```
import unittest
from my_module import generate_grid, get_user_words, get_words, get_pure_user_words

class TestGamsfunctions(unittest.TestCase):
    def test_generate_grid(self):
        grid = generate_grid(self):
        grid = generate_grid(self):
        self.assertEqual(len(grid()), 3)
        self.assertEqual(len(grid()), 3)
        self.assertEqual(len(grid()), 3)

def test_get_user_words(self):
    # Mocking user input for testing
    user_input = '\upper', 'pet', 'uproot', 'open', 'opto')
    expected_output = ('upper', 'pet', 'uproot', 'open', 'opto')
    with patch('builtins.input', side_effect-user_input):
        user_input
        self.assertEqual(user_words, expected_output)

def test_get_words(self):
    letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
    words_from_dict = get_words('en.txt', letters)
        self.assertEqual(len(words_from_dict), 12)
        self.assertIn('open', words_from_dict)
        self.assertIn('pent', words_from_dict)
        self.assertIn('pent', words_from_dict)
        self.assertIn('pent', words_from_dict)
        if Add more assertIons based on your expected dictionary words

def test_get_pure_user_words get_pure_user_words letters, words_from_dict)
        if words_from_dict = ['open', 'pent']
        pure_user_words = get_pure_user_words letters, words_from_dict)
        if __name__ =- '__main__':
        unittest.main[D]
```

Obviously, it messed up, as the task has become unbearable to it.

After investing a remarkable amount of effort in prompting, it managed to resolve its own code issues. What can we deduce from this? Undertaking challenging tasks often results in a failure to write comprehensive tests with Copilot. It seems unable to thoroughly analyze the code to generate reasonable tests.

```
ERROR: test_get_user_words (_main__.TestTargetGame.test_get_user_words)

Traceback (most recent call last):
File "/usr/lib/python3.1l/unittest/mock.py", line 1375, in patched return func(*newargs, **newkeyargs)

File "/neme/julfy/Documents/2nd_term/my_mini_projects/miniproject2/black_dungeon_mAIsters-1/test_targer_en.py", line 19, in test_get_user_words user_words = get_user_words()

Annowards = get_user_words()

Annowards = get_user_words()

Annowards = get_user_words()

Annowards = get_user_words()

File "/neme/julfy/Documents/2nd_term/my_mini_projects/miniproject2/black_dungeon_mAIsters-1/terget_en.py", line 13, in get_user_words string=input()

File "/usr/lib/python3.1l/unittest/mock.py", line 1124, in __call__
    return self_mock_call(*args, **wkargs)

File "/usr/lib/python3.1l/unittest/mock.py", line 1128, in _mock_call
    result = next(effect)

Annowards = mock_call(*args, **wkargs)

StopIteration

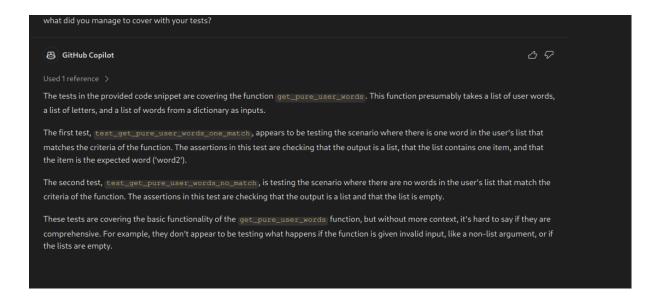
FAIL: test_get_pure_user_words (_main__.TestTargetGame.test_get_pure_user_words)

Traceback (most recent call last):
File "/nome/julfy/Documents/2nd_term/my_mini_projects/miniproject2/black_dungeon_mAIsters-1/test_targer_en.py", line 38, in test_get_pure_user_words self_assertEqual(len(pure_user_words), 1)

AssertionError: 0 != 1

Ban 6 tests in 0.096s
```

And it says something in here, but it doesn't matter when you don't have properly working code



It optimizes something, but it is not needed here. The code breaks