William Giddens

Wgg51

GitHub: WilliamGiddens

In our python file, we defined our class by APP. The APP class will help determine whether to run the BMI function or the Retirement function.

As mention, the first function is the BMI and when prompted, it will run by asking your height and weight. In the background it is running the calculations outlined by BMI formula. Once the calculations are done, it will output your BMI and display your category. Your category describes whether or not you are underweight, normal weight, overweight, or obese.

The second function is the Retirement function. When prompted, it will ask for your age, salary, savings percentage, desired retirement age, and retirement savings goal. Using basic math calculations, it will tell you whether or not you will meet your retirement goal based on your current trajectory.

Questions:

Q1: How do you know functions tests are complete/thorough enough?

A1: We know our functions are complete/thorough enough by testing if there are calculation errors, input errors, or any other general errors. If not, we should be off to a good start.

Q2: What testing strategies did you use?

A2: One of the strategies I used to test the program was inputting things that were not recommended by the program. Sometimes the program still ran even though I technically didn't put in the right thing. Other times, the program shut down. It helps you figure out whether or not your program is stable enough under stress.

Q3: How did you apply them (explain)?

A3: For example, when the user is prompted to pick whether or not they want to use the BMI calculator or the Retirement calculator, instead of just picking 1 or 2, I would pick some random number. This caused the program to still run but would technically still go with 2. So, the only way to get the BMI calculator is to put in 1 exactly. Although, no matter what number or letter I put in, the Retirement function would execute every time.

Q4: Discuss logic/motivation for each test (1-3 sentences)

A4: The logic behind this testing style is to put your program under stress. Obviously, it's great if the user does what you intend them to do every time, but we should be prepared for user error. Coincidentally, my program needed to account for that more.

Detailed setup and execution instructions:

So, I used PyCharm IDE to develop the program. You can download PyCharm at https://www.jetbrains.com/pycharm/ and make sure to get the community version. Once you have that done, open the python file and run it in the terminal. I used Python 3.9, so I am not sure if you can get away with using an older version. If not, download and install the newer version at https://www.python.org/downloads/release/python-391/. Once you run the program, respond to the prompts on the screen and it should give you the information required. I ran it on Windows 10, but you may be able to run it on other OSs. I haven't tried to. To access the program file, you need to use the GitHub repository at https://github.com/WilliamGiddens/Assignment2/tree/main.

Screenshots:

```
 Please enter your selection: enter 1 or 2
1


================================================

 How tall are you? Enter feet and inch separated by a space: 5 8
 Enter your weight(lbs): 190
BMI= 29.584775086505186
29.584775086505186 Overweight
```

Description → In this screenshot, I am testing whether or not BMI function is running correctly if the user inputs correctly. The test has passed.

```
 Enter your current age: 22
Enter your yearly-salary in dollars: 32000
Note : Now please enter the savings percentage
 For example:  20. means you want to save 20% of your salary
Enter your planned savings percentage now :20
 Tell us about your retirement plan!
 Note that we assume you will live to 100 years old,
 So think about your retirement goal (dollar amount) for a while!
Please enter your desired retirement age: 66
 Please enter retirement-saving goal (dollar amount): 1000000

 You will have saved : 380160.0 dollars at the planned retiring age!
Sorry, You won't meet your goal
 you need to save more or increase your salary
```

Description → In this screenshot, I am testing whether or not the Retirement function is running correctly if the user inputs correctly. The test has passed.

```
 Please enter your selection: enter 1 or 2
455y2

================================================

 Enter your current age: 3433
Enter your yearly-salary in dollars: sddwq
Traceback (most recent call last):
  File "C:/Users/willt/Downloads/app_will_2.py", line 77, in <module>
    ans=app1.Retirement()
  File "C:/Users/willt/Downloads/app_will_2.py", line 36, in Retirement
    salary = float(input('Enter your yearly-salary in dollars: '))
ValueError: could not convert string to float: 'sddwq'
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs incorrectly. At first, the program runs fine even if I input unrealistic numbers. Where it falls apart is when I eventually add a letter where it wants a number. Which is good, but the program should have stopped a lot sooner. Therefore, the test has failed.

```
 Please enter your selection: enter 1 or 2
1

================================================

 How tall are you? Enter feet and inch separated by a space: 100 8
 Enter your weight(lbs): -12
BMI= -0.0059207929476777325
-0.0059207929476777325 Underweight
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs unrealistic or negative numbers. The program runs fine even if I input unrealistic numbers or negative numbers. Although, no one should ever input these kinds of numbers, it still outputs the BMI calculation. Therefore, the test has passed.

```
 Please enter your selection: enter 1 or 2
1

================================================

 How tall are you? Enter feet and inch separated by a space: -19
Traceback (most recent call last):
  File "C:/Users/willt/Downloads/app_will_2.py", line 74, in <module>
    BMI_v, cat = app1.Body_mass_index()
  File "C:/Users/willt/Downloads/app_will_2.py", line 13, in Body_mass_index
    height = int(list1[0]) * 12 + int(list1[1])
IndexError: list index out of range
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs a non-spaced number in the “how tall are you.” The program stops once the user inputs wrong. Therefore, the test has passed.

```
 Enter your current age: -10
Enter your yearly-salary in dollars: -111
Note : Now please enter the savings percentage
 For example:  20. means you want to save 20% of your salary
Enter your planned savings percentage now :-1
 Tell us about your retirement plan!
 Note that we assume you will live to 100 years old,
 So think about your retirement goal (dollar amount) for a while!
Please enter your desired retirement age: -22
 Please enter retirement-saving goal (dollar amount): -0

 You will have saved : -17.982 dollars at the planned retiring age!
Sorry, You won't meet your goal
 you need to save more or increase your salary
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs negative number in the Retirement function. The program continues to run and even outputs a negative result. Although there is no reason a user should actually input these numbers, it still works. Therefore, the test has passed.

```
 Please enter your selection: enter 1 or 2
2

================================================

 Enter your current age: y
Traceback (most recent call last):
  File "C:/Users/willt/Downloads/Assignment2CalculatorWGG51.py", line 77, in <module>
    ans=app1.Retirement()
  File "C:/Users/willt/Downloads/Assignment2CalculatorWGG51.py", line 35, in Retirement
    age = float(input(' Enter your current age: '))
ValueError: could not convert string to float: 'y'

Process finished with exit code 1
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs a letter in the "enter your age." The program shuts down. The program should not continue because the user should input an actual number. Therefore, the test has passed.

```
This App can do some calculations for you!
 Please select from the following two options:
 (1) Compute your Body-mass-index for you
 (2) Check your retirement plan
=====================================================

 Please enter your selection: enter 1 or 2
BMI

================================================

 Enter your current age:
```

Description → In this screenshot, I am testing whether or not the program is running correctly if the user inputs "BMI" when asked to select which function to run. Although the user inputted BMI, the Retirement function begins to run. This could cause confusion for the user. Therefore, the test has failed.