Reflection: Part 1

Approaching the former section of MP1, I knew my first step would require me to set a variable for the text comprising the Bliss autograph of the Gettysburg Address. As a string spanning multiple lines in Python, I knew identifying this variable in full would require triple quotes at the beginning and end of its associated string, which did indeed allow me to set the variable, entitled ***bliss***. Once that was written, I researched case folding via StackOverflow, in turn yielding a number of viable candidates for implementation, the most concise being a ***for*** statement within which I wrote the following: for all ***words*** in ***bliss,*** fold ***words*** into the lowercase then identify that variable again as ***bliss***. I recognized from past experience that it helps to test out the viability of newly incorporated practices like this one, so I went ahead and printed ***bliss*** in its newly identified form, which displayed a lowercase version of the Gettysburg Address as I'd hoped for. I then created the variable for vowels as ***vows***, comprised of a list in which 'a', 'e', 'i', 'o', 'u' would appear. Moving forward, I had some trouble dealing with an unsupported operand type for integers and strings but realized that this error was due to the fact that I hadn't explained to Python that each vowel within ***bliss*** should be equal to the integer of 1. I did my fair share of researching as to how I might resolve this problem without nesting multiple ***for*** and ***if*** statements within one another, and I came across the method of identifying the integer of 1 to a variable (i.e. ***v***) in ***bliss*** if ***v*** also exists within ***vows.*** This approach entailed that I declare the ***for*** and ***if*** statements together inside of a list, which in turn would operate inside of a set, whose function would be ***sum*** in order to compile each iteration of ***v*** given those conditionals — all of which I identified as ***vowel_count***. Ultimately, my final line consisted of printing ***vowel_count***, which then yielded 449.

Begin code:

```
bliss = """Four score and seven years ago our fathers brought forth on this
continent, a new nation, conceived in liberty, and dedicated to the
proposition that all men are created equal. Now we are engaged in a
great civil war, testing whether that nation, or any nation so conceived
and so dedicated can long endure. We are met on a great battle-field of
that war. We have come to dedicate a portion of that field, as a final
resting place for those who here gave their lives that that nation might
live. It is altogether fitting and proper that we should do this. But,
in a larger sense, we can not dedicate---we can not consecrate---we can
not hallow---this ground. The brave men, living and dead, who struggled
here, have consecrated it, far above our poor power to add or detract.
The world will little note, nor long remember what we say here, but it
can never forget what they did here. It is for us the living, rather, to
be dedicated here to the unfinished work which they who fought here have
thus far so nobly advanced. It is rather for us to be here dedicated to
the great task remaining before us---that from these honored dead we
take increased devotion to that cause for which they gave the last full
measure of devotion---that we here highly resolve that these dead shall
not have died in vain---that this nation, under God, shall have a new
birth of freedom---and that government of the people, by the people, for
the people, shall not perish from the earth."""

for words in bliss:
        bliss = bliss.lower()

vows = ['a', 'e', 'i', 'o', 'u']

vowel_count = sum([1 for v in bliss if v in vows])

print(vowel_count)
```

Reflection: Part 2

In approaching the latter section of MP2, I believe the first step was the one that took me the longest to surpass, if only because it required setting the correct parameters among the ***for*** statement. To begin, that is, I impulsively wrote the following ***for*** statement — "for ***num*** in (1 - 100):" — which I quickly realized warranted a ***range ()*** function in order for Python to plot out the variables between 1 and 100. I also recognized that I needed a comma rather than a dash, and that I need the full range to be 1 and 101 in order for the number of 100 to be inclusive among my output. Afterward, my next problem was formatting my Python code so that "fizzbuzz" would be printed for all multiples of both 3 and 5 to be included, because my preexisting code would stop computing with multiples of 3, rather than of 3 *and* 5. In order to fix this problem, and knowing that computed multiples of 3 and 5 would together equal zero, I rewrote my code so that when Python combined these two figures to a sum of zero, the output would be consequently print "fizzbuzz." Following this step, the rest of the process was straightforward. I wrote two ***elif*** statements for "fizz" and "buzz," then included an ***else*** statement for all the remaining numbers that were neither multiples of 3 or 5, which would print those number as their basic integers.

Begin code:

```
for num in range(1, 101):
	if (num % 3) + (num % 5) == 0:
		print('fizzbuzz')
	elif num % 3 == 0:
		print('fizz')
	elif num % 5 == 0:
		print('buzz')
	else:
		print(num)
```