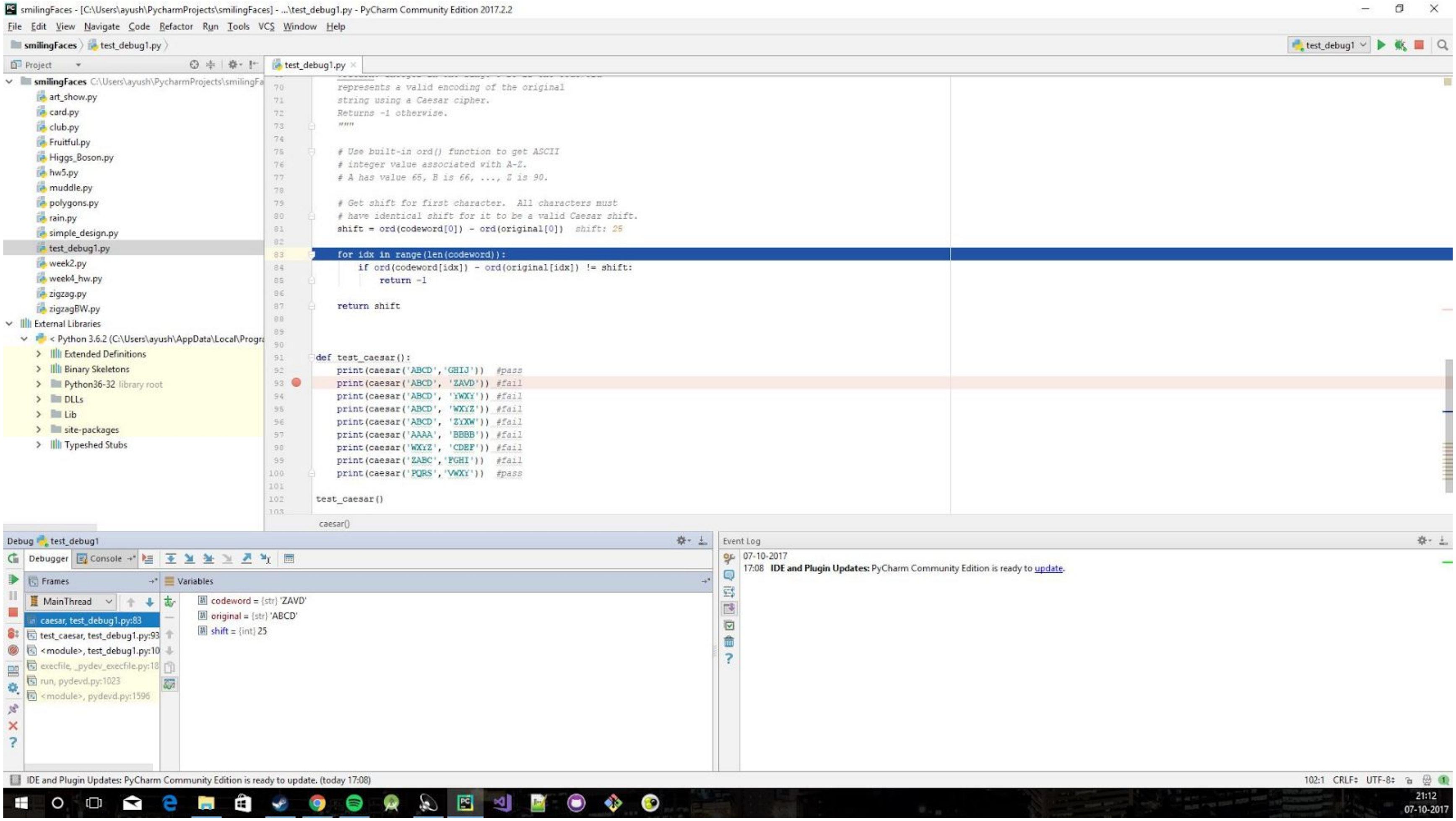
Debug_1



smilingFaces - [C:\Users\ayush\PycharmProjects\smilingFaces] - ...\test_debug1.py - PyCharm Community Edition 2017.2.2

```python
    represents a valid encoding of the original
    string using a Caesar cipher.
    Returns -1 otherwise.
    """

    # Use built-in ord() function to get ASCII
    # integer value associated with A-Z.
    # A has value 65, B is 66, ..., Z is 90.

    # Get shift for first character.  All characters must
    # have identical shift for it to be a valid Caesar shift.
    shift = ord(codeword[0]) - ord(original[0])  shift: 25

    for idx in range(len(codeword)):
        if ord(codeword[idx]) - ord(original[idx]) != shift:
            return -1

    return shift



def test_caesar():
    print(caesar('ABCD','GHIJ'))  #pass
    print(caesar('ABCD', 'ZAVD')) #fail
    print(caesar('ABCD', 'YWXY')) #fail
    print(caesar('ABCD', 'WXYZ')) #fail
    print(caesar('ABCD', 'ZYXW')) #fail
    print(caesar('AAAA', 'BBBB')) #fail
    print(caesar('WXYZ', 'CDEF')) #fail
    print(caesar('ZABC','FGHI'))  #fail
    print(caesar('PQRS','VWXY'))  #pass

test_caesar()
```

caesar()

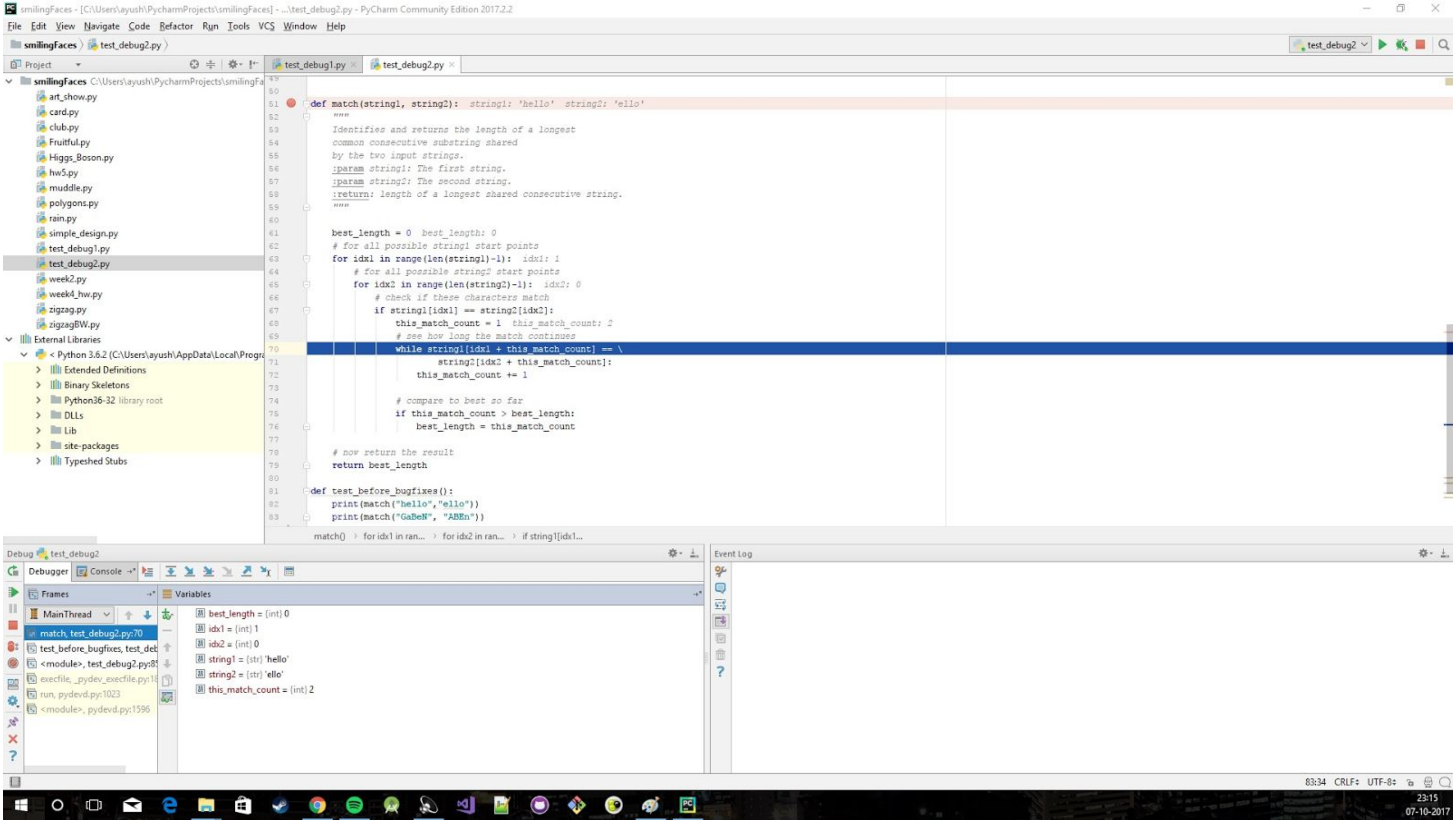
```
caesar, test_debug1.py:83
test_caesar, test_debug1.py:93
<module>, test_debug1.py:10
execfile, _pydev_execfile.py:18
run, pydevd.py:1023
<module>, pydevd.py:1596
```

```
codeword = {str} 'ZAVD'
original = {str} 'ABCD'
shift = {int} 25
```

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

IDE and Plugin Updates: PyCharm Community Edition is ready to update. (today 17:08)

Debug_2

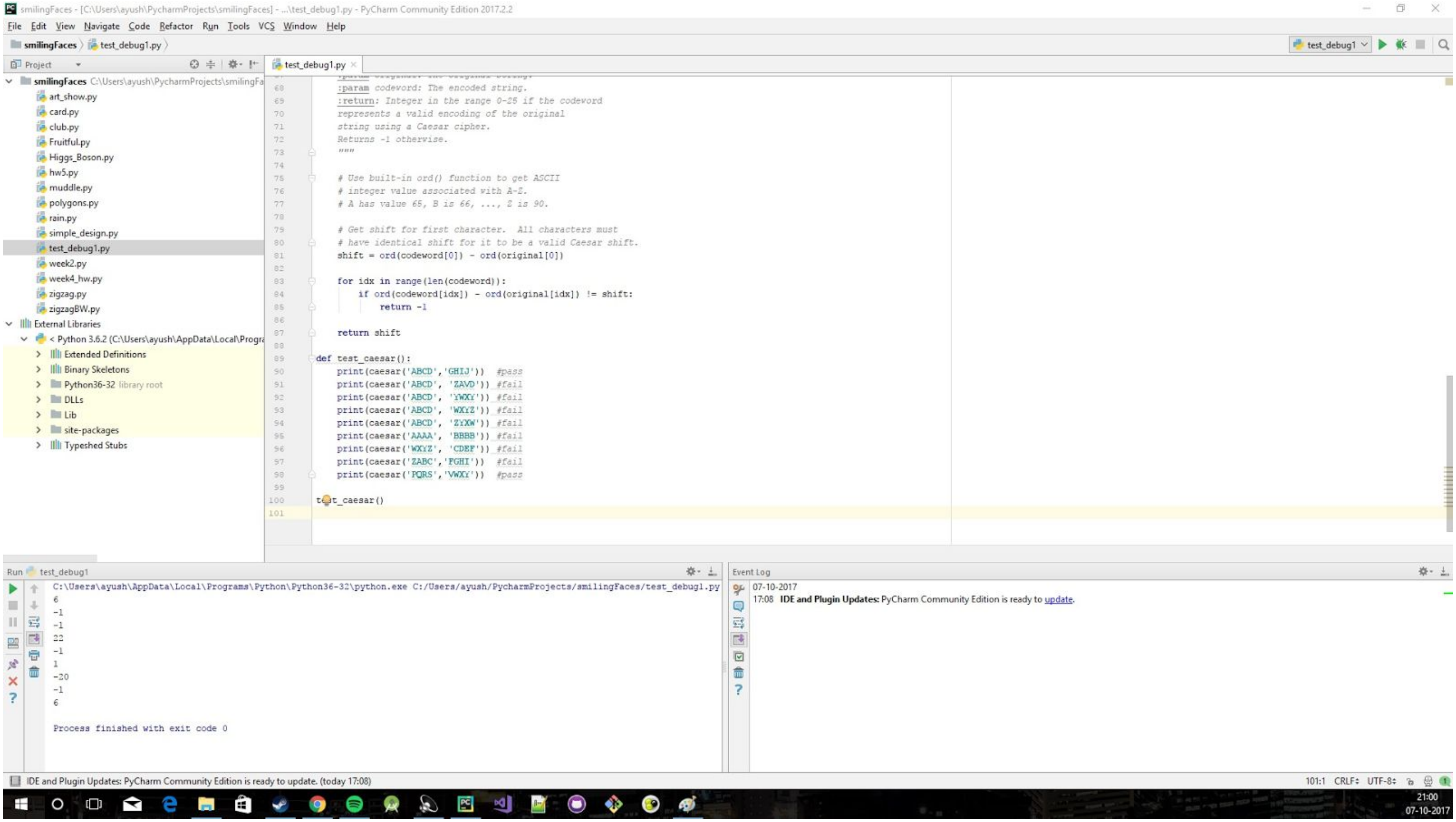


```python
def match(string1, string2):  string1: 'hello'  string2: 'ello'
    """
    Identifies and returns the length of a longest
    common consecutive substring shared
    by the two input strings.
    :param string1: The first string.
    :param string2: The second string.
    :return: length of a longest shared consecutive string.
    """

    best_length = 0  best_length: 0
    # for all possible string1 start points
    for idx1 in range(len(string1)-1):  idx1: 1
        # for all possible string2 start points
        for idx2 in range(len(string2)-1):  idx2: 0
            # check if these characters match
            if string1[idx1] == string2[idx2]:
                this_match_count = 1  this_match_count: 2
                # see how long the match continues
                while string1[idx1 + this_match_count] == \
                        string2[idx2 + this_match_count]:
                    this_match_count += 1

                # compare to best so far
                if this_match_count > best_length:
                    best_length = this_match_count

    # now return the result
    return best_length

def test_before_bugfixes():
    print(match("hello","ello"))
    print(match("GaBeN", "ABEn"))
```

```
match, test_debug2.py:70
test_before_bugfixes, test_deb
<module>, test_debug2.py:85
execfile, _pydev_execfile.py:18
run, pydevd.py:1023
<module>, pydevd.py:1596
```

```
best_length = {int} 0
idx1 = {int} 1
idx2 = {int} 0
string1 = {str} 'hello'
string2 = {str} 'ello'
this_match_count = {int} 2
```

Test_suites1



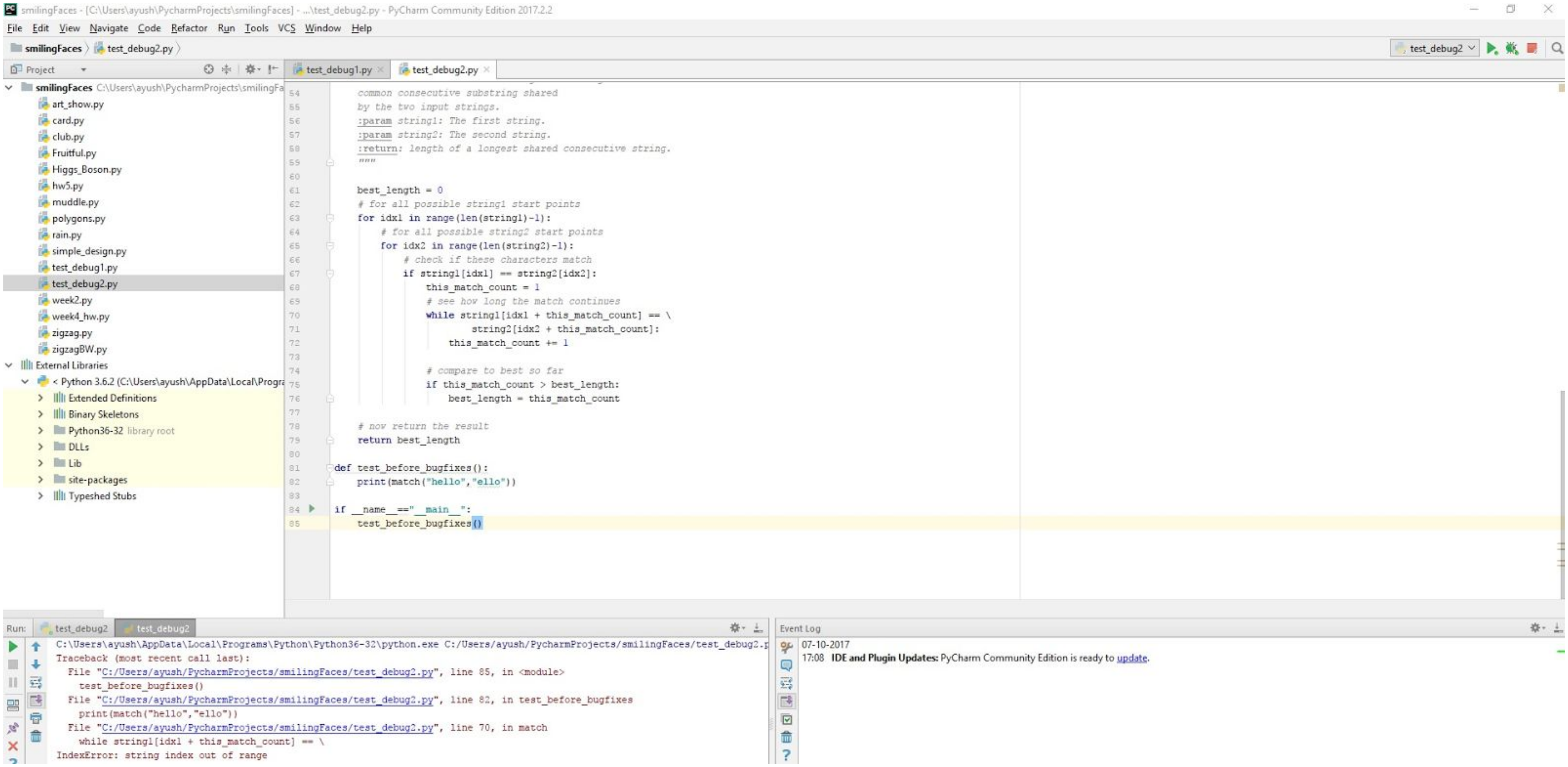
```
        :param original: The original string.
        :param codeword: The encoded string.
        :return: Integer in the range 0-25 if the codeword
        represents a valid encoding of the original
        string using a Caesar cipher.
        Returns -1 otherwise.
        """

        # Use built-in ord() function to get ASCII
        # integer value associated with A-Z.
        # A has value 65, B is 66, ..., Z is 90.

        # Get shift for first character.  All characters must
        # have identical shift for it to be a valid Caesar shift.
        shift = ord(codeword[0]) - ord(original[0])

        for idx in range(len(codeword)):
            if ord(codeword[idx]) - ord(original[idx]) != shift:
                return -1

        return shift

    def test_caesar():
        print(caesar('ABCD','GHIJ'))  #pass
        print(caesar('ABCD', 'ZAVD')) #fail
        print(caesar('ABCD', 'YWXY')) #fail
        print(caesar('ABCD', 'WXYZ')) #fail
        print(caesar('ABCD', 'ZYXW')) #fail
        print(caesar('AAAA', 'BBBB')) #fail
        print(caesar('WXYZ', 'CDEF')) #fail
        print(caesar('ZABC','FGHI'))  #fail
        print(caesar('PQRS','VWXY'))  #pass

    test_caesar()
```

```
C:\Users\ayush\AppData\Local\Programs\Python\Python36-32\python.exe C:/Users/ayush/PycharmProjects/smilingFaces/test_debug1.py
6
-1
-1
22
-1
1
-20
-1
6

Process finished with exit code 0
```

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

IDE and Plugin Updates: PyCharm Community Edition is ready to update. (today 17:08)

Test_suites2



```
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()
```

```
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 string1[idx1 + this_match_count] == \
IndexError: string index out of range
```

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:

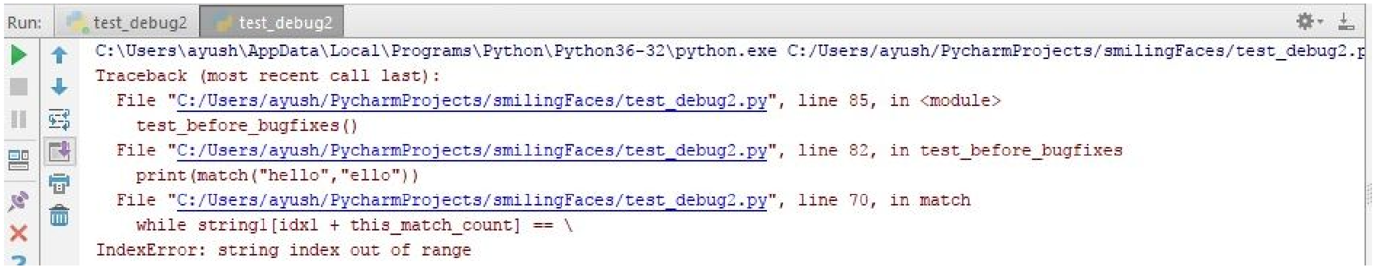
Questions:

(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")`



```
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 string1[idx1 + 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.