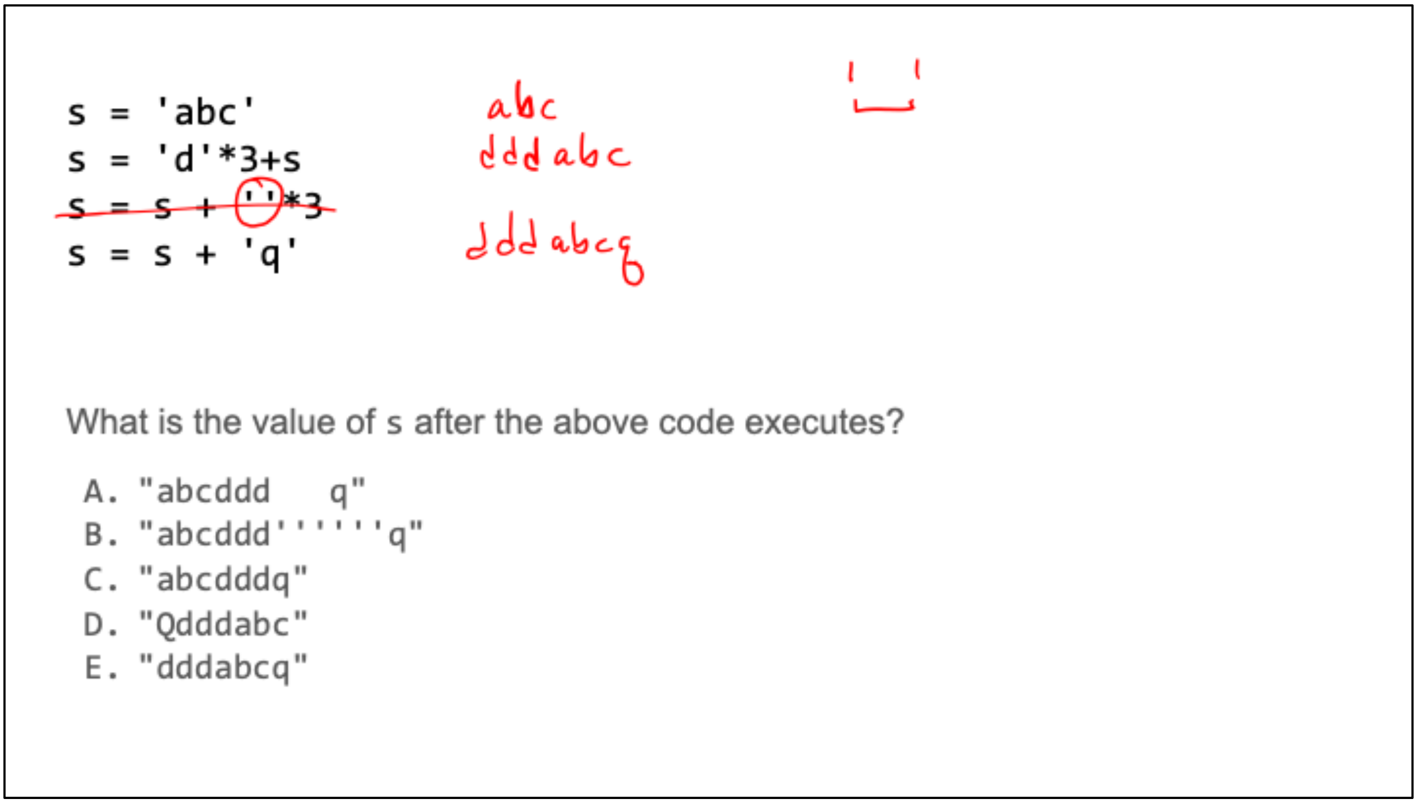


s = 'abc'
s = 'd'*3+s
~~s = s + ''*3~~
s = s + 'q'

abc

dddabc

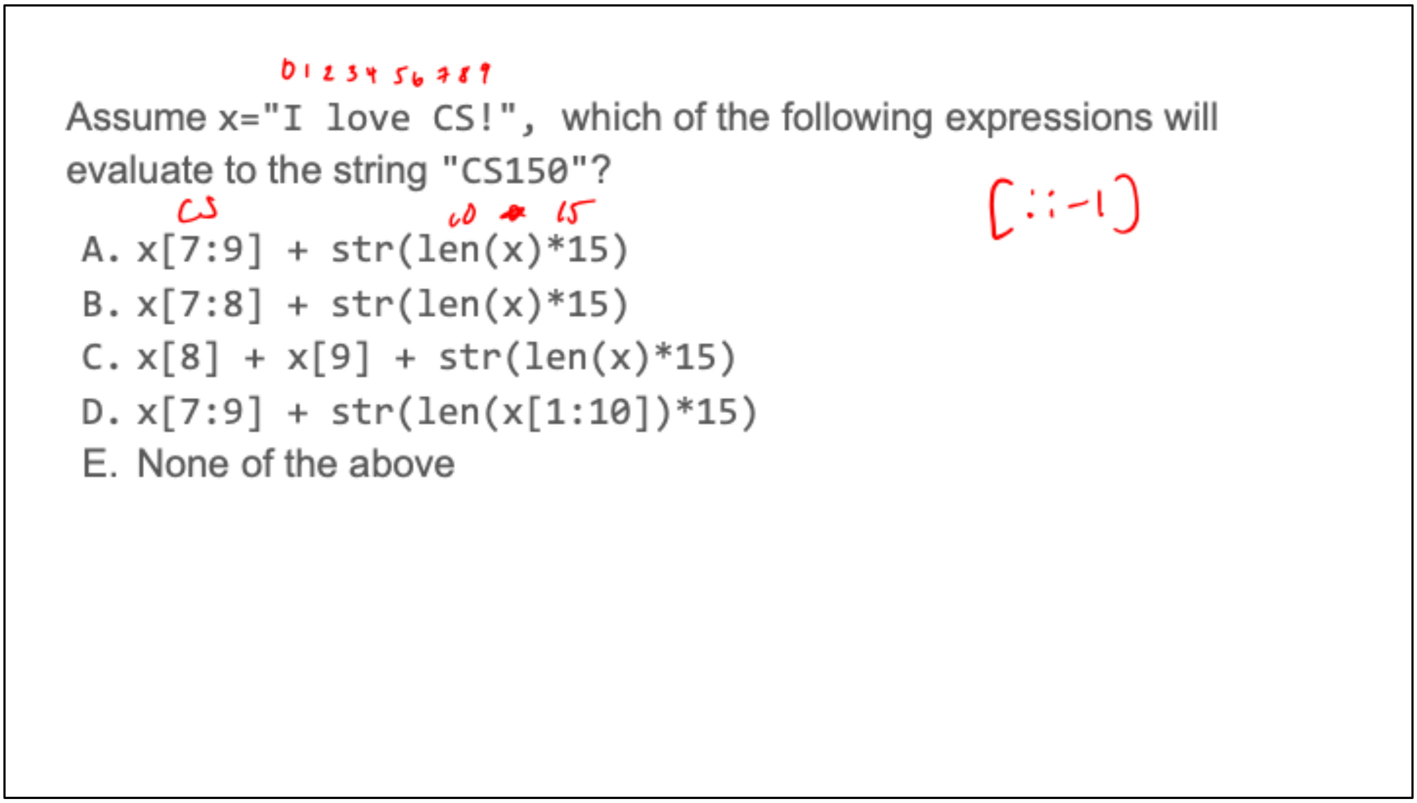
dddabcq

'␣'

What is the value of s after the above code executes?

A. "abcddd   q"
B. "abcddd''''''q"
C. "abcdddq"
D. "Qdddabc"
E. "dddabcq"

Answer: E
Recall that ‘’ is not a single double quote, that is invalid. Instead it is the empty string.
Thus these expressions are the same as:
s = ‘abc’
s = ‘ddd’ + s
s = s + ‘’
s = s + ‘q’
or ‘dddabcq’



0 1 2 3 4 5 6 7 8 9

Assume x="I love CS!", which of the following expressions will evaluate to the string "CS150"?

[::-1]

CS 10 * 15

A. x[7:9] + str(len(x)*15)

B. x[7:8] + str(len(x)*15)

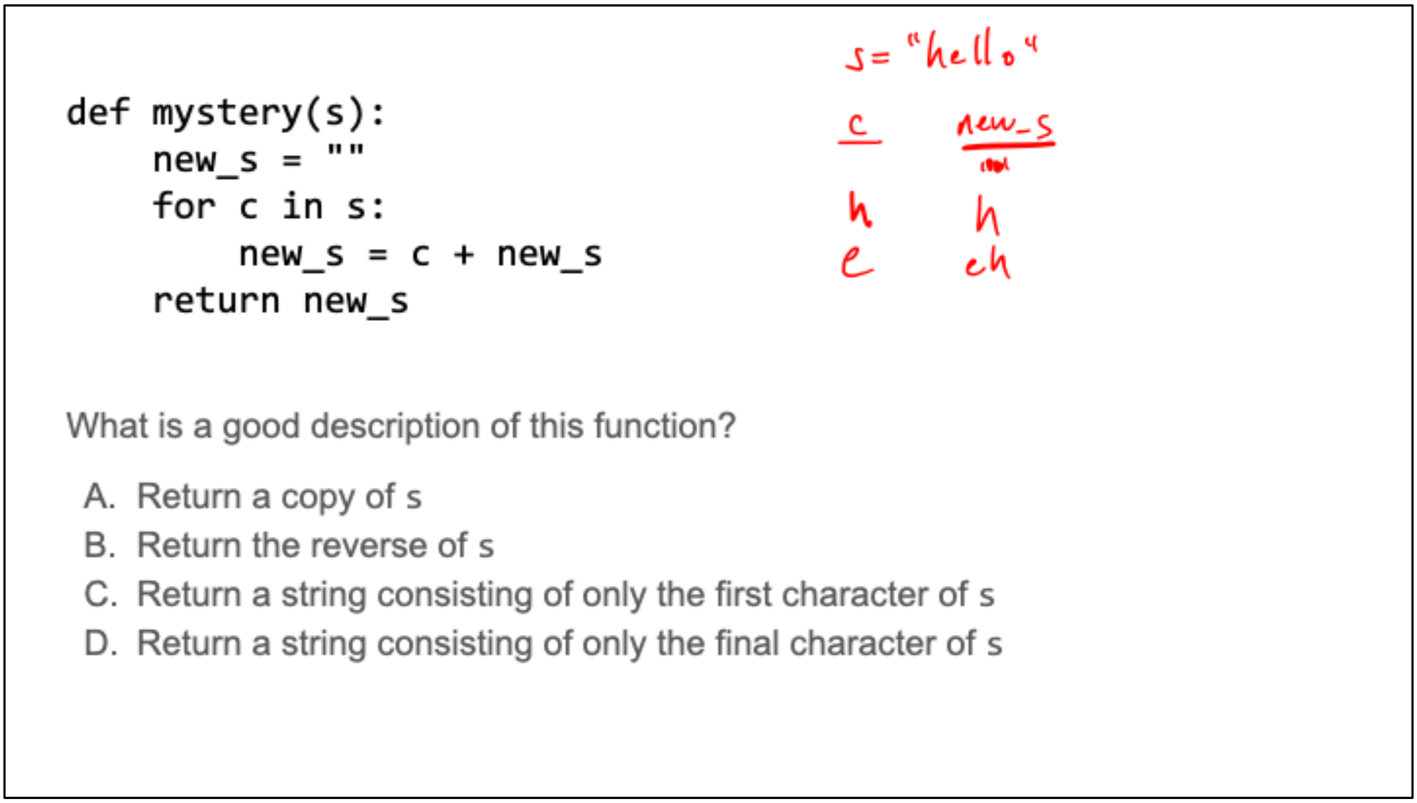
C. x[8] + x[9] + str(len(x)*15)

D. x[7:9] + str(len(x[1:10])*15)

E. None of the above

Answer: A

The first operand to the plus, concatenation, operator, “slices” out “CS”, while the second operand evaluates to `str(10*15)`or “150”. B evaluates to “C150” (recall slice is an exclusive end), C to “S!150” (recall indexing starts at 0), and D is “CS135”, because len(x[1:10]) is 9.



```
def mystery(s):
    new_s = ""
    for c in s:
        new_s = c + new_s
    return new_s
```

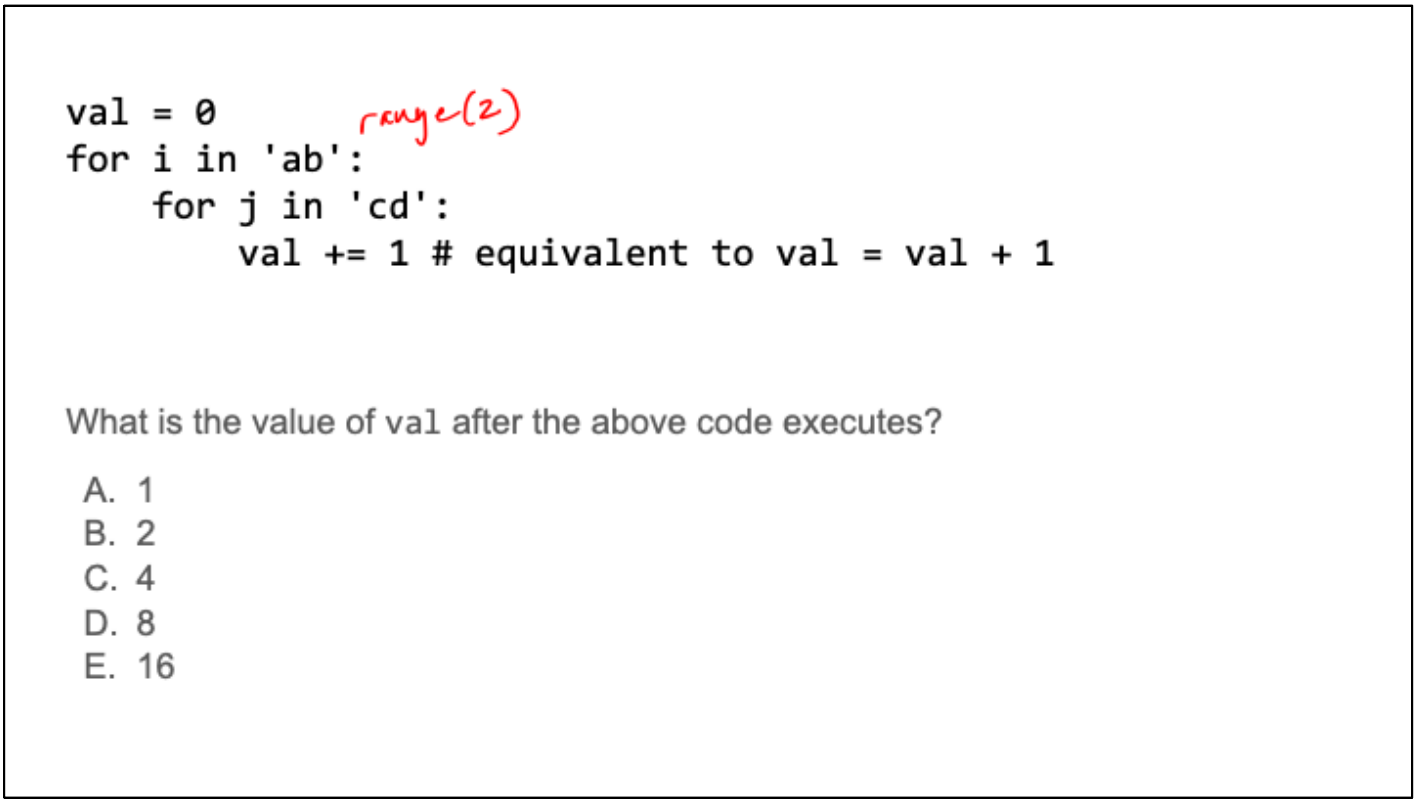
s= "hello"

c	new_s
	""
h	h
e	eh

What is a good description of this function?

A. Return a copy of s
B. Return the reverse of s
C. Return a string consisting of only the first character of s
D. Return a string consisting of only the final character of s

Answer: B
Since we build up the new string, new_s, by prepending, we reverse the string.



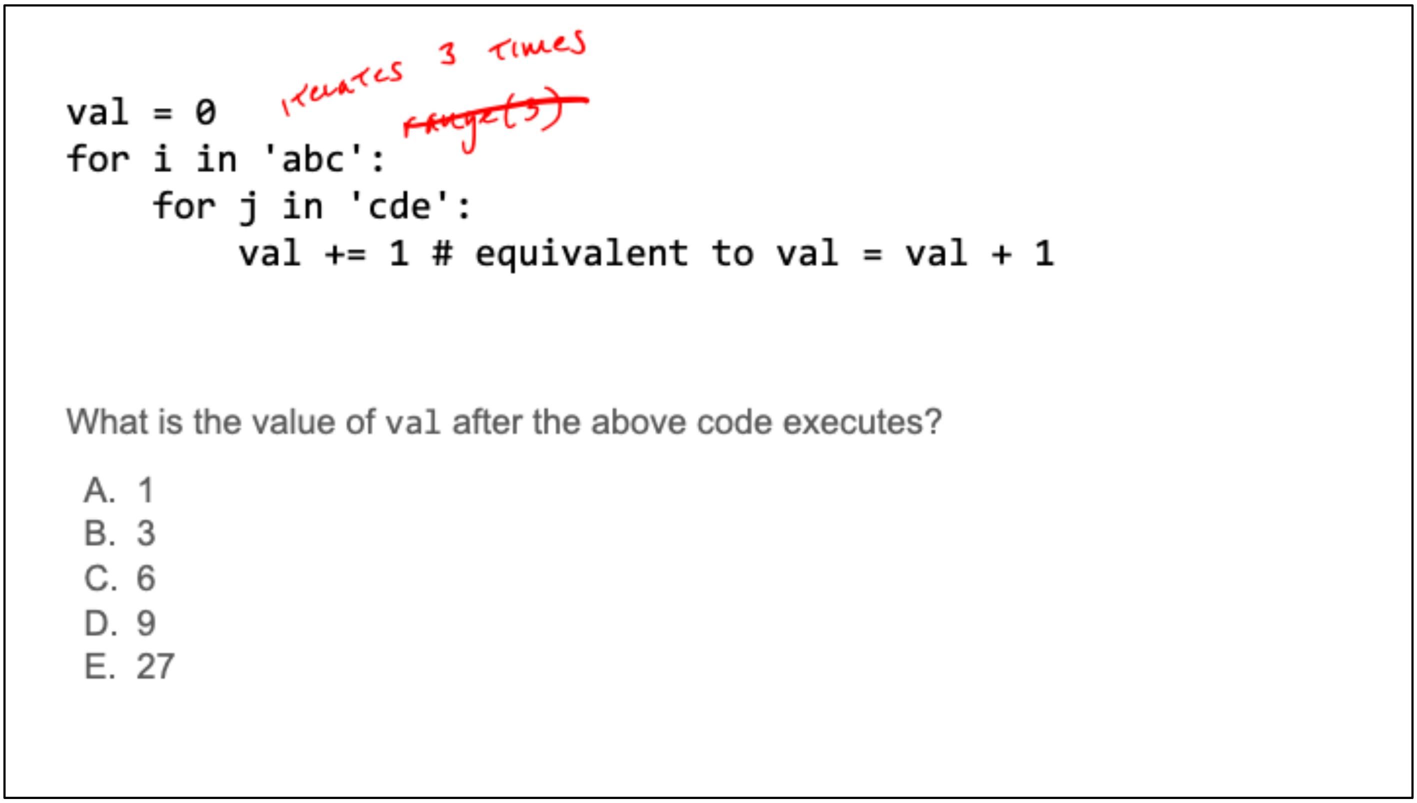
```python
val = 0
for i in 'ab':
    for j in 'cd':
        val += 1 # equivalent to val = val + 1
```

range(2)

What is the value of val after the above code executes?

A. 1
B. 2
C. 4
D. 8
E. 16

Answer: C
There are 2 characters, so each loop will have 2 iterations, and thus we will add 1 to val 4 times (since loops are nested).



iterates 3 times

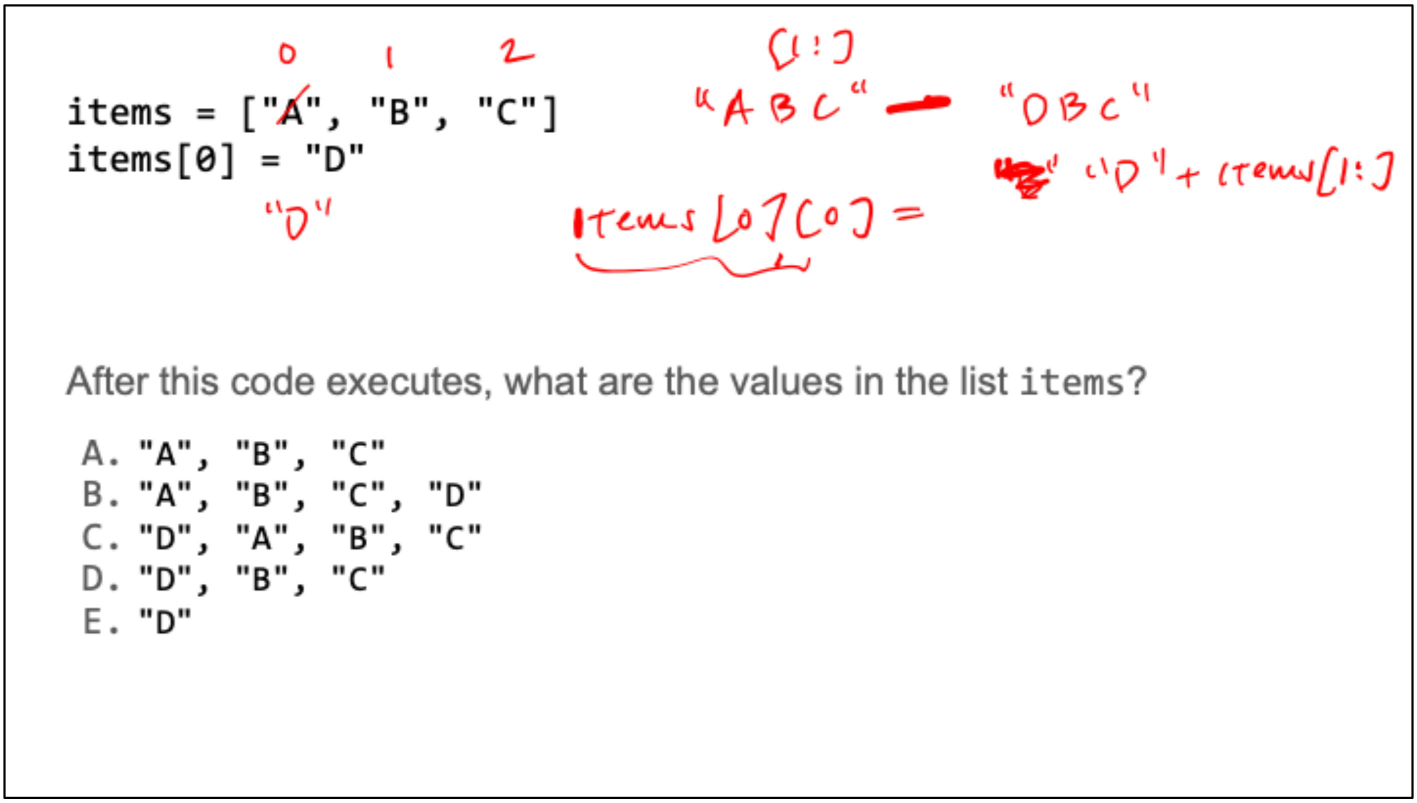
~~range(3)~~

```
val = 0
for i in 'abc':
    for j in 'cde':
        val += 1 # equivalent to val = val + 1
```

What is the value of `val` after the above code executes?

A. 1
B. 3
C. 6
D. 9
E. 27

Answer: D
There are 3 characters, so each loop will have 3 iterations, and thus we will add 1 to val 9 times (since loops are nested).



```
items = ["A", "B", "C"]
items[0] = "D"
```

After this code executes, what are the values in the list `items`?

A. "A", "B", "C"
B. "A", "B", "C", "D"
C. "D", "A", "B", "C"
D. "D", "B", "C"
E. "D"

Answer: D
The second statement reassigns the value at index 0. It does not change the length of the list. Note that you can’t perform the same operation on strings. Because strings are immutable, strings do not support item assignment (i.e., assigning to specific indices/characters).