100 XP

Exercise - Debug code with Copilot Chat

10 minutes

Copilot Chat allows you to interact with Copilot using natural language. In this exercise, you'll use Copilot Chat to debug and improve code in the Trie project. Let's get started!

> **Important**
>
> This module requires the Github Copilot Chat extension, which is currently in public beta and subject to changes.

Use Copilot Chat to debug code

1. Open the **Trie.cs** file
2. Navigate to the `LevenshteinDistance` method

   The Levenshtein distance is a method for measuring the differences between two strings. The Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into the other. The edit operations include insertion, deletion, or substitution of a character.

   In this code, the Levenshtein distance is calculated incorrectly. Let's use Copilot Chat to fix the error.
3. Select all of the code for the `LevenshteinDistance` method
4. Right click the selected code. Under **Copilot**, click **Start Copilot Chat**
5. In the textbox, enter "Fix this code"

   Copilot Chat may ask "Sure, what seems to be the issue with the code?" If so, enter "The Levenshtein distance is calculated incorrectly"
6. Wait for Copilot to generate new code.

   Copilot should suggest code that fixes an off by one error.
7. Click **Accept** to apply the fix.

Use Copilot Chat to improve code

You can use Copilot Chat to improve code by adding new features or refactoring existing code. Let's use Copilot Chat to improve the `PrintTrie` method.

1. Open the **Program.cs** file
2. Navigate to the `PrintTrie` method

   This method prints the contents of the trie to the console.
3. Select all of the code for the `PrintTrie` method
4. Right click the selected code. Under **Copilot**, click **Start Copilot Chat**
5. In the textbox, enter "Improve this code by separating the words into five columns"
6. Wait for Copilot to generate the new code.

   Copilot should suggest code that prints the words in five columns.
7. Click **Accept** to apply the changes.

Check your work

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

1. Open the **Program.cs** file
2. Uncomment the `GetSpellingSuggestions` method call:

   c#

   ```
   Trie dictionary = InitializeTrie(words);
   // SearchWord();
   // PrefixAutocomplete();
   // DeleteWord();
   GetSpellingSuggestions();
   ```
3. In the file explorer, right click the **Program.cs** file and click **Open in Integrated Terminal**
4. Enter `dotnet run` to run the program.
5. Enter a misspelled word to generate spelling suggestions for, such as "cae"

6. Verify that your output is similar to the following

Output

```
The dictionary contains the following words:
as              cars            follows         monday          plans
astronaut       cares           from            monster         the
asteroid        careful         front           place           their
are             carefully       mellow          plan            they
around          for             mean            planet          there
cat             forgot          money           planets         towards

Enter a word to get spelling suggestions for, or press Enter to exit.
cae
Spelling suggestions for "cae":
cat
cars
cares
```

If your code displays different results, review your code to find your 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.

7. Comment out the `GetSpellingSuggestions` method call.

Next unit: Exercise - Test code with Copilot

Previous Next