100 XP

Exercise - Test code with Copilot

5 minutes

Copilot's ability to generate unit tests is a great time-saver for developers. In this exercise, you'll use Copilot to generate unit tests for the Trie project. Let's get started!

Use Copilot to test code

In this exercise, you'll use Copilot to generate basic tests for the trie methods.

1. Open the **TrieTests.cs** file under **TrieDictionaryTest**
2. In the `TrieTest` class, enter the following code:

```c#
// Test that a word is inserted in the trie
[TestMethod]
```

3. Enter a new line, then wait for Copilot to generate code

   Copilot should generate code that inserts a word and asserts the word was inserted.

4. On a new line, enter the following code:

```c#
// Test that a word is deleted from the trie
[TestMethod]
```

5. Enter a new line, then wait for Copilot to generate code

   Copilot should generate code that inserts a word and asserts the word was deleted.

6. Repeat the steps with the following prompts:

```c#
// Test that a word is not inserted twice in the trie

// Test that a word is deleted from the trie
```

```
// Test that a word is not deleted from the trie if it is not present

// Test that a word is deleted from the trie if it is a prefix of an-
other word
```

As you continue generating test code, you may notice that Copilot autocompletes the prompt, the `[TestMethod]` stub, and the code. Copilot's ability to predict the code you need makes generating unit tests faster. Be sure to check that the generated assertions are correct.

However, Copilot may not always predict the exact test you need. You may need to review the generated code and make changes to the test, or use more specific prompting

7. On a new line, enter the following code comment:

```
c#
```

```
// Test AutoSuggest for the prefix "cat" not present in the
// trie containing "catastrophe", "catatonic", and "caterpillar"
```

8. Enter a new line and wait for Copilot to generate code

   Check that the generated test is similar to the following, making any changes as need:

```
c#
```

```
[TestMethod]
public void TestAutoSuggest()
{
    Trie dictionary = new Trie();
    dictionary.Insert("catastrophe");
    dictionary.Insert("catatonic");
    dictionary.Insert("caterpillar");
    List<string> suggestions = dictionary.AutoSuggest("cat");
    Assert.AreEqual(3, suggestions.Count);
    Assert.AreEqual("catastrophe", suggestions[0]);
    Assert.AreEqual("catatonic", suggestions[1]);
    Assert.AreEqual("caterpillar", suggestions[2]);
}
```

9. On a new line, enter the following code comment:

```
c#
```

```
// Test GetSpellingSuggestions for a word not present in the trie
```

10. Enter a new line and wait for Copilot to generate code

    Check that the generated test is similar to the following, making any changes as need:

    c#

```
[TestMethod]
public void TestGetSpellingSuggestions()
{
    Trie dictionary = new Trie();
    dictionary.Insert("cat");
    dictionary.Insert("caterpillar");
    dictionary.Insert("catastrophe");
    List<string> suggestions =
dictionary.GetSpellingSuggestions("caterpiller");
    Assert.AreEqual(1, suggestions.Count);
    Assert.AreEqual("caterpillar", suggestions[0]);
}
```

Check your work

In this task, you'll test the methods you created with Copilot to verify that they work as expected.

1. In the file explorer, right click the **TrieTest.cs** file and click **Open in Integrated Terminal**
2. Enter `dotnet test` to run the tests.
3. Verify that all tests pass.

    Output

```
Passed!  - Failed:     0, Passed:     7, Skipped:     0, Total:     7,
Duration: 638 ms - TrieDictionaryTest.dll (net7.0)
```

    If your tests produce different results, review your code to find the error and make updates. Run the code again to see if you've fixed the problem. Continue updating and running your code until your code produces the expected results.

Next unit: Module assessment

Previous Next