**Description of the code changing to tokens:**

This Python script is a simple lexical analyzer, or tokenizer, that processes a given source code and converts it into a sequence of tokens. Here’s a breakdown of what each part of the code does:

1. \*\*Importing `re` Module\*\*: The script starts by importing the regular expression module (`re`), which is used to match patterns in the source code.

2. \*\*Defining Token Types\*\*: The `token\_specification` list defines different types of tokens the lexer should recognize. Each tuple in the list contains:

- A token name (like `NUMBER`, `ID`, `ASSIGN`, etc.).

- A regular expression pattern to match the token.

3. \*\*Compiling Token Specification\*\*: The `tok\_regex` variable compiles these token specifications into a single regular expression pattern. The `get\_token` function uses this pattern to find all matches in the source code.

4. \*\*Source Code Input\*\*: The `source\_code` variable contains a simple example of source code to be tokenized.

5. \*\*Tokenizer Function\*\*:

- The `tokenize` function takes the source code as input.

- It uses the `get\_token` function to find all matches in the code.

- For each match (`mo`), it determines the type of token (`kind`), its value (`value`), and its position in the code (`column`).

- It converts the `NUMBER` tokens to their appropriate types (integer or float).

- If a token is a space or tab (`SKIP`), it continues without yielding it.

- If an unmatched character is found (`MISMATCH`), it raises an error.

- The function then yields each token as a tuple containing the token type, its value, the line number, and the column number.

6. \*\*Tokenize the Source Code\*\*: The `tokens` variable stores the list of tokens produced by the `tokenize` function when run on the provided source code.

7. \*\*Printing Tokens\*\*: Finally, it prints each token in the list, which shows the token type, value, line number, and column number.

By running this script, you will get the output of the tokens identified in the `source\_code`, providing a structured way of analyzing and understanding the components of the code.