FLOATS

Coercing to Integers

Float → Integer data loss

different ways to configure this data loss

10.4

10.5

10.6 10? 11?

10.0001

10.9999

truncation

floor

ceiling

rounding

data loss in all cases

pick your poison!

Truncation

truncating a float simply returns the integer portion of the number

i.e. ignores everything after the decimal point

The math module provides us the trunc( ) function:

```
import math

math.trunc(10.4) → 10
math.trunc(10.5) → 10
math.trunc(10.6) → 10

math.trunc(-10.4) → -10
math.trunc(-10.5) → -10
math.trunc(-10.6) → -10
```

The `int` Constructor

The Python `int` constructor will accept a `float`

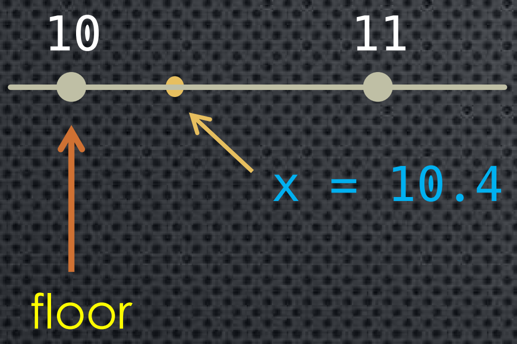
uses truncation when casting the `float` to an `int`

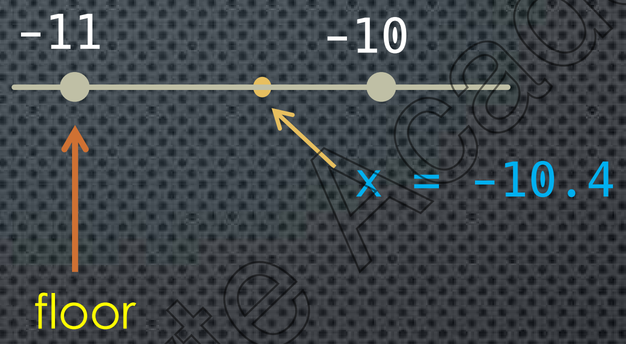
```
int(10.4) → 10
int(10.5) → 10
int(10.6) → 10

int(-10.4) → -10
int(-10.5) → -10
int(-10.6) → -10
```

Floor

Definition: the floor of a number is the largest integer less than (or equal to) the number

`( )` `{` `}`





For positive numbers, floor and truncation are equivalent but not for negative numbers!

Recall also our discussion on integer division – aka floor division: `//`

We defined floor division in combination with the mod operation `n = d * (n // d) + (n % d)`

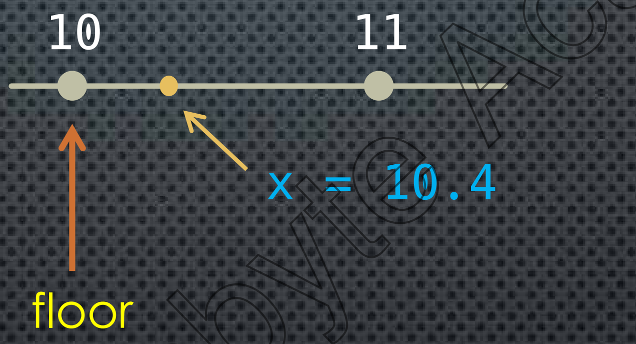
But in fact, floor division defined that way yields the same result
as taking the floor of the floating point division

```
a // b == floor (a / b)
```

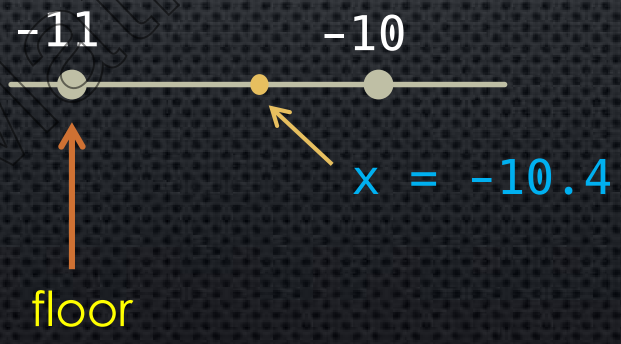
Floor

The math module provides us the floor( ) function:

```
import math

math.floor(10.4) → 10
math.floor(10.5) → 10
math.floor(10.6) → 10
```



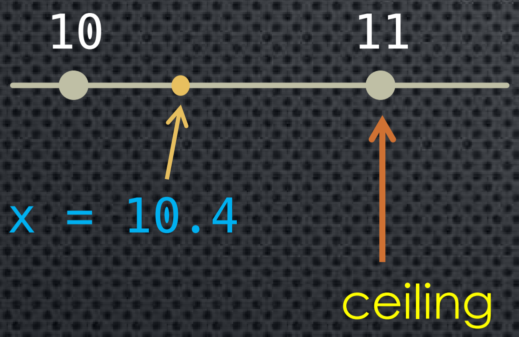
```
math.floor(-10.4) → -11
math.floor(-10.5) → -11
math.floor(-10.6) → -11
```



Ceiling

Definition: the ceiling of a number is the smallest integer greater than (or equal to) the number

( ) { }

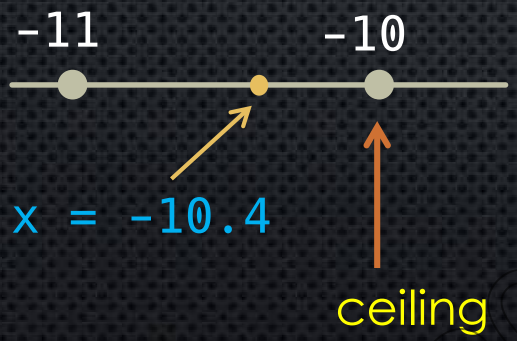


10 11

x = 10.4

ceiling

```
math.ceil(10.4) → 11
math.ceil(10.5) → 11
math.ceil(10.6) → 11
```



-11 -10

x = -10.4

ceiling

```
math.ceil(-10.4) → -10
math.ceil(-10.5) → -10
math.ceil(-10.6) → -10
```

Code