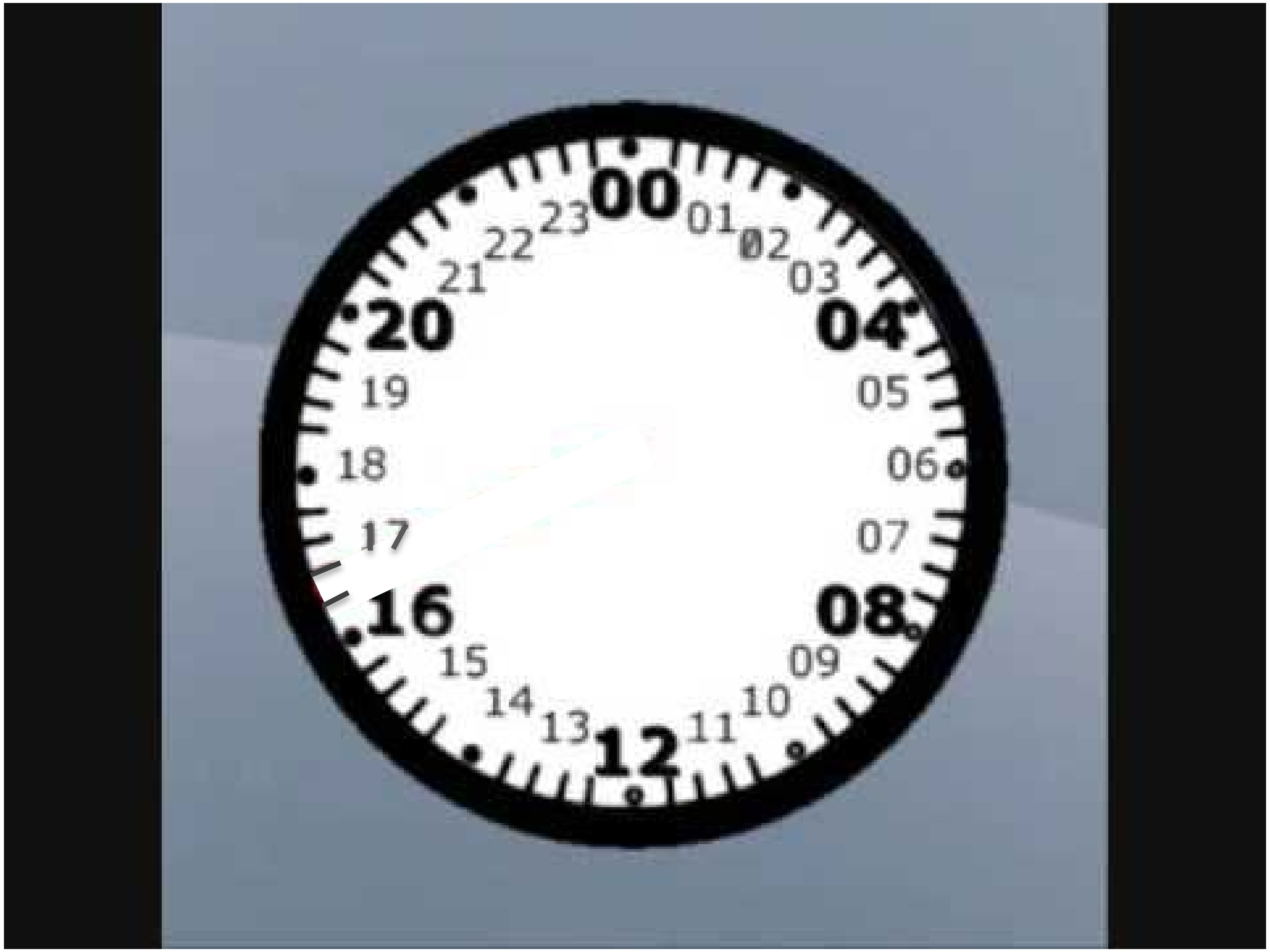
Clock Arithmetic

Mathematical Modulo (“mod”)

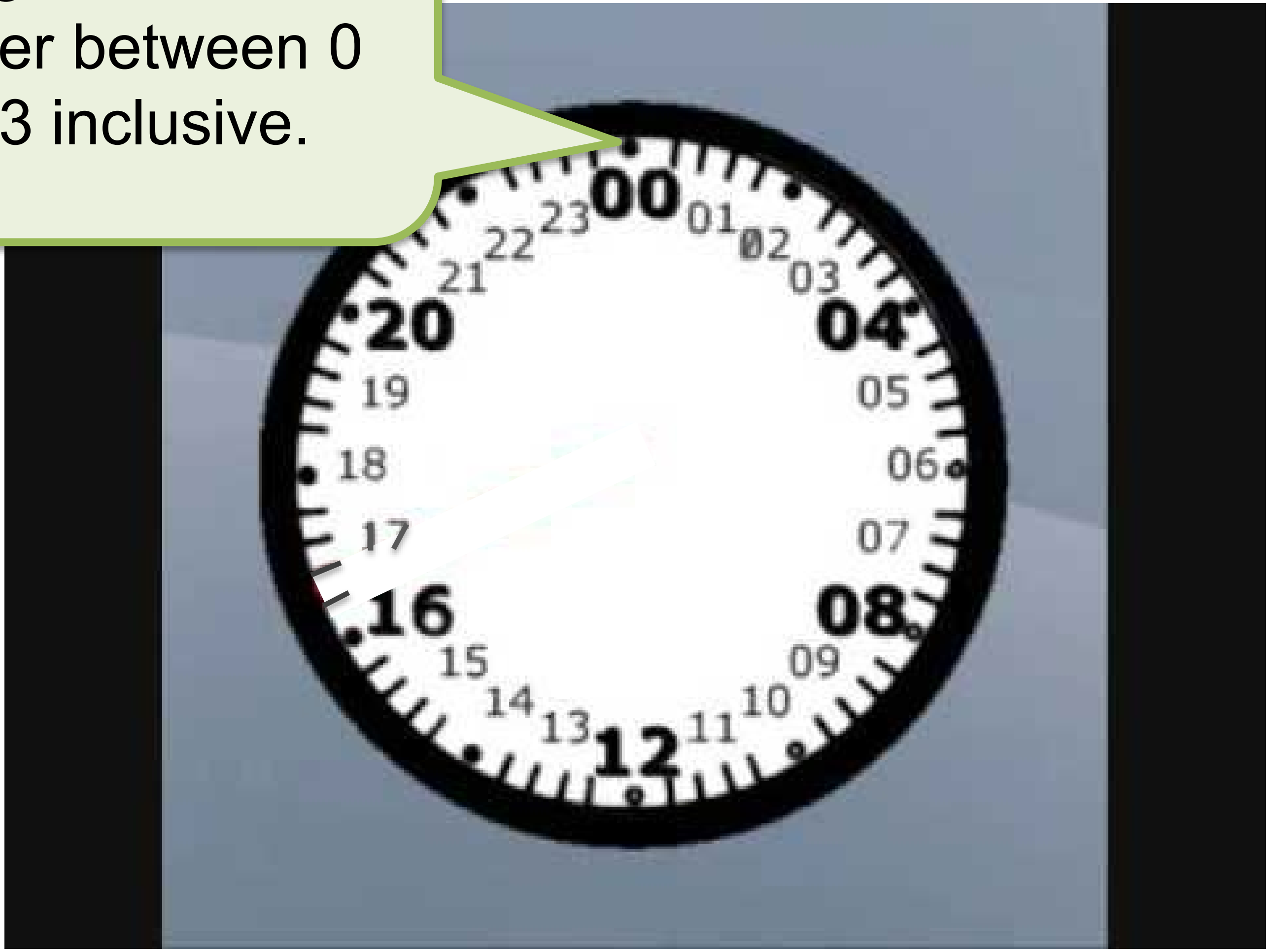
- The value of `a` **`mod`** `b`, or `a` ***modulo*** `b`, where `a` and `b` are mathematical **`integer`**s and `b > 0`, is computed by doing ***clock arithmetic*** on a clock face with `b` positions
  - If `a > 0`, the “hand” on the clock starts at `0` and moves `|a|` positions clockwise
  - If `a < 0`, it moves `|a|` counter-clockwise
  - Where it ends up is the value of `a` **`mod`** `b`

Example: 24-hr Clock



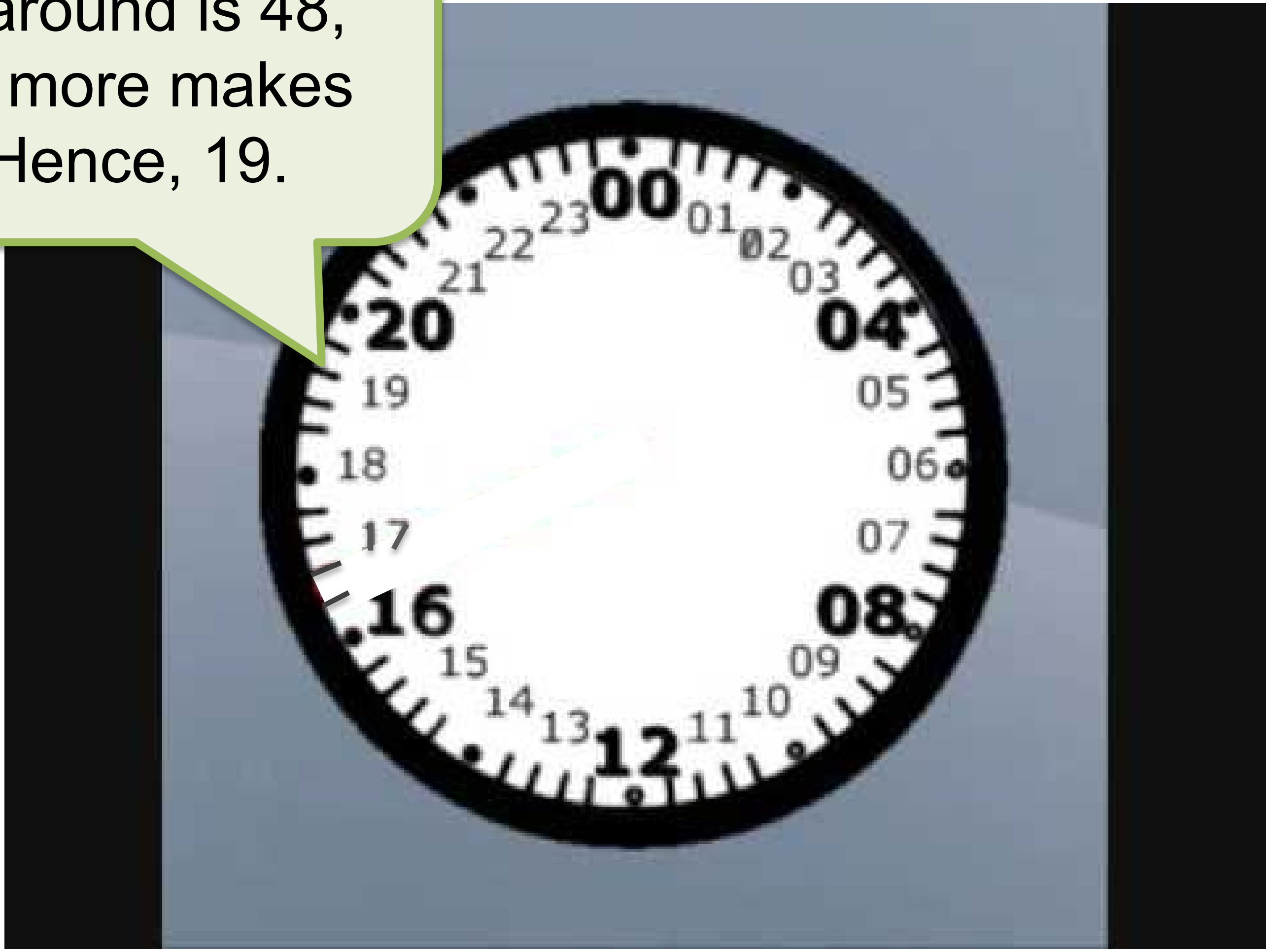
Example: 24-hr Clock

Any integer ***mod*** *24* is a number between 0 and 23 inclusive.



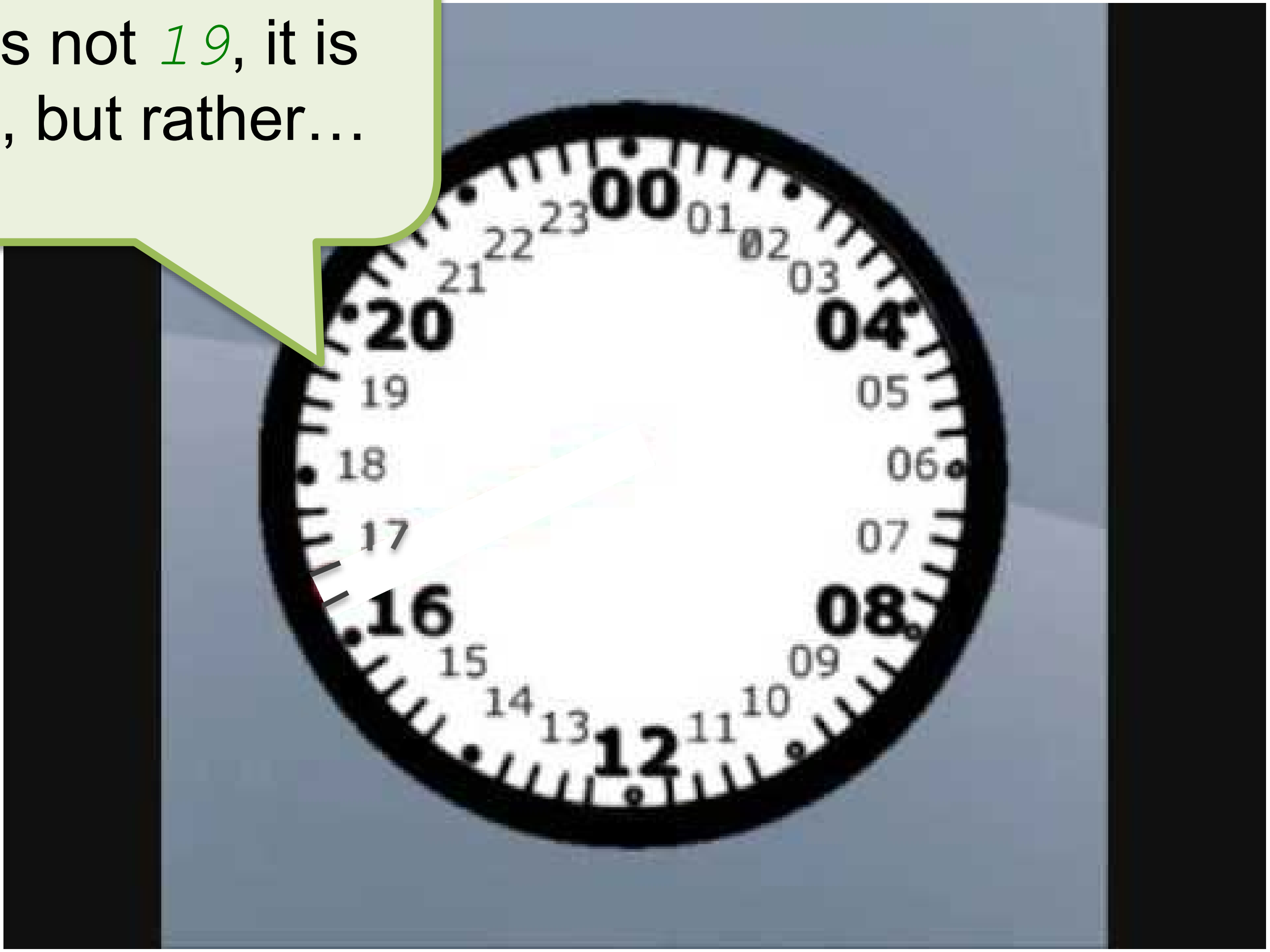
Example: 24-hr Clock

What is `67 mod 24`?
Twice around is 48,
and 19 more makes
67. Hence, 19.



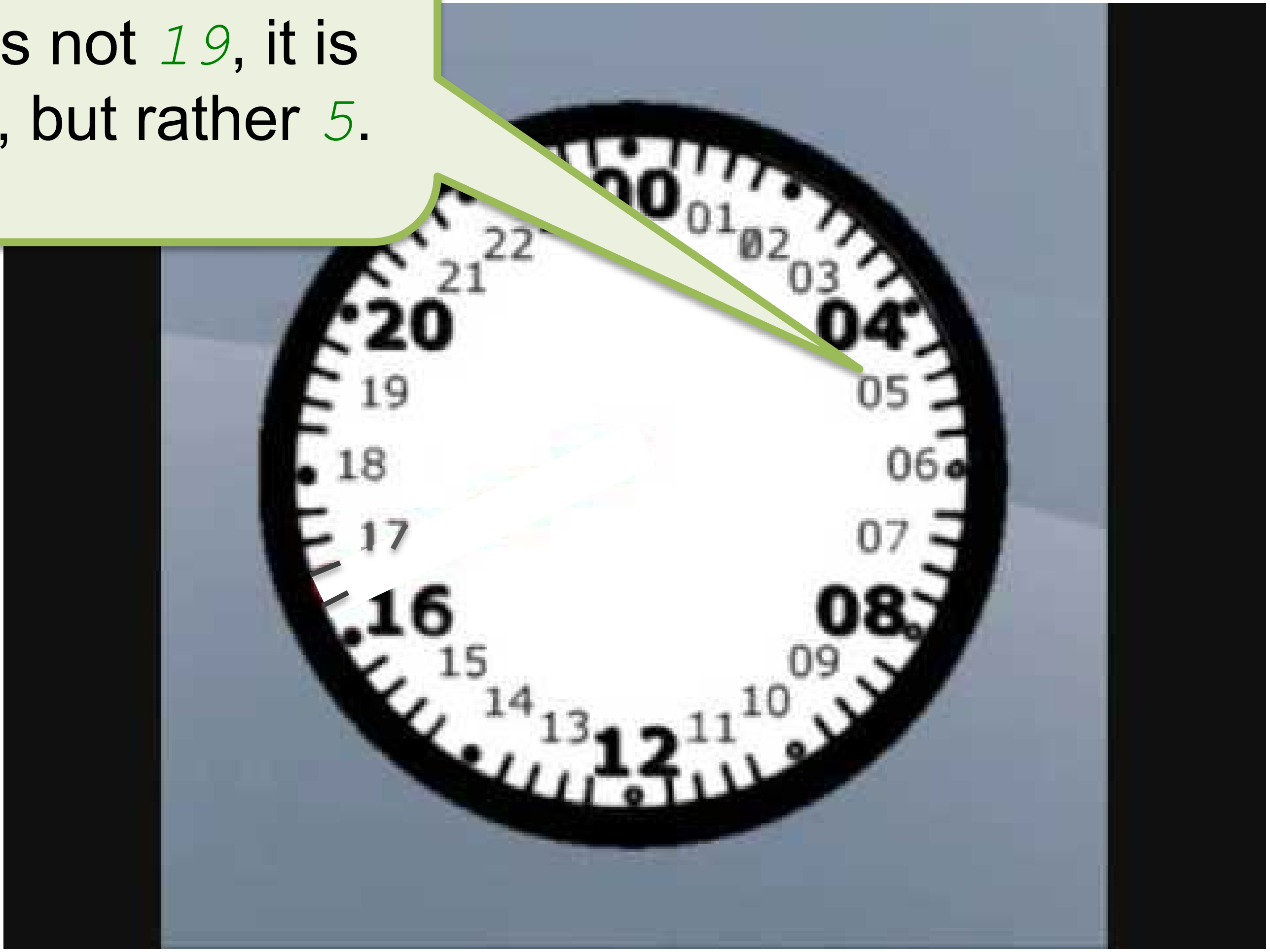
Example: 24-hr Clock

And *(−67)* ***mod*** *24*?
Hint: it is not *19*, it is
not *−19*, but rather…



Example: 24-hr Clock

And *(−67)* ***mod*** *24*?
Hint: it is not *19*, it is
not *−19*, but rather *5*.



Modulo ≠ Remainder (`%`)

- What is the ***remainder*** upon dividing 67 by 24? It is 19.
- What is the ***remainder*** upon dividing –67 by 24? It is –19.
  - At least most people would say it is, and indeed this is how Java evaluates the expression:

```
(-67) % 24
```