

Debug_1

The screenshot displays the PyCharm IDE interface with a project named 'smilingFaces'. The main editor shows a file 'test_debug1.py' containing a Caesar cipher implementation. The code includes a function 'caesar' that takes a string and a shift value, and a 'test_caesar' function that tests the cipher with various inputs. The code is as follows:

```
70 represents a valid encoding of the original
71 string using a Caesar cipher.
72 Returns -1 otherwise.
73
74
75 # Use built-in ord() function to get ASCII
76 # integer value associated with A-Z.
77 # A has value 65, B is 66, ..., Z is 90.
78
79 # Get shift for first character. All characters must
80 # have identical shift for it to be a valid Caesar shift.
81 shift = ord(codeword[0]) - ord(original[0]) shift: 25
82
83 for idx in range(len(codeword)):
84     if ord(codeword[idx]) - ord(original[idx]) != shift:
85         return -1
86
87 return shift
88
89
90
91 def test_caesar():
92     print(caesar('ABCD', 'GHIJ')) #pass
93     print(caesar('ABCD', 'ZAVD')) #fail
94     print(caesar('ABCD', 'WXYZ')) #fail
95     print(caesar('ABCD', 'WXYZ')) #fail
96     print(caesar('ABCD', 'ZIXW')) #fail
97     print(caesar('AAAA', 'BBBB')) #fail
98     print(caesar('WXYZ', 'CDEF')) #fail
99     print(caesar('ZABC', 'FGHI')) #fail
100     print(caesar('PQRS', 'VWXY')) #pass
101
102 test_caesar()
103
104 caesar()
```

The bottom panel shows the 'Debugger' window with the 'Variables' tab selected. It displays the following variables:

- codeword = (str) 'ZAVD'
- original = (str) 'ABCD'
- shift = (int) 25

The 'Event Log' on the right shows a message: 'IDE and Plugin Updates: PyCharm Community Edition is ready to update.' The status bar at the bottom indicates '102:1 CRLF UTF-8' and the time '21:12 07-10-2017'.

Debug_2

smilingFaces - [C:\Users\ayush\PycharmProjects\smilingFaces] - ...test_debug2.py - PyCharm Community Edition 2017.2.2

File Edit View Navigate Code Refactor Run Tools VCS Window Help

smilingFaces > test_debug2.py

Project

- smilingFaces
 - art_show.py
 - card.py
 - club.py
 - Fruitful.py
 - Higgs_Boson.py
 - hw5.py
 - muddle.py
 - polygons.py
 - rain.py
 - simple_design.py
 - test_debug1.py
 - test_debug2.py
 - week2.py
 - week4_hw.py
 - zigzag.py
 - zigzagBW.py
- External Libraries
 - Python 3.6.2 (C:\Users\ayush\AppData\Local\Programs\Python\Python36-32\lib\site-packages)
 - Extended Definitions
 - Binary Skeletons
 - Python36-32 library root
 - DLLs
 - Lib
 - site-packages
 - Typeshed Stubs

```
49
50
51 def match(string1, string2): string1: 'hello' string2: 'ello'
52 """
53 Identifies and returns the length of a longest
54 common consecutive substring shared
55 by the two input strings.
56 :param string1: The first string.
57 :param string2: The second string.
58 :return: length of a longest shared consecutive string.
59 """
60
61 best_length = 0 best_length: 0
62 # for all possible string1 start points
63 for idx1 in range(len(string1)-1): idx1: 1
64     # for all possible string2 start points
65     for idx2 in range(len(string2)-1): idx2: 0
66         # check if these characters match
67         if string1[idx1] == string2[idx2]:
68             this_match_count = 1 this_match_count: 0
69             # see how long the match continues
70             while string1[idx1 + this_match_count] == \
71                 string2[idx2 + this_match_count]:
72                 this_match_count += 1
73
74         # compare to best so far
75         if this_match_count > best_length:
76             best_length = this_match_count
77
78 # now return the result
79 return best_length
80
81 def test_before_bugfixes():
82     print(match("hello", "ello"))
83     print(match("GaBeN", "ABeN"))
84
85 match() > for idx1 in ran... > for idx2 in ran... > if string1[idx1...
```

Debugger test_debug2

Frames

- MainThread
- match, test_debug2.py:70
- test_before_bugfixes, test_debug2.py:82
- <module>, test_debug2.py:85
- execfile, _pydev_execfile.py:18
- run, pydevd.py:1023
- <module>, pydevd.py:1596

Variables

- best_length = (int) 0
- idx1 = (int) 1
- idx2 = (int) 0
- string1 = (str) 'hello'
- string2 = (str) 'ello'
- this_match_count = (int) 2

Event Log

83:34 CRLF UTF-8 22:15 07-10-2017

Test_suites1

The screenshot displays the PyCharm IDE interface with the following components:

- Project View:** Shows the project structure for 'smilingFaces'. The file 'test_debug1.py' is selected under the 'Project' tab.
- Code Editor:** Contains the implementation of the 'test_caesar' function. The function takes a 'codeword' as input and returns an integer representing a valid encoding of the original string using a Caesar cipher. It includes a docstring and a test suite.
- Run Console:** Shows the output of the test suite. The tests are labeled with '6' and '22', indicating they passed. The console output is: 'Process finished with exit code 0'.
- Event Log:** Shows a message from PyCharm: 'IDE and Plugin Updates: PyCharm Community Edition is ready to update.'.

```
68 def test_caesar():
69     """
70     :param codeword: The encoded string.
71     :return: Integer in the range 0-25 if the codeword
72     represents a valid encoding of the original
73     string using a Caesar cipher.
74     Returns -1 otherwise.
75     """
76     # Use built-in ord() function to get ASCII
77     # integer value associated with A-Z.
78     # A has value 65, B is 66, ..., Z is 90.
79     # Get shift for first character. All characters must
80     # have identical shift for it to be a valid Caesar shift.
81     shift = ord(codeword[0]) - ord(original[0])
82
83     for idx in range(len(codeword)):
84         if ord(codeword[idx]) - ord(original[idx]) != shift:
85             return -1
86
87     return shift
88
89 def test_caesar():
90     print(caesar('ABCD', 'GHIJ')) #pass
91     print(caesar('ABCD', 'ZAVD')) #fail
92     print(caesar('ABCD', 'YMXK')) #fail
93     print(caesar('ABCD', 'WXYZ')) #fail
94     print(caesar('ABCD', 'ZXXN')) #fail
95     print(caesar('AAAA', 'BBBB')) #fail
96     print(caesar('WXYZ', 'CDEF')) #fail
97     print(caesar('ZABC', 'FGHI')) #fail
98     print(caesar('PQRS', 'VWXY')) #pass
99
100 t = test_caesar()
101
```

Test suites2

smilingFaces - [C:\Users\ayush\PycharmProjects\smilingFaces] - ...test_debug2.py - PyCharm Community Edition 2017.2.2

File Edit View Navigate Code Refactor Run Tools VCS Window Help

smilingFaces test_debug2.py

Project

- smilingFaces C:\Users\ayush\PycharmProjects\smilingFaces
 - art_show.py
 - card.py
 - club.py
 - Fruitful.py
 - Higgs_Boson.py
 - hw5.py
 - muddle.py
 - polygons.py
 - rain.py
 - simple_design.py
 - test_debug1.py
 - test_debug2.py
 - week2.py
 - week4_hw.py
 - zigzag.py
 - zigzagBW.py
- External Libraries
 - Python 3.6.2 (C:\Users\ayush\AppData\Local\Programs\Python\Python36-32\python.exe)
 - Extended Definitions
 - Binary Skeletons
 - Python36-32 library root
 - DLLs
 - Lib
 - site-packages
 - Typeshed Stubs

```
54 common consecutive substring shared
55 by the two input strings.
56 :param string1: The first string.
57 :param string2: The second string.
58 :return: length of a longest shared consecutive string.
59 """
60
61 best_length = 0
62 # for all possible string1 start points
63 for idx1 in range(len(string1)-1):
64     # for all possible string2 start points
65     for idx2 in range(len(string2)-1):
66         # check if these characters match
67         if string1[idx1] == string2[idx2]:
68             this_match_count = 1
69             # see how long the match continues
70             while string1[idx1 + this_match_count] == \
71                 string2[idx2 + this_match_count]:
72                 this_match_count += 1
73
74             # compare to best so far
75             if this_match_count > best_length:
76                 best_length = this_match_count
77
78 # now return the result
79 return best_length
80
81 def test_before_bugfixes():
82     print(match("hello", "ello"))
83
84 if __name__ == "__main__":
85     test_before_bugfixes()
```

Run: test_debug2 test_debug2

Traceback (most recent call last):

- File "C:\Users\ayush\PycharmProjects\smilingFaces\test_debug2.py", line 85, in <module>: test_before_bugfixes()
- File "C:\Users\ayush\PycharmProjects\smilingFaces\test_debug2.py", line 82, in test_before_bugfixes: print(match("hello", "ello"))
- File "C:\Users\ayush\PycharmProjects\smilingFaces\test_debug2.py", line 70, in match: while string1[idx1 + this_match_count] == \

IndexError: string index out of range

Event Log

07-10-2017 17:08 IDE and Plugin Updates: PyCharm Community Edition is ready to update.

For test_debug_1:

Questions:

(a) Give 2 examples of inputs for which the provided code gives a correct answer despite the fact that it is flawed.

- `caesar('ABCD', 'GHIJ')`
- `caesar('PQRS', 'VWXY')`

(b) For each example, explain why the faulty code produced the correct answer, despite the flaw(s).

Solution: This is because, in each case, shift didn't go into negative int values.

(c) Describe the bug(s) present in the code, and for each bug, indicate what the fix is.

Solution: The function returned negative values because it did not have a condition for backward shifts where it returned negative values for backward shifts.

Fix: Added a condition that whenever it does backward shift, that is whenever the shift values gets less than 0, it changes into its equivalent positive value.

For test_debug_2:

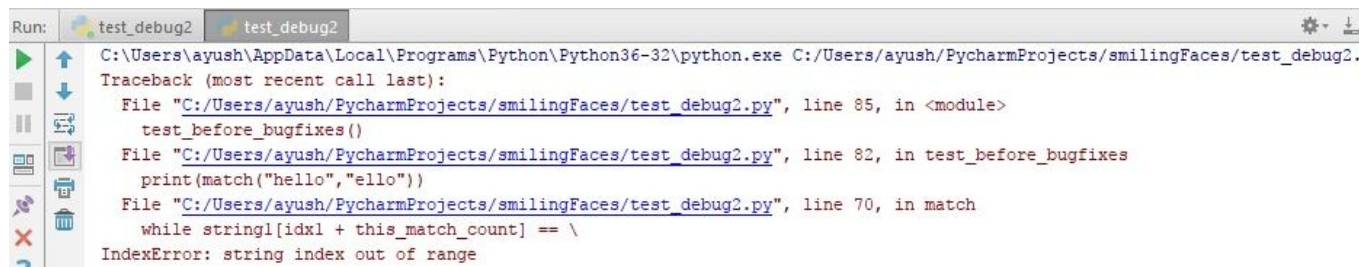
Questions:

(a) Give 2 examples of inputs for which the provided code gives a correct answer despite the fact that it is flawed.

- The provided code didn't work with test function calls

Tried with the following test functions:

- `match("hello", "ello")`
- `match("GaBeN", "ABEn")`

A screenshot of a Python IDE's Run window showing a traceback error. The window has two tabs, both labeled 'test_debug2'. The traceback text is as follows:

```
Run: test_debug2 test_debug2
C:\Users\ayush\AppData\Local\Programs\Python\Python36-32\python.exe C:/Users/ayush/PycharmProjects/smilingFaces/test_debug2.p
Traceback (most recent call last):
  File "C:/Users/ayush/PycharmProjects/smilingFaces/test_debug2.py", line 85, in <module>
    test_before_bugfixes()
  File "C:/Users/ayush/PycharmProjects/smilingFaces/test_debug2.py", line 82, in test_before_bugfixes
    print(match("hello", "ello"))
  File "C:/Users/ayush/PycharmProjects/smilingFaces/test_debug2.py", line 70, in match
    while stringl[idxl + this_match_count] == \
IndexError: string index out of range
```

(b) For each example, explain why the faulty code produced the correct answer, despite the flaw(s).

Solution: Did not produce any output.

(c) Describe the bug(s) present in the code, and for each bug, indicate what the fix is.

Solution:

- The program didn't check for the case of the words that is, if the first word is in capital letters and the second word is in camelcase. **Fix:** Converted both the strings to lowercase so that there is no conflict for the case of the letters of the strings.

- The program contained the while loop where it checks for the equal letters in the string and counts the matches by the increment of this_match_count by 1 every time it found a match. But it doesn't check if the sum of index and this_match_count is greater than string or not, so when the while loop gets executed, the index+this_match_count value goes greater than the length of the string. So, it throws the error: "Index value out of range" **Fix:** Added a condition where it checks if index+this_match_count value goes greater than length of the string, it breaks the while loop.
- PS- Assumed that the length of first string is always greater than length of second string.