div 1 2 => 0, denn 1/2 = 0.5, aber div wird für ganzzahlige Teilungen verwendet und immer nach unten

gerundet

8^0 => 1, denn x^0 immer 1 ist

rem 5 2 => 1 denn rem das Rest nach einer Teilung kalkuliert

sqrt 3 => 1.7320508075688772 (Würzel aus drei ist 1.732)

"ba"<"aba" => False, weil ‘b’ nach ‘a’ alphabetisch steht

8^200 => 41495155688809929585124639970704894710379428181803401509523685376 (8 hoch 200)

4 /= 5 => True, weil 4 nicht gleich 5 ist

mod 5 2 => 1, weil das Rest von 5/2 1 ist

sqrt (-1) => NaN (Not a Number), weil kein Würzel von eine negative Zahl gibt

True < False, weil (ich vermute) True als 1 gelten kann und False als 0. Man kann aber nicht Bool mit Zahl verglichn, deswegen... o.O

* 1. == 0.3/3 => False, weil 0.1 im Speicher nicht genau vorstellbar ist

rem 5 (-2) => 1, weil das positive Rest von 5/(-2) gleich 1 ist

exp 1 => e^1 = 2.718281828459045

True || False => True, weil bei “ODER” muss nur eine Aussage wahr sein

2\*\*1024 => Infinity – eigentlich ist das 2^1024, aber \*\* ist ^ für nicht ganze Zahle, wobei nicht genug Speicher gibt (?)

abs -7 => Fehler, weil es abs (-7) sein muss. So wäre es 7 sein, weil abs immer eine positive Zahl gibt

mod 5 (-2) => -1

'z'<'A' => False, weil ASCII Code von ‘z’ 122 ist und von ‘A‘ 65

True && False => False, weil bei „AND“ müssen beide Aussagn wahr sein