Challenge: Python Stack Trace Interpretation

```
Traceback Problem 1
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 45, in <lambda>
    run_trace(1, lambda: perform_calculation(add, '1', 3))
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 12, in add
    return x + y
TypeError: can only concatenate str (not "int") to str
```

Answer:
In Python, the value you are concatenating needs to be the same type, both int or str. However, JavaScript doesn't look at this rule.

```
Traceback Problem 2
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 46, in <lambda>
    run_trace(2, lambda: perform_calculation(add, 7, '3'))
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 12, in add
    return x + y
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Answer:
The plus “+” is used to add two numbers as well as two strings. The arithmetic addition of the two numbers is for numbers. For strings, two strings are concatenated. The string should be converted to an integer before it is added to another number.

```
Traceback Problem 3
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
```

```
    f()
  File "stack_traces.py", line 47, in <lambda>
    run_trace(3, lambda: perform_calculation(mult, '3', '3'))
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 15, in mult
    return x * y
TypeError: can't multiply sequence by non-int of type 'str'
```

Answer:
This issue occurs if you try to multiply two string values together. You can fix this problem by ensuring that you either multiply two numerical values together or that you only multiply a string by an integer.

```
Traceback Problem 4
===================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 48, in <lambda>
    run_trace(4, lambda: perform_calculation(mult, [4], [3]))
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 15, in mult
    return x * y
TypeError: can't multiply sequence by non-int of type 'list'
```

Answer:
This problem is similar to prev problem except that it uses a different data type. It can fix by rewriting “[3]” to 3 as a number at line 48.

```
Traceback Problem 5
===================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 49, in <lambda>
    run_trace(5, lambda: perform_calculation(innoc, '1', 3))
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 22, in innoc
    spelunk()
```

```
  File "stack_traces.py", line 21, in spelunk
    raise ValueError('Invalid')
ValueError: Invalid
```

Answer:
It is an issue with the exception of "spelunk()" method. Probably, the condition is not valid at line 21.

Traceback Problem 6

```
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 50, in <lambda>
    run_trace(6, lambda: comp_calc([1, 2, 3], 1, add))
  File "stack_traces.py", line 30, in comp_calc
    return [perform_calculation(calc, x_i, y_i) for x_i, y_i in zip(x, y)]
TypeError: zip argument #2 must support iteration
```

Answer:
The "zip()" funtion is an iterator. The second passed argument is not iterable. Type such as array and string is iterable. However, the type of second passed argument is number.

Traceback Problem 7

```
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 51, in <lambda>
    run_trace(7, lambda: comp_calc([1, 2, [3]], [4, 5, 6], add))
  File "stack_traces.py", line 30, in comp_calc
    return [perform_calculation(calc, x_i, y_i) for x_i, y_i in zip(x, y)]
  File "stack_traces.py", line 30, in <listcomp>
    return [perform_calculation(calc, x_i, y_i) for x_i, y_i in zip(x, y)]
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 12, in add
    return x + y
TypeError: can only concatenate list (not "int") to list
```

Answer:
The "zip()" funtion is an iterator. The second passed argument is not iterable. Type such as array and string is iterable. However, the type of second passed argument is number.

Traceback Problem 8

```
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 52, in <lambda>
    run_trace(8, lambda: calc_dict({'one': 1, 'two': '2'}, 'one', 'two', add))
  File "stack_traces.py", line 26, in calc_dict
    return perform_calculation(calc, d[k1], d[k2])
  File "stack_traces.py", line 8, in perform_calculation
    calc(x, y)
  File "stack_traces.py", line 12, in add
    return x + y
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Answer:

This issue is kind of problem 2, but this error is caused by the data types being mismatched for the dictionary keys “one” and “two”. One of the ways to fix the issue is to decide if the operation is supposed to be “arithmetic” or “stringy” and then change the argument inputs as needed at line 52.

Traceback Problem 9

```
==================
Traceback (most recent call last):
  File "stack_traces.py", line 36, in run_trace
    f()
  File "stack_traces.py", line 53, in <lambda>
    run_trace(9, lambda: calc_dict({}, 'one', 'two', add))
  File "stack_traces.py", line 26, in calc_dict
    return perform_calculation(calc, d[k1], d[k2])
KeyError: 'one'
```

Answer:

The key “one” in the dictionary is dismissed. However, the dictionary at line 53 is indeed empty, so it should be filled with {key: value} pairs for containing data for the keys “one” and “two”.