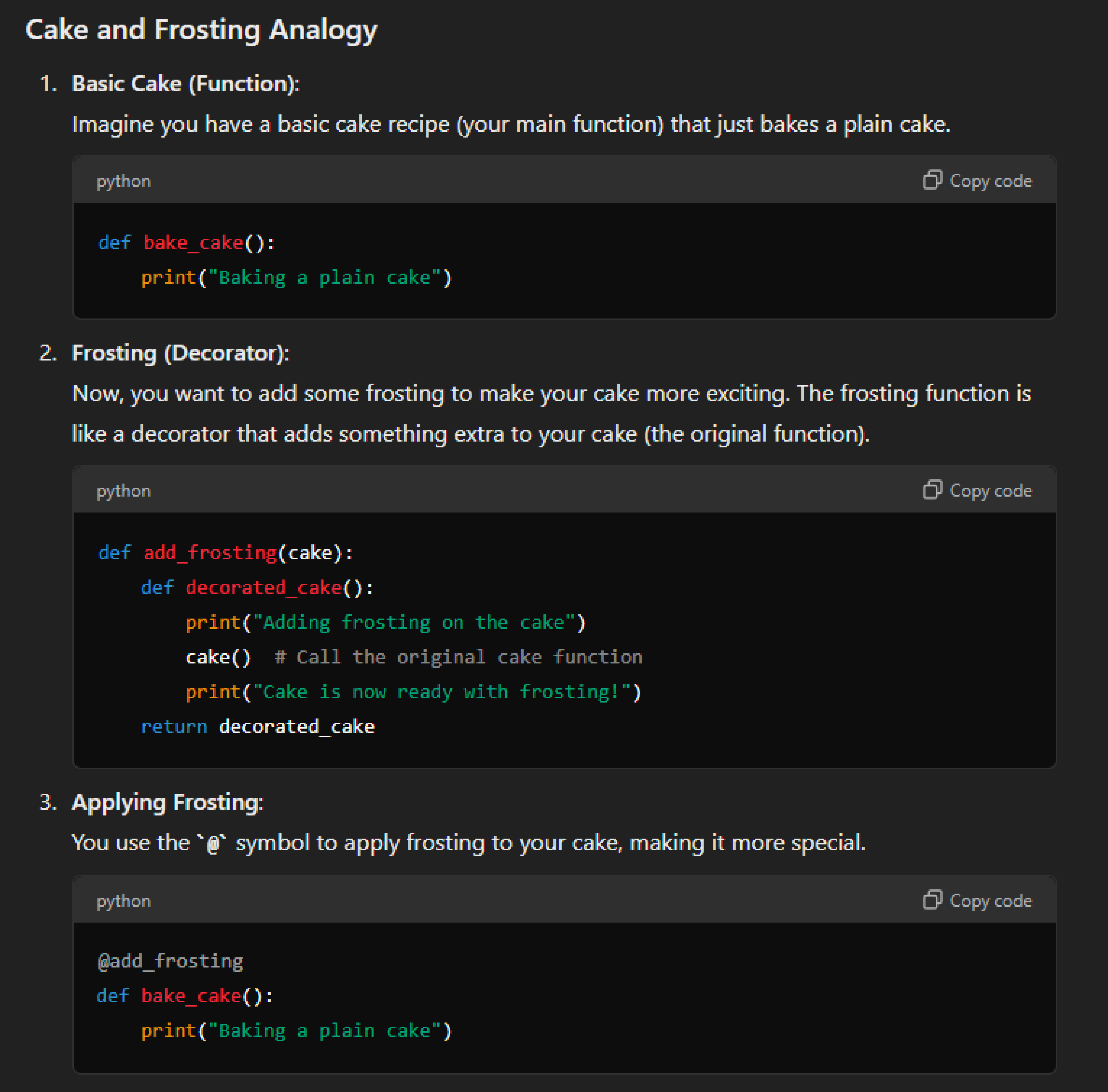


## Cake and Frosting Analogy

1. **Basic Cake (Function):**

   Imagine you have a basic cake recipe (your main function) that just bakes a plain cake.

```python
def bake_cake():
    print("Baking a plain cake")
```

2. **Frosting (Decorator):**

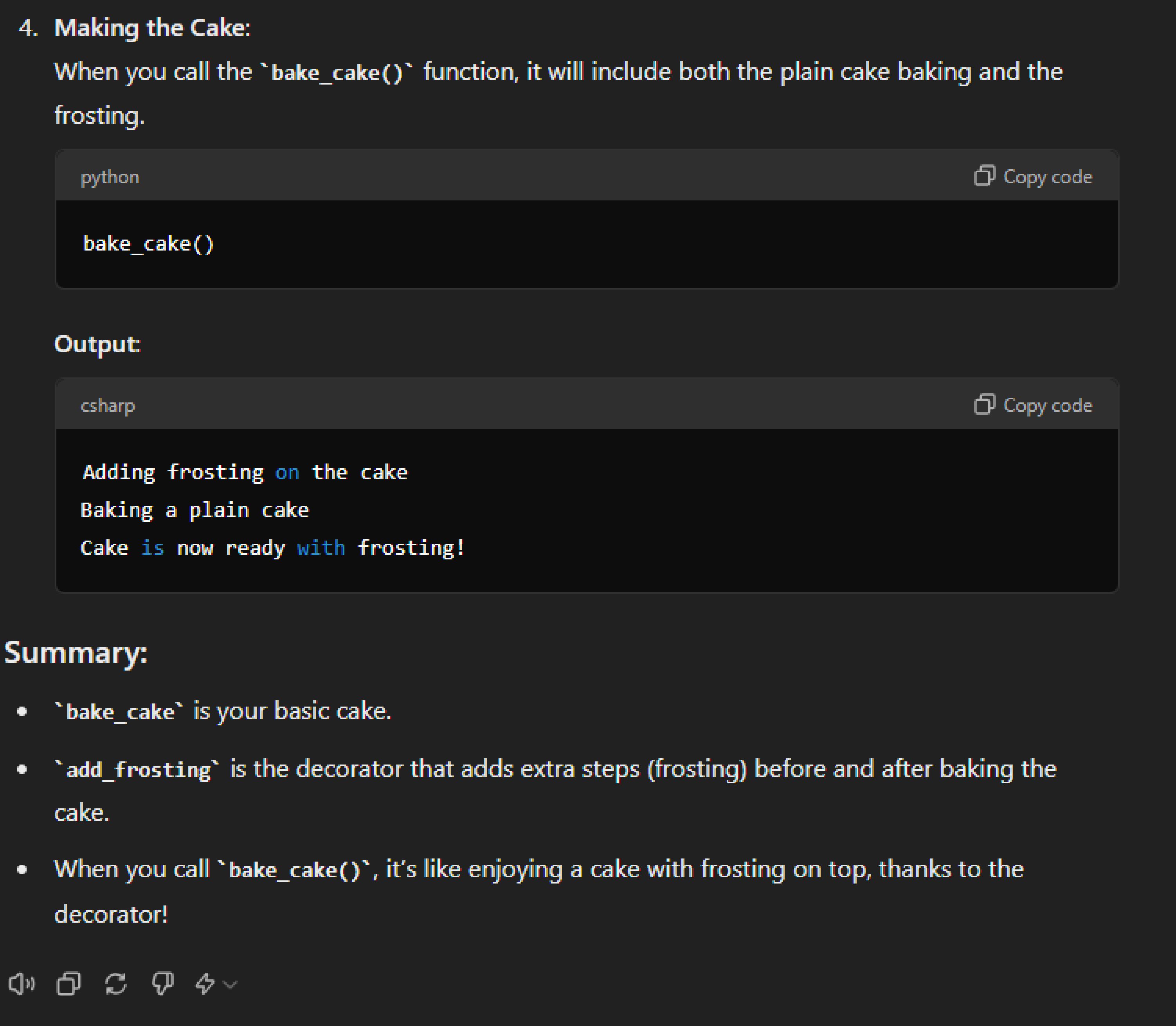
   Now, you want to add some frosting to make your cake more exciting. The frosting function is like a decorator that adds something extra to your cake (the original function).

```python
def add_frosting(cake):
    def decorated_cake():
        print("Adding frosting on the cake")
        cake()  # Call the original cake function
        print("Cake is now ready with frosting!")
    return decorated_cake
```

3. **Applying Frosting:**

   You use the `@` symbol to apply frosting to your cake, making it more special.

```python
@add_frosting
def bake_cake():
    print("Baking a plain cake")
```



4. **Making the Cake:**

   When you call the `bake_cake()` function, it will include both the plain cake baking and the frosting.

   ```python
   bake_cake()
   ```

   **Output:**

   ```csharp
   Adding frosting on the cake
   Baking a plain cake
   Cake is now ready with frosting!
   ```

## Summary:

- `bake_cake` is your basic cake.
- `add_frosting` is the decorator that adds extra steps (frosting) before and after baking the cake.
- When you call `bake_cake()`, it's like enjoying a cake with frosting on top, thanks to the decorator!