1. Seeing NaN when the input was (five \* 2);
2. What caused the line to break was the \n
3. In this step the output computes 100 / 2 and that is 50 and so the output

Will print 50 instead of the 100 / 2

1. 5.2: console.log (half of 100 is ${100 / 2});

Output: half of 100 is 50

Explanation: The expression ${100 / 2} evaluates to 50, and the backticks allow for template literals. This means the value is embedded directly in the string.

5.3: console.log ("con" + "cat" + "e" + "nate");

Output: concatenate

Explanation: The + operator concatenates the strings together. So, it combines "con", "cat", "e", and "nate" into one continuous string.

5.4: console.log (8 \* null);

Output: 0

Explanation: In JavaScript, null is treated as 0 when used in a numeric context. Therefore, 8 \* null equals 8 \* 0, which results in 0.

5.5: console.log ("5" - 1);

Output: 4

Explanation: The - operator causes JavaScript to convert the string "5" to the number 5 for the operation. Thus, the calculation becomes 5 - 1, resulting in 4.

5.6: console.log ("5" + 1);

Output: 51

Explanation: The + operator, when used with a string, triggers concatenation. Therefore, the number 1 is converted to a string, and the result is "5" + "1" which equals "51".

5.7: console.log ("five" \* 2);

Output: NaN

Explanation: "five" cannot be converted into a number, and the \* operator expects numeric operands. Since it can't perform the multiplication, it returns NaN (Not a Number).

5.8: console.log (false == true);

Output: false

Explanation: The comparison checks if the two Boolean values are equal. Since false is not equal to true, the result is false.