```
# import necessary modules
from unittest import mock


import pytest
from rich.console import Console
from rich.markdown import Markdown
from rich.rule import Rule


from swarms.utils import display_markdown_message




def test_basic_message():
    # Test basic message functionality
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message("This is a test")
        mock_print.assert_called_once_with(
            Markdown("This is a test", style="cyan")
        )




def test_empty_message():
    # Test how function handles empty input
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message("")
        mock_print.assert_called_once_with("")
```

```
@pytest.mark.parametrize("color", ["cyan", "red", "blue"])
def test_colors(color):
    # Test different colors
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message("This is a test", color)
        mock_print.assert_called_once_with(
            Markdown("This is a test", style=color)
        )


def test_dash_line():
    # Test how function handles "---"
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message("---")
        mock_print.assert_called_once_with(Rule(style="cyan"))


def test_message_with_whitespace():
    # Test how function handles message with whitespaces
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message(" \n Test \n --- \n Test \n")
        calls = [
            mock.call(""),
            mock.call(Markdown("Test", style="cyan")),
            mock.call(Rule(style="cyan")),
            mock.call(Markdown("Test", style="cyan")),
            mock.call(""),
        ]
        mock_print.assert_has_calls(calls)



def test_message_start_with_greater_than():
    # Test how function handles message line starting with ">"
    with mock.patch.object(Console, "print") as mock_print:
        display_markdown_message(">This is a test")
        calls = [
            mock.call(Markdown(">This is a test", style="cyan")),
            mock.call(""),
        ]
        mock_print.assert_has_calls(calls)
```