1. Create an assert statement that throws an AssertionError if the variable spam is a negative integer.

In [3]:

```
spam=-2
assert spam>=0,"Spam must be positive"
```

```
---------------------------------------------------------------------------
-
AssertionError                            Traceback (most recent call las
t)
Cell In[3], line 2
      1 spam=-2
----> 2 assert spam>=0,"Spam must be positive"

AssertionError: Spam must be positive
```

2. Write an assert statement that triggers an AssertionError if the variables eggs and bacon contain strings that are the same as each other, even if their cases are different (that is, 'hello' and 'hello' are considered the same, and 'goodbye' and 'GOODbye' are also considered the same).

In [5]:

```
eggs="Ali"
bacon="ali"
assert eggs.lower()!=bacon.lower(),"Strings are same"
```

```
---------------------------------------------------------------------------
-
AssertionError                            Traceback (most recent call las
t)
Cell In[5], line 3
      1 eggs="Ali"
      2 bacon="ali"
----> 3 assert eggs.lower()!=bacon.lower(),"Strings are same"

AssertionError: Strings are same
```

3. Create an assert statement that throws an AssertionError every time.

In [6]:

```
1 assert False,"This always arrise AssertionError every time"
```

```
---------------------------------------------------------------------------
-
AssertionError                            Traceback (most recent call las
t)
Cell In[6], line 1
----> 1 assert False,"This always arrise AssertionError every time"

AssertionError: This always arrise AssertionError every time
```

4. What are the two lines that must be present in your software in order to call logging.debug()?

logging must be imported level must be set to DEBUG

import logging

logging.basicConfig(filename="filename.log",level=logging.DEBUG)

logging.debug("now we can use this")

5. What are the two lines that your program must have in order to have logging.debug() send a logging message to a file named programLog.txt?

import logging

logging.basicConfig(filename="programLog.txt",level=logging.DEBUG)

logging.debug("now we can sned debug message to programLog.txt")

6. What are the five levels of logging?

1.CRITICAL 2.ERROR 3.Warning 4.Info 5.DEBUG

7. What line of code would you add to your software to disable all logging messages?

In [ ]:

```
import logging
logging.disable(logging.CRITICAL)
```

8.Why is using logging messages better than using print() to display the same message?

IN production environment print() is not alloed .Logging is used to track the execution of the program without effecting the flow of the program loging is used instead of print().

9. What are the differences between the Step Over, Step In, and Step Out buttons in the debugger?

The Step Over button allows you to execute the current line of code and move to the next line without stepping into any function calls on that line. The Step In button allows you to enter into the function call on the current line. Executes the remaining lines of the current function and returns to the calling function, pausing at the line after the current function's call.

10.After you click Continue, when will the debugger stop ?

The Debugger will stop after program completion or at breakpoint or ocuurence of Exception

11. What is the concept of a breakpoint?

If there is a breakpoint set at a specific line of code, the debugger will stop when it reaches that line during program execution.