Question 1:

Part 1:

Names of the local variables are offsets, pattern_length, text_length at Line 16.

offsets = {list} <class 'list'>: []

__len__ = {int} 0

pattern = {str} 'felis'

pattern_length = {int} 5

text = {str} Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce semper vestibulum nunc ut rhoncus. In ut tellus a velit commodo ullamcorper quis non odio. Mauris bibendum viverra condimentum. Ut porttitor, nisl eget dictum rhoncus, ipsum lacus tristique felis, vitae rutrum nibh ex tempor tortor. Proin condimentum felis id viverra malesuada. Sed efficitur libero eu sagittis interdum. Nullam tempor placerat felis rhoncus gravida. Vivamus porta ornare interdum. Proin eu tempus neque, viverra condimentum nibh. Maecenas a lacus vitae ex aliquet tristique. Etiam et tincidunt lorem. Proin ultrices accumsan euismod. Nullam sed leo erat. Morbi massa neque, convallis non interdum vel, euismod.

text_length = {int} 845

Part 2:

Value of Pattern Length is 5

Part 3:

value of the variable search_index the _rst time line 23 is executed is 4.

Line 23 first time

bmbc = {tuple} <class 'tuple'>: (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

char = {str} 'i'

index = {int} 3

offsets = {list} <class 'list'>: []

pattern = {str} 'felis'

pattern_index = {int} 4

pattern_length = {int} 5

search_index = {int} 4

text = {str} 'Lorem ipsum dolor sit amet, consectetur adipiscing elit.   Fusce semper vestibulum nunc ut rhoncus. In ut tellus a   velit commodo ullamcorper quis non odio. Mauris bibendum viverra condimentum. Ut porttitor, nisl eget di

text_length = {int} 845

Part 4:

value of the variable text_index the second time line 23 is executed is 4.

pattern_index = {int} 4

index = {int} 3

bmbc = {tuple} <class 'tuple'>: (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

char = {str} 'i'

offsets = {list} <class 'list'>: []

pattern = {str} 'felis'

pattern_length = {int} 5

search_index = {int} 9

text = {str} 'Lorem ipsum dolor sit amet, consectetur adipiscing elit.   Fusce semper vestibulum nunc ut rhoncus. In ut tellus a   velit commodo ullamcorper quis non odio. Mauris bibendum viverra condimentum. Ut porttitor, nisl eget di

text_index = {int} 4

text_length = {int} 845

Part 5:

the value and type of the variable offests at line 29 when the variable
text_index has the value 377 is **304 with type list**

bmbc = {tuple} <class 'tuple'>: (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

char = {str} 'i'

index = {int} 3

offsets = {list} <class 'list'>: [304]

0 = {int} 304

__len__ = {int} 1

pattern = {str} 'felis'

pattern_index = {int} -1

pattern_length = {int} 5

search_index = {int} 382

text = {str} 'Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce semper vestibulum nunc ut rhoncus. In ut tellus a velit commodo ullamcorper quis non odio. Mauris bibendum viverra condimentum. Ut porttitor, nisl eget di

text_index = {int} 377

text_length = {int} 845

Question 2:

Help on function test_case_1 in module testing:

test_case_1(self)

The test_case_1 function tests the user defined list.

It calls the merge_sort function which takes the user defined list and return the sorted list using Merge Sort

algorithm.

After running the algorithm, it checks the correctness of the output with the one that user expects.

Help on function test_case_2 in module testing:

test_case_2(self)

The test_case_2 function tests the user defined list.

It calls the merge_sort function which takes the user defined list and return the sorted list using Merge Sort

algorithm.

It first checks if the user defined list and the merge sorted list has the same number of elements or not.

After checking the equality of number of elements, it checks the correctness of the output with the one that

user expects.