Code with Explanation

Original Code

```
def greet(name):
    return f"Hello, {name}!"

print(greet("Broski11"))
```

Explanation

Sure! Let's break down this Python code piece by piece.

1. The `def` Statement

```
def greet(name):
```

- **`def`**: This keyword is used to define a function in Python.
- **`greet`**: This is the name of the function. You can choose any name that makes sense for what the function does.
- **`(name)`**: This is the parameter of the function. A parameter is a variable that the function expects to receive when it is called. In this case, the function expects a single value (which will be a name).

2. The `return` Statement

```
    return f"Hello, {name}!"
```

- **`return`**: This keyword is used to send a result back to the caller of the function. Whatever follows `return` will be the output of the function.
- **`f"Hello, {name}!"`**: This is a formatted string (often called an f-string). The `f` before the quotes tells Python that the string should be formatted. Inside the string, `{name}` is a placeholder that will be replaced by the value of the `name` variable. For example, if `name` is "Broski11", the string will become "Hello, Broski11!".

3. The `print` Statement

```
print(greet("Broski11"))
```

- **`greet("Broski11")`**: This is a function call. It calls the `greet` function and passes "Broski11" as the argument for the `name` parameter.
- **`print`**: This function is used to display output to the console. It prints the result of the `greet` function.

Putting It All Together

- The `greet` function takes a name as input, formats a greeting string, and returns it.
- The `print` function then takes the result of `greet("Broski11")`, which is "Hello, Broski11!", and prints it to the console.

Final Output

When you run this code, the output will be:

```
Hello, Broski11!
```

This is a simple example of how functions work in Python, and it shows how to use formatted strings to create dynamic outputs.