Answers with Explanations

1. a. `System.out.println(2 + "bc");`
    - Output: `2bc`
    - Explanation: `2` is treated as part of a string concatenation with `"bc"`, resulting in `"2bc"`.
2. b. `System.out.println(2 + 3 + "bc");`
    - Output: `5bc`
    - Explanation: `2 + 3` is evaluated first, resulting in `5`, which is then concatenated with `"bc"`.
3. c. `System.out.println((2 + 3) + "bc");`
    - Output: `5bc`
    - Explanation: The parentheses force `2 + 3` to be evaluated first, resulting in `5`, which is then concatenated with `"bc"`.
4. d. `System.out.println("bc" + (2 + 3));`
    - Output: `bc5`
    - Explanation: The expression inside the parentheses `(2 + 3)` is evaluated first, resulting in `5`, and then `"bc"` is concatenated with `5`.
5. e. `System.out.println("bc" + 2 + 3);`
    - Output: `bc23`
    - Explanation: `"bc"` is concatenated with `2`, resulting in `"bc2"`, and then `3` is concatenated, producing `"bc23"`.
6. f. `System.out.println("bc" + (2 + 3) + "a" + 5);`
    - Output: `bc5a5`
    - Explanation: `(2 + 3)` is evaluated first, resulting in `5`, which is then concatenated with `"bc"`, giving `"bc5"`. The `"a"` is then concatenated, resulting in `"bc5a"`, and finally `5` is concatenated, giving `"bc5a5"`.
7. g. `System.out.println(5 + "bc" + (2 * 3));`
    - Output: `5bc6`
    - Explanation: `5` is concatenated with `"bc"`, resulting in `"5bc"`. Then, `(2 * 3)` is evaluated, resulting in `6`, which is concatenated to give `"5bc6"`.

These explanations should clarify the behavior of operator precedence and associativity in

Java, especially with the `+` operator for string concatenation and arithmetic operations.