

Github Repository Link: <https://github.com/Maxwell-Lam/BMICalculator>

Objective:

Apply test-driven development (TDD) to implement a set of software requirements. Write unit tests to provide adequate coverage of code-base using your chosen unit testing framework and test runner.

Project / Program Description:

The project is a web application that asks the user to input a height in inches and a weight in pounds. With these input values, the program will return a BMI value and their respective category. BMI values and their respective category are displayed in Figure 1.

BMI	Category
<18.5	Underweight
18.5–24.9	Normal weight
25–29.9	Overweight
>=30	Obese

Figure 1. Body Mass Index

The project has 2 main functions to properly determine the BMI and Category, shown in Figure 2.

```
functions.py > ...
1  def calculateBMI(totalInches, totalPounds):
2      totalKg = float(totalPounds) * 0.45
3      totalMeters = float(totalInches) * 0.025
4
5      totalMetersSquared = totalMeters * totalMeters
6      BMI = totalKg / totalMetersSquared
7      BMI = round(BMI, 2)
8      return(BMI)
9
10 def calculateWeightClass(BMI):
11     if (BMI < 18.5):
12         return "Underweight"
13     elif (BMI < 25.0):
14         return "Normal Weight"
15     elif (BMI < 30.0):
16         return "Overweight"
17     else:
18         return "Obese"
```

Figure 2 - Project Functions.

The first function “calculateBMI” takes in the input values of “totalInches” and “totalPounds” and formulates the numerical value of “BMI”. The second function “calculateWeightClass” takes the “BMI” calculated from the first function and return a string value of the appropriate weight class.

Note: The function given to us within the assignment link is either outdated or inaccurate. Expected outputs for this program should not be used for any medical decisions and are used only for the purposes for software testing. All decisions are centered around these functions whether or not they are accurate or not.

Testing Setup:

Since there are 2 possible ranges of user input, we will use 2 Weak Nx1 types tests. Weight and Height will be tested be changed independently from each other. As we test height inputs, weight will be predetermined based on this graph and vice versa. We will use the axes of Weight = 150lbs for 150lbs have all weight categories, and Height = 5’7” being generally agreed upon to be the average height of a human adult.

There are 4 categories of weight classes: Underweight, Normal, Overweight and Obese. Because any weight with any height can fall to any of these categories, each independent array of inputs will be partitioned. We can determine the boundaries with algebra. Test cases will test these ranges below.

For 150lbs:

- Underweight ranges over 76 inches
- Normal ranges from 76 to 66 inches
- Overweight ranges from 65 to 61 inches
- Obese ranges under 61 inches

For 5’7” (67 inches)

- Underweight ranges under 116 lbs.
- Normal ranges from 116 to 155lbs.
- Overweight ranges from 156 to 187 lbs
- Obese ranges over 187 lbs

I would like one point within each boundary for the 3 boundaries, giving us 7 test cases ((1 + 1) x 3 + 1 test cases) per input array (height and weight inputs), giving us a total of 14 test cases minimum. For the boundary test, each tests case will test an ON, OFF and the boundary point per shared boundary area.

Boundary Shift:

For testing purposes, I induce a boundary shift by 0.1 at the lower boundary of “Normal Weight”. Below is a screenshot of this temporary change:

```
def calculateWeightClass(BMI):  
    if (BMI < 18.6):  
        return "Underweight"  
    elif (BMI < 25.0):  
        return "Normal Weight"  
    elif (BMI < 30.0):  
        return "Overweight"  
    else:  
        return "Obese"
```

Test Results:

Normal Results (No Boundary Shift):

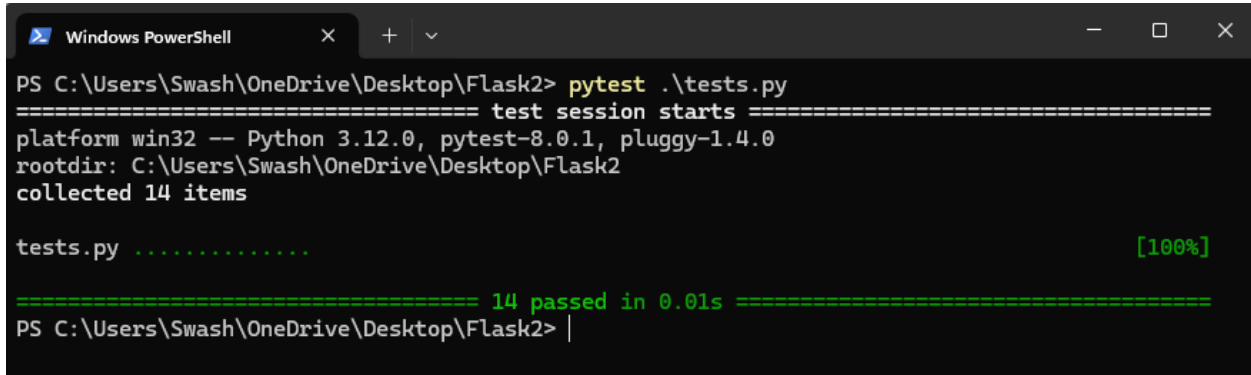
All test cases pass - Expected Output

Boundary Shift Results:

All test cases pass - Not Expected Output

My test cases did not catch this boundary shift problem because the results show that all test pass. They did not catch this boundary shift problem is because the test cases uses whole integer inputs, making it harder (or not possible) to detect a smaller change in range.

Normal Test Cases Results:



```
Windows PowerShell  
PS C:\Users\Swash\OneDrive\Desktop\Flask2> pytest .\tests.py  
===== test session starts =====  
platform win32 -- Python 3.12.0, pytest-8.0.1, pluggy-1.4.0  
rootdir: C:\Users\Swash\OneDrive\Desktop\Flask2  
collected 14 items  
  
tests.py ..... [100%]  
  
===== 14 passed in 0.01s =====  
PS C:\Users\Swash\OneDrive\Desktop\Flask2> |
```

Boundary Shift Test Cases Results:

```
Windows PowerShell
PS C:\Users\Swash\OneDrive\Desktop\Flask2> pytest .\tests.py
===== test session starts =====
platform win32 -- Python 3.12.0, pytest-8.0.1, pluggy-1.4.0
rootdir: C:\Users\Swash\OneDrive\Desktop\Flask2
collected 14 items

tests.py ..... [100%]

===== 14 passed in 0.01s =====
PS C:\Users\Swash\OneDrive\Desktop\Flask2> |
```

Addition Screenshots:

Underweight Input Front Page:

Welcome to the BMI Calculator!

Enter height in inches :

Enter weight in pounds:

Underweight Result Page:

Based on our advanced calculations.

Your BMI is : 16.88

You are: Underweight

Normal Weight Result Page: (Height: 71, Weight: 150)

Based on our advanced calculations.

Your BMI is : 21.42

You are: Normal Weight

Overweight Result Page: (Height: 63, Weight: 150)

Based on our advanced calculations.

Your BMI is : 27.21

You are: Overweight

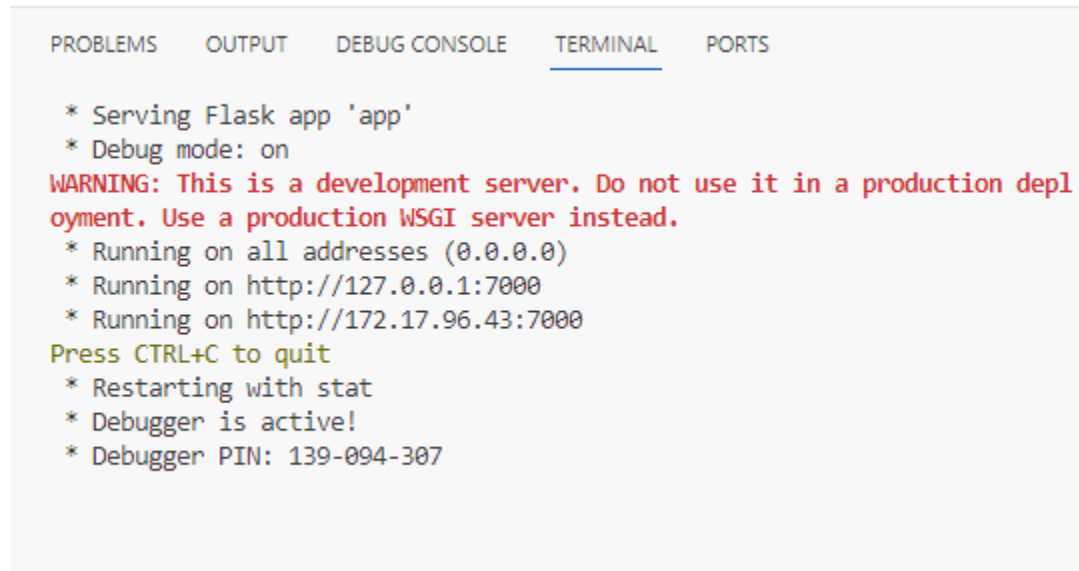
Obese Result Page: (Height: 50, Weight: 150)

Based on our advanced calculations.

Your BMI is : 43.2

You are: Obese

VS Code Terminal (Application Running):

A screenshot of the VS Code terminal interface. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected), and PORTS. The output text in the terminal is as follows:

```
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:7000
* Running on http://172.17.96.43:7000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 139-094-307
```

Addition Information:

Used this link for reference for project:

<https://www.geeksforgeeks.org/flask-http-method/?ref=lbp>

BMI calculations given to us are different from the calculators from other online sources:

Ex: https://www.nhlbi.nih.gov/health/educational/lose_wt/BMI/bmi-m.htm

Detail Setup and execution Instructions:

Software Needed:

- Python: <https://www.python.org/downloads/>
- Flask: <https://pypi.org/project/Flask/>
- Python (Pytest) : <https://pypi.org/project/pytest/>
- Visual Studio Code: <https://code.visualstudio.com/download>
- Windows Command Terminal (Or similar software)

Instructions:

1. Install software.
2. Open folder in Visual Studio Code and open/navigate to “app.py”.
3. Run “app.py”
4. Open any of the open addresses displayed in the VS Code terminal:
 - a. Web Application is now online.
5. Input your height and weight and click “Submit” to see BMI results.