1. Purpose

The goal of this test plan is to make sure my BMI Calculator program works the way it is supposed to. I want to check that the BMI math is correct, the program handles bad input, and the categories are correct. I also want to make sure the program doesn't crash when the user types something unexpected.

2. What Is Being Tested

I will test the main parts of the program including the BMI calculation, the BMI category function, input validation, the printed results, the BMI table, and the loop that lets the user run the program again or quit.

3. Not Included

I am not testing performance, graphics, or different languages. This is a console program so I'm only testing basic functions and user input.

4. Test Items

The functions I will test are calculate_bmi(), bmi_category(), display_results(), display_bmi_table(), and main().

5. How I Will Test

Using unit testing, I will write simple pytest tests to check if BMI is calculated correctly, if the right category is returned, and if errors are raised when the input is bad. I will also run the program myself and manually try different inputs like negative numbers, letters instead of numbers, inches values outside the 0–11 range, and using the quit option at different places. I will also test values that are right on the edge of the boundaries to make sure they still work correctly.

6. Testing Environment

- Python 3
- VS Code
- pytest

7. Test Cases

Unit Test Cases

Test	What Is Being Checked	Expected Result
TC1	Normal BMI calculation	Correct BMI
TC2	Weight = 0	Error
TC3	Inches = 12	Error
TC4	BMI < 18.5	Underweight
TC5	BMI between 18.5 and 24.9	Normal weight

TC6	BMI >= 25	Overweight
TC7	BMI is not a number	Error

Manual / Functional Test Cases

TC8	Enter valid weight and height	BMI and category print
TC9	Type quit at weight prompt	Program exits
TC10	Enter negative feet	Error
TC11	Enter inches = 11	Accepted
TC12	Enter inches = 12	Error
TC13	Type yes to run again	Program repeats
TC14	Type no to run again	Program exits

8. Risks

The program might crash if input is not handled correctly. The loop might not stop if the user types something unexpected. BMI rounding might cause wrong categories

9. When Testing Is Done

Testing will be finished when all the tests pass, the program doesn't crash, all error messages show up correctly, and the BMI categories are correct.

```python
def test_normal_bmi():
    bmi = calculate_bmi(150, 5, 6)
    # Expected BMI ≈ 24.2
    assert round(bmi, 1) == 24.2

def test_weight_zero():
    with pytest.raises(ValueError):
        calculate_bmi(0, 5, 6)

def test_inches_too_high():
    with pytest.raises(ValueError):
        calculate_bmi(150, 5, 12)

def test_negative_feet():
    with pytest.raises(ValueError):
        calculate_bmi(150, -1, 5)

def test_non_numeric_weight():
    with pytest.raises(TypeError):
        calculate_bmi("abc", 5, 6)

def test_underweight_category():
    assert bmi_category(18.4) == "Underweight"

def test_normal_category():
    assert bmi_category(22) == "Normal weight"

def test_overweight_category():
    assert bmi_category(27) == "Overweight"

def test_bmi_not_numeric():
    with pytest.raises(TypeError):
        bmi_category("hello")

def test_bmi_zero_or_negative():
    with pytest.raises(ValueError):
        bmi_category(0)
```