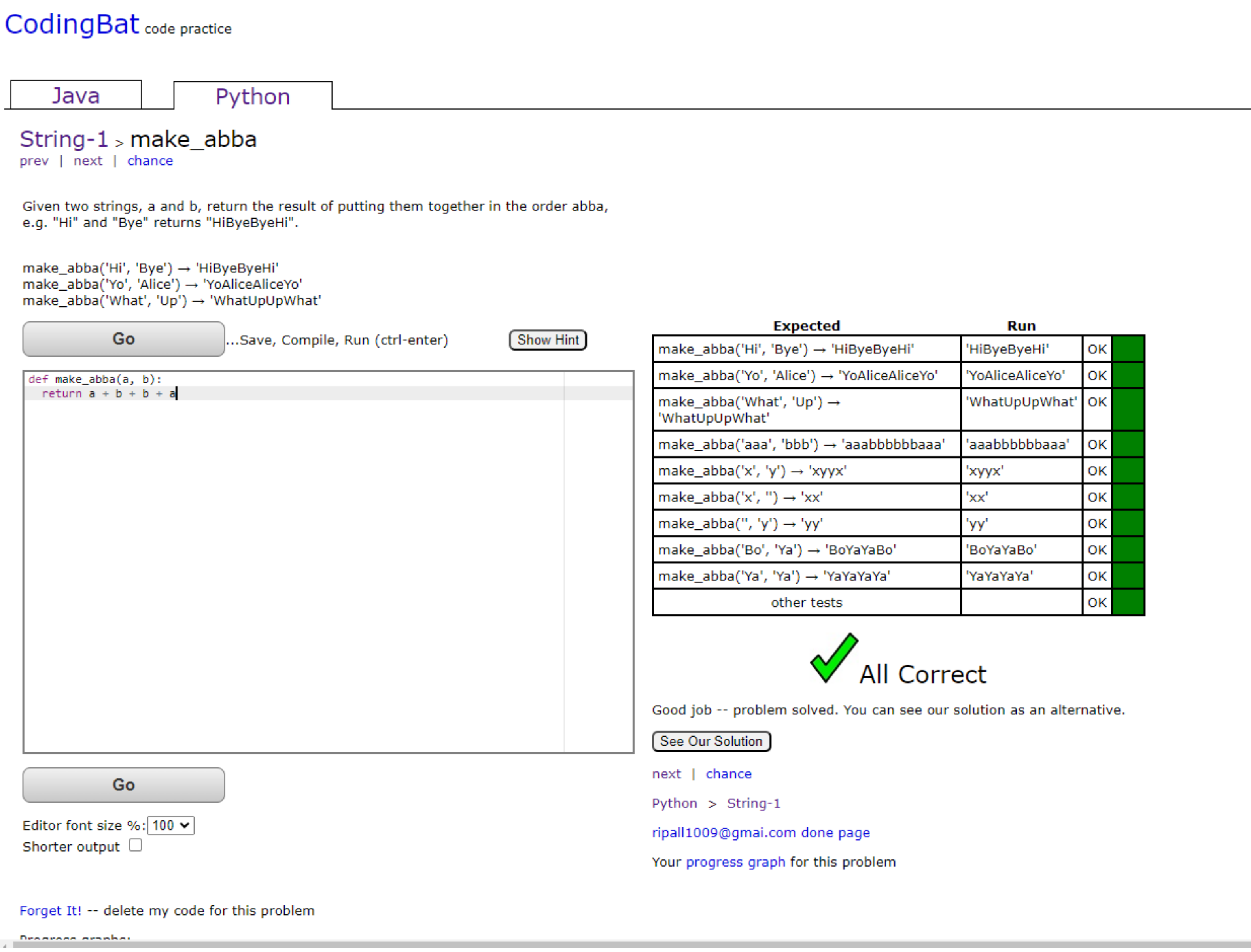


CodingBat code practice

String-1 > make_abba

Given two strings, a and b, return the result of putting them together in the order abba, e.g. "Hi" and "Bye" returns "HiByeByeHi".

make_abba('Hi', 'Bye') → 'HiByeByeHi'
make_abba('Yo', 'Alice') → 'YoAliceAliceYo'
make_abba('What', 'Up') → 'WhatUpUpWhat'

```
def make_abba(a, b):
  return a + b + b + a
```

Editor font size %: 100
Shorter output ☐

Expected	Run		
make_abba('Hi', 'Bye') → 'HiByeByeHi'	'HiByeByeHi'	OK	🟩
make_abba('Yo', 'Alice') → 'YoAliceAliceYo'	'YoAliceAliceYo'	OK	🟩
make_abba('What', 'Up') → 'WhatUpUpWhat'	'WhatUpUpWhat'	OK	🟩
make_abba('aaa', 'bbb') → 'aaabbbbbbaaa'	'aaabbbbbbaaa'	OK	🟩
make_abba('x', 'y') → 'xyyx'	'xyyx'	OK	🟩
make_abba('x', '') → 'xx'	'xx'	OK	🟩
make_abba('', 'y') → 'yy'	'yy'	OK	🟩
make_abba('Bo', 'Ya') → 'BoYaYaBo'	'BoYaYaBo'	OK	🟩
make_abba('Ya', 'Ya') → 'YaYaYaYa'	'YaYaYaYa'	OK	🟩
other tests		OK	🟩

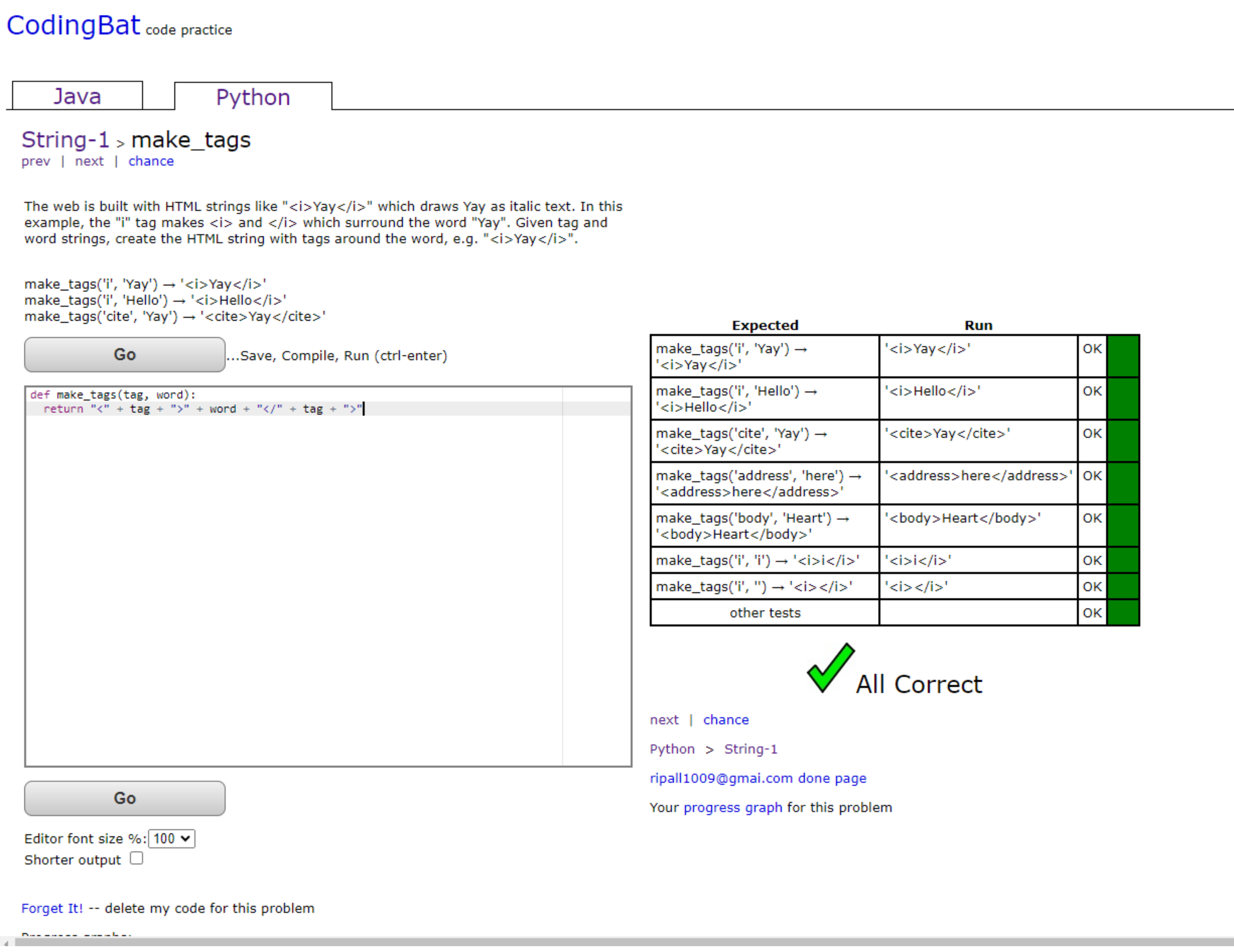
✓ All Correct

Good job -- problem solved. You can see our solution as an alternative.

ripall1009@gmai.com done page

Your progress graph for this problem

Forget It! -- delete my code for this problem



CodingBat code practice

String-1 > make_tags

The web is built with HTML strings like "<i>Yay</i>" which draws Yay as italic text. In this example, the "i" tag makes <i> and </i> which surround the word "Yay". Given tag and word strings, create the HTML string with tags around the word, e.g. "<i>Yay</i>".

make_tags('i', 'Yay') → '<i>Yay</i>'
make_tags('i', 'Hello') → '<i>Hello</i>'
make_tags('cite', 'Yay') → '<cite>Yay</cite>'

...Save, Compile, Run (ctrl-enter)

```python
def make_tags(tag, word):
  return "<" + tag + ">" + word + "</" + tag + ">"
```

Editor font size %: 100
Shorter output ☐

Forget It! -- delete my code for this problem

Expected	Run		
make_tags('i', 'Yay') → '<i>Yay</i>'	'<i>Yay</i>'	OK	🟩
make_tags('i', 'Hello') → '<i>Hello</i>'	'<i>Hello</i>'	OK	🟩
make_tags('cite', 'Yay') → '<cite>Yay</cite>'	'<cite>Yay</cite>'	OK	🟩
make_tags('address', 'here') → '<address>here</address>'	'<address>here</address>'	OK	🟩
make_tags('body', 'Heart') → '<body>Heart</body>'	'<body>Heart</body>'	OK	🟩
make_tags('i', 'i') → '<i>i</i>'	'<i>i</i>'	OK	🟩
make_tags('i', '') → '<i></i>'	'<i></i>'	OK	🟩
other tests		OK	🟩

✓ All Correct

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

CodingBat code practice

Java | Python

String-1 > make_out_word

prev | next | chance

Given an "out" string length 4, such as "<<>>", and a word, return a new string where the word is in the middle of the out string, e.g. "<<word>>".

make_out_word('<<>>', 'Yay') → '<<Yay>>'
make_out_word('<<>>', 'WooHoo') → '<<WooHoo>>'
make_out_word('[[]]', 'word') → '[[word]]'

Go ...Save, Compile, Run (ctrl-enter)

```
def make_out_word(out, word):
  return out[:2] + word + out[2:]
```

Expected	Run		
make_out_word('<<>>', 'Yay') → '<<Yay>>'	'<<Yay>>'	OK	🟩
make_out_word('<<>>', 'WooHoo') → '<<WooHoo>>'	'<<WooHoo>>'	OK	🟩
make_out_word('[[]]', 'word') → '[[word]]'	'[[word]]'	OK	🟩
make_out_word('HHoo', 'Hello') → 'HHHellooo'	'HHHellooo'	OK	🟩
make_out_word('abyz', 'YAY') → 'abYAYyz'	'abYAYyz'	OK	🟩
other tests		OK	🟩

✓ All Correct

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

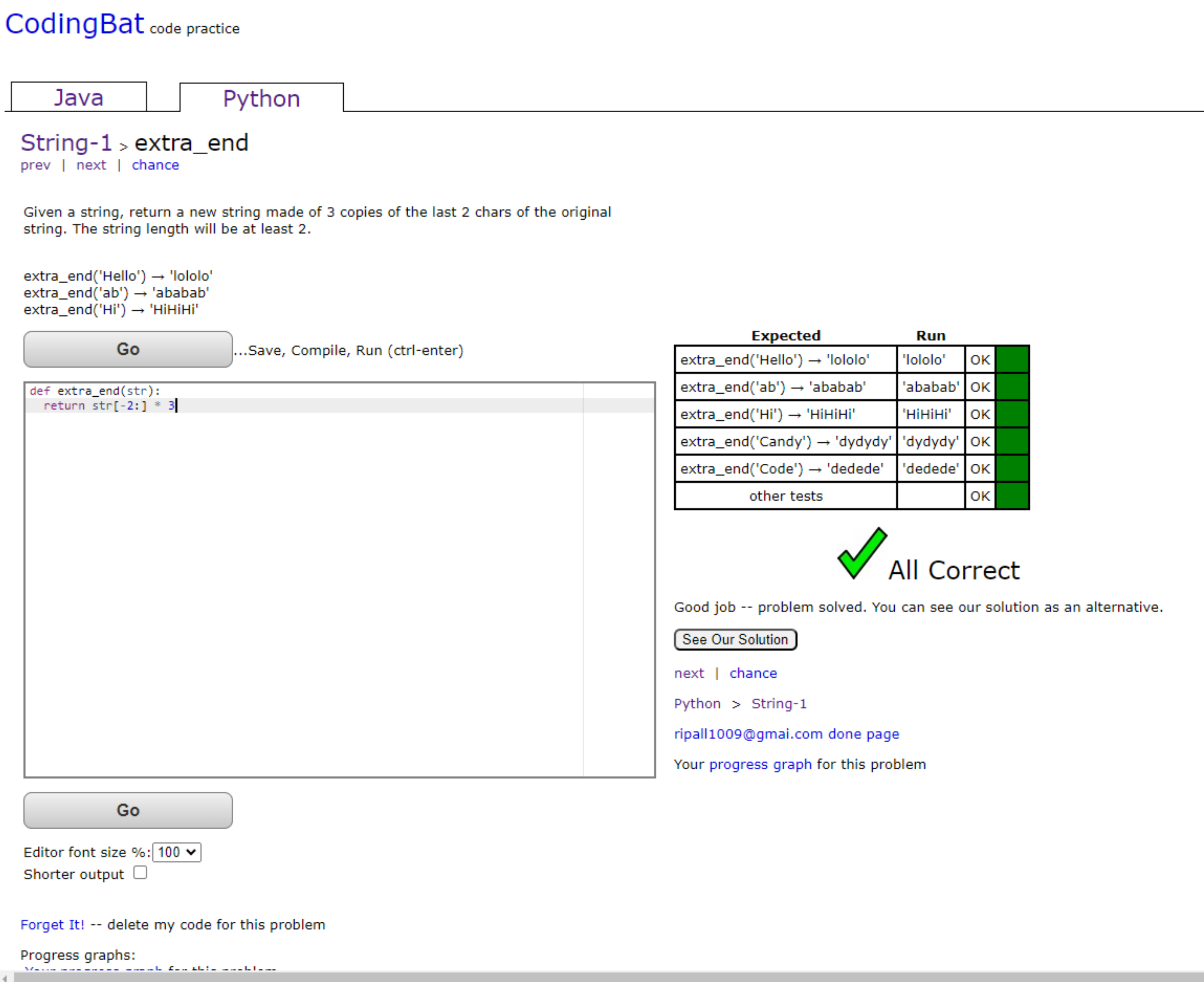
Go

Editor font size %: 100

Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:



CodingBat code practice

String-1 > extra_end

Given a string, return a new string made of 3 copies of the last 2 chars of the original string. The string length will be at least 2.

extra_end('Hello') → 'lololo'
extra_end('ab') → 'ababab'
extra_end('Hi') → 'HiHiHi'

```
def extra_end(str):
  return str[-2:] * 3
```

Editor font size %: 100
Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:
Your progress graph for this problem

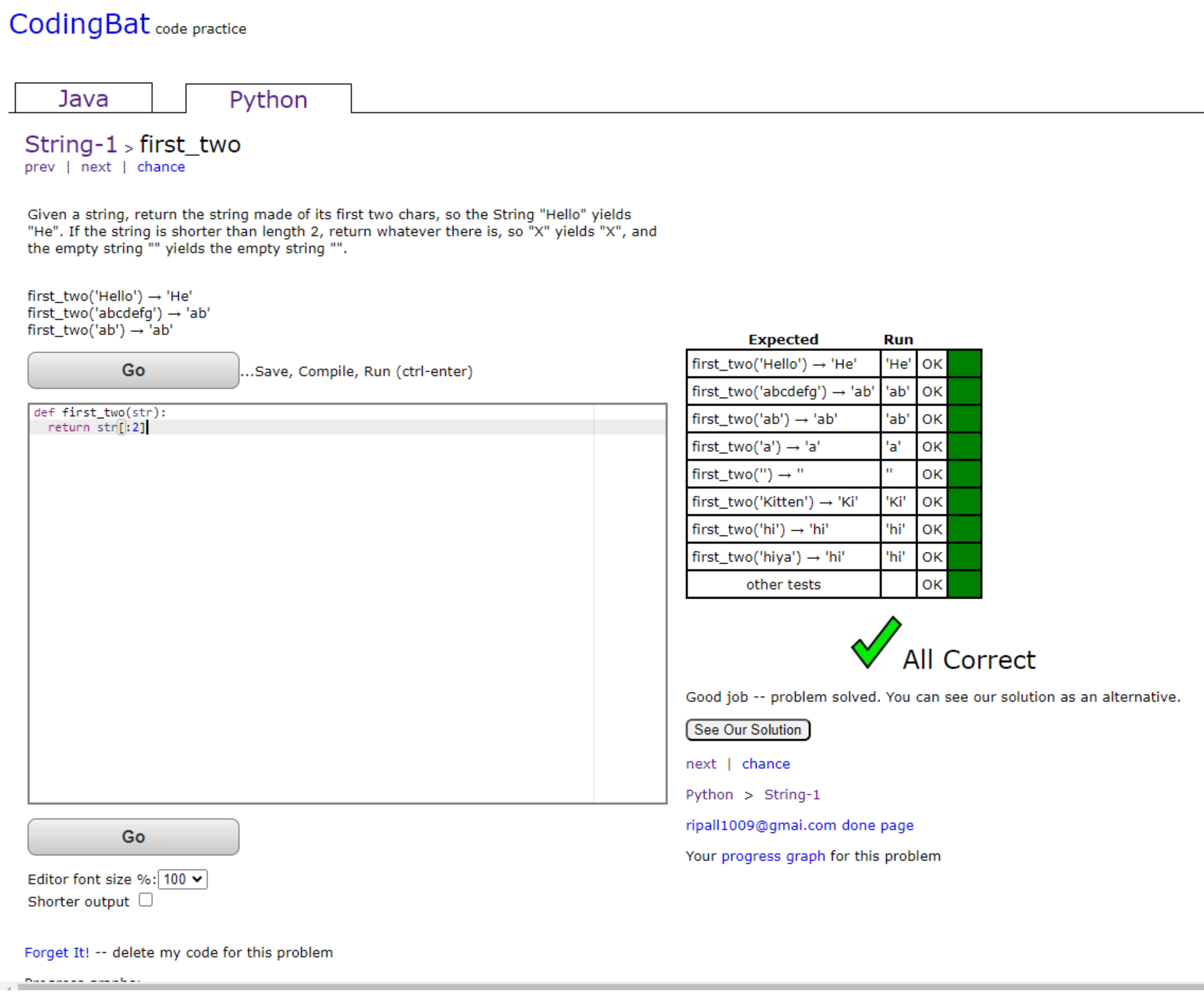
Expected	Run		
extra_end('Hello') → 'lololo'	'lololo'	OK	🟩
extra_end('ab') → 'ababab'	'ababab'	OK	🟩
extra_end('Hi') → 'HiHiHi'	'HiHiHi'	OK	🟩
extra_end('Candy') → 'dydydy'	'dydydy'	OK	🟩
extra_end('Code') → 'dedede'	'dedede'	OK	🟩
other tests		OK	🟩

✓ All Correct

Good job -- problem solved. You can see our solution as an alternative.

ripall1009@gmai.com done page

Your progress graph for this problem



CodingBat code practice

String-1 > first_two

prev | next | chance

Given a string, return the string made of its first two chars, so the String "Hello" yields "He". If the string is shorter than length 2, return whatever there is, so "X" yields "X", and the empty string "" yields the empty string "".

first_two('Hello') → 'He'
first_two('abcdefg') → 'ab'
first_two('ab') → 'ab'

...Save, Compile, Run (ctrl-enter)

```python
def first_two(str):
  return str[:2]
```

Editor font size %: 100

Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:

Expected	Run		
first_two('Hello') → 'He'	'He'	OK	🟩
first_two('abcdefg') → 'ab'	'ab'	OK	🟩
first_two('ab') → 'ab'	'ab'	OK	🟩
first_two('a') → 'a'	'a'	OK	🟩
first_two('') → ''	''	OK	🟩
first_two('Kitten') → 'Ki'	'Ki'	OK	🟩
first_two('hi') → 'hi'	'hi'	OK	🟩
first_two('hiya') → 'hi'	'hi'	OK	🟩
other tests		OK	🟩

✓ All Correct

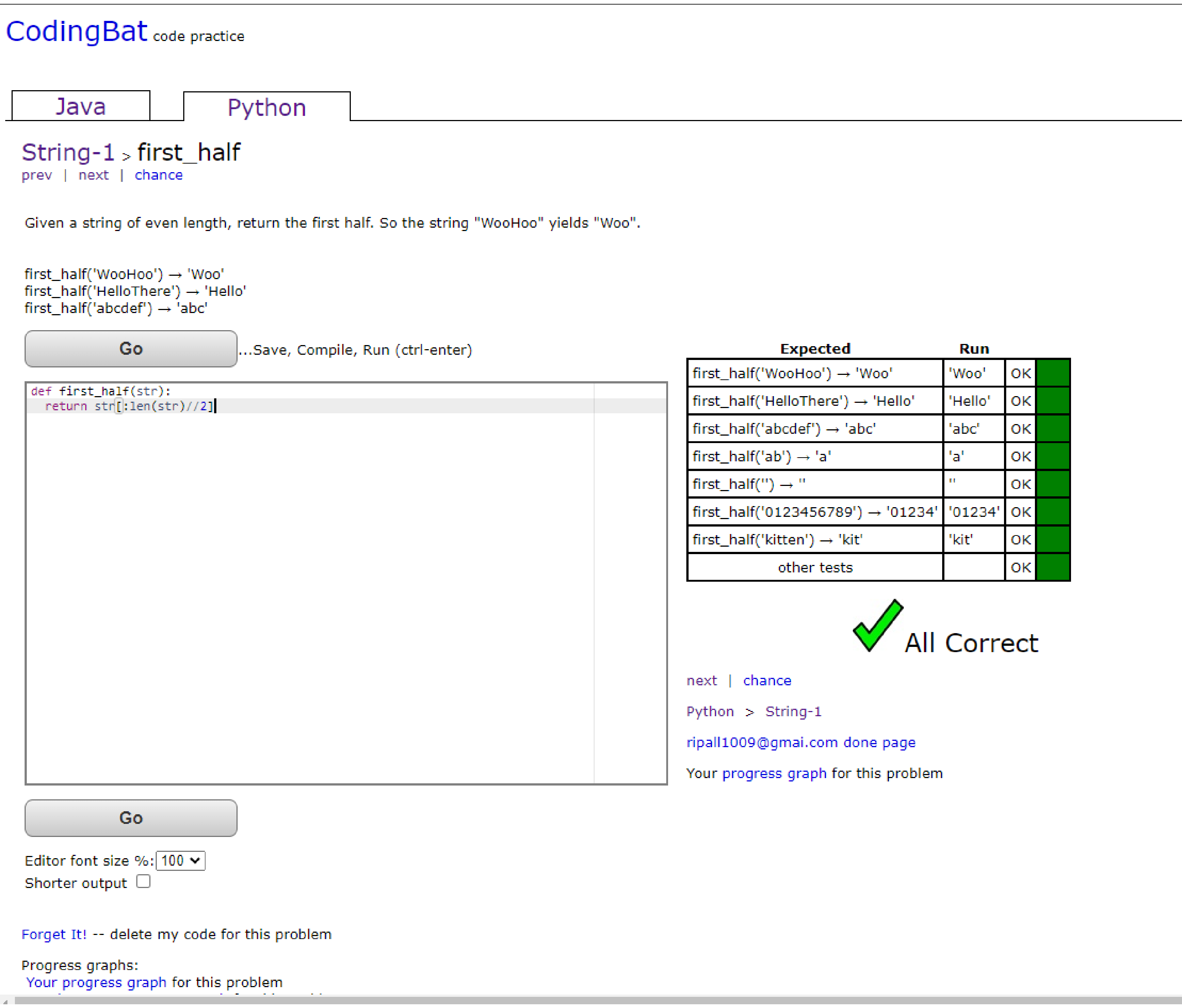
Good job -- problem solved. You can see our solution as an alternative.

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem



CodingBat code practice

Java | Python

String-1 > first_half
prev | next | chance

Given a string of even length, return the first half. So the string "WooHoo" yields "Woo".

first_half('WooHoo') → 'Woo'
first_half('HelloThere') → 'Hello'
first_half('abcdef') → 'abc'

Go ...Save, Compile, Run (ctrl-enter)

```
def first_half(str):
  return str[:len(str)//2]
```

Expected	Run		
first_half('WooHoo') → 'Woo'	'Woo'	OK	🟩
first_half('HelloThere') → 'Hello'	'Hello'	OK	🟩
first_half('abcdef') → 'abc'	'abc'	OK	🟩
first_half('ab') → 'a'	'a'	OK	🟩
first_half('') → ''	''	OK	🟩
first_half('0123456789') → '01234'	'01234'	OK	🟩
first_half('kitten') → 'kit'	'kit'	OK	🟩
other tests		OK	🟩

✓ All Correct

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

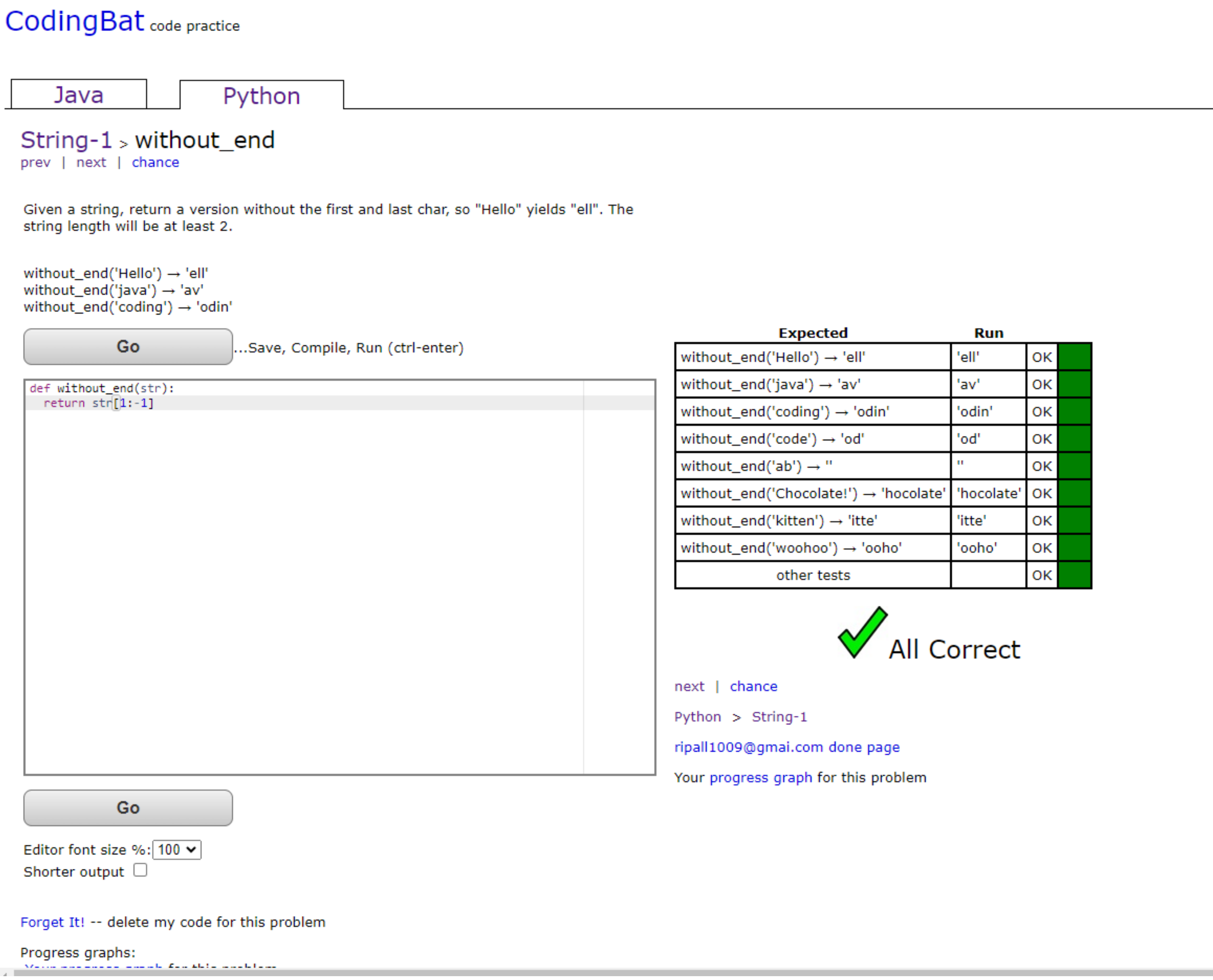
Go

Editor font size %: 100

Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:
Your progress graph for this problem



CodingBat code practice

Java | Python

String-1 > without_end

prev | next | chance

Given a string, return a version without the first and last char, so "Hello" yields "ell". The string length will be at least 2.

without_end('Hello') → 'ell'
without_end('java') → 'av'
without_end('coding') → 'odin'

Go ...Save, Compile, Run (ctrl-enter)

```
def without_end(str):
  return str[1:-1]
```

Expected	Run		
without_end('Hello') → 'ell'	'ell'	OK	🟩
without_end('java') → 'av'	'av'	OK	🟩
without_end('coding') → 'odin'	'odin'	OK	🟩
without_end('code') → 'od'	'od'	OK	🟩
without_end('ab') → ''	''	OK	🟩
without_end('Chocolate!') → 'hocolate'	'hocolate'	OK	🟩
without_end('kitten') → 'itte'	'itte'	OK	🟩
without_end('woohoo') → 'ooho'	'ooho'	OK	🟩
other tests		OK	🟩

✓ All Correct

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

Go

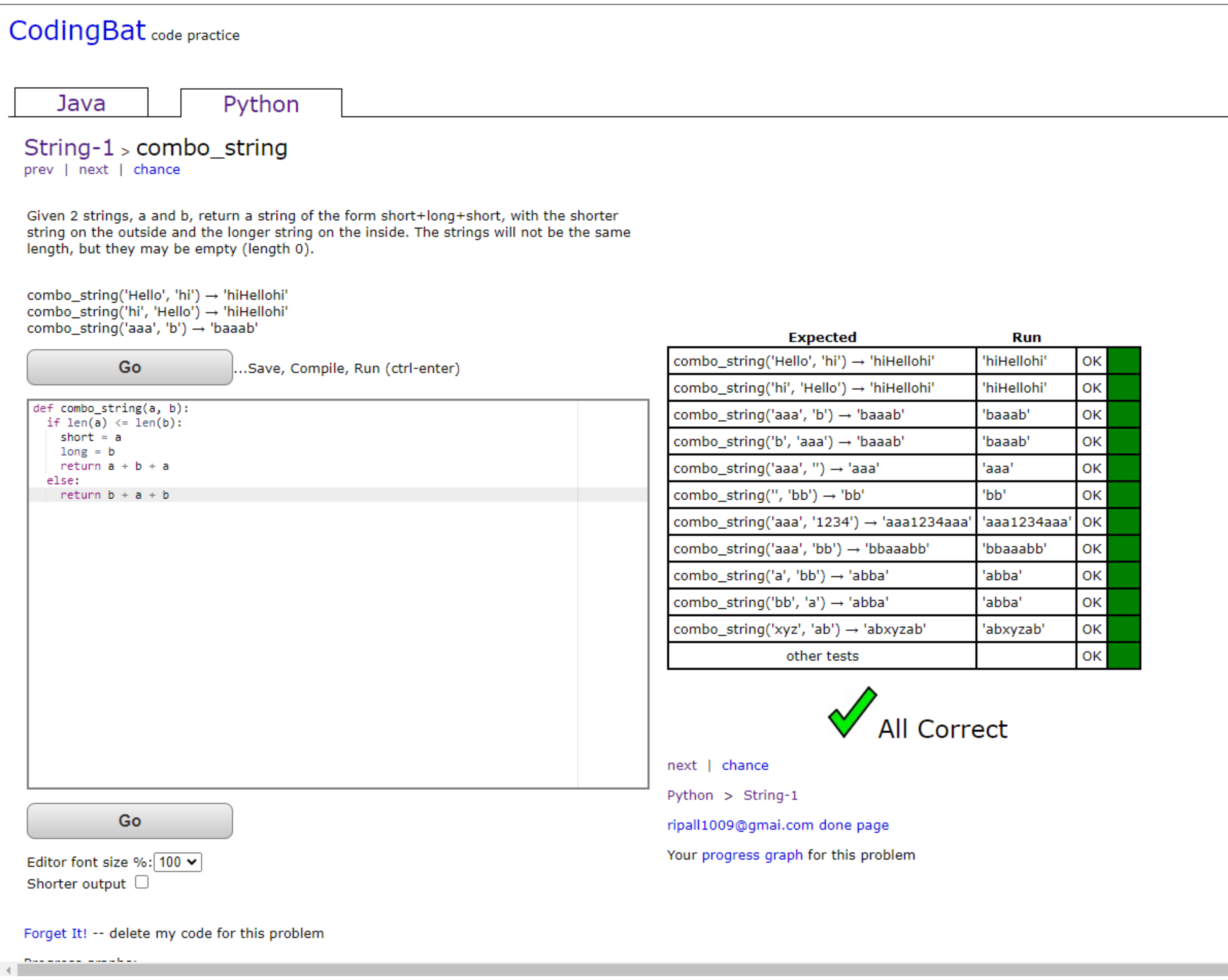
Editor font size %: 100

Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:

Your progress graph for this problem



String-1 > combo_string

prev | next | chance

Given 2 strings, a and b, return a string of the form short+long+short, with the shorter string on the outside and the longer string on the inside. The strings will not be the same length, but they may be empty (length 0).

combo_string('Hello', 'hi') → 'hiHellohi'
combo_string('hi', 'Hello') → 'hiHellohi'
combo_string('aaa', 'b') → 'baaab'

```python
def combo_string(a, b):
  if len(a) <= len(b):
    short = a
    long = b
    return a + b + a
  else:
    return b + a + b
```

Expected	Run		
combo_string('Hello', 'hi') → 'hiHellohi'	'hiHellohi'	OK	🟩
combo_string('hi', 'Hello') → 'hiHellohi'	'hiHellohi'	OK	🟩
combo_string('aaa', 'b') → 'baaab'	'baaab'	OK	🟩
combo_string('b', 'aaa') → 'baaab'	'baaab'	OK	🟩
combo_string('aaa', '') → 'aaa'	'aaa'	OK	🟩
combo_string('', 'bb') → 'bb'	'bb'	OK	🟩
combo_string('aaa', '1234') → 'aaa1234aaa'	'aaa1234aaa'	OK	🟩
combo_string('aaa', 'bb') → 'bbaaabb'	'bbaaabb'	OK	🟩
combo_string('a', 'bb') → 'abba'	'abba'	OK	🟩
combo_string('bb', 'a') → 'abba'	'abba'	OK	🟩
combo_string('xyz', 'ab') → 'abxyzab'	'abxyzab'	OK	🟩
other tests		OK	🟩

✓ All Correct

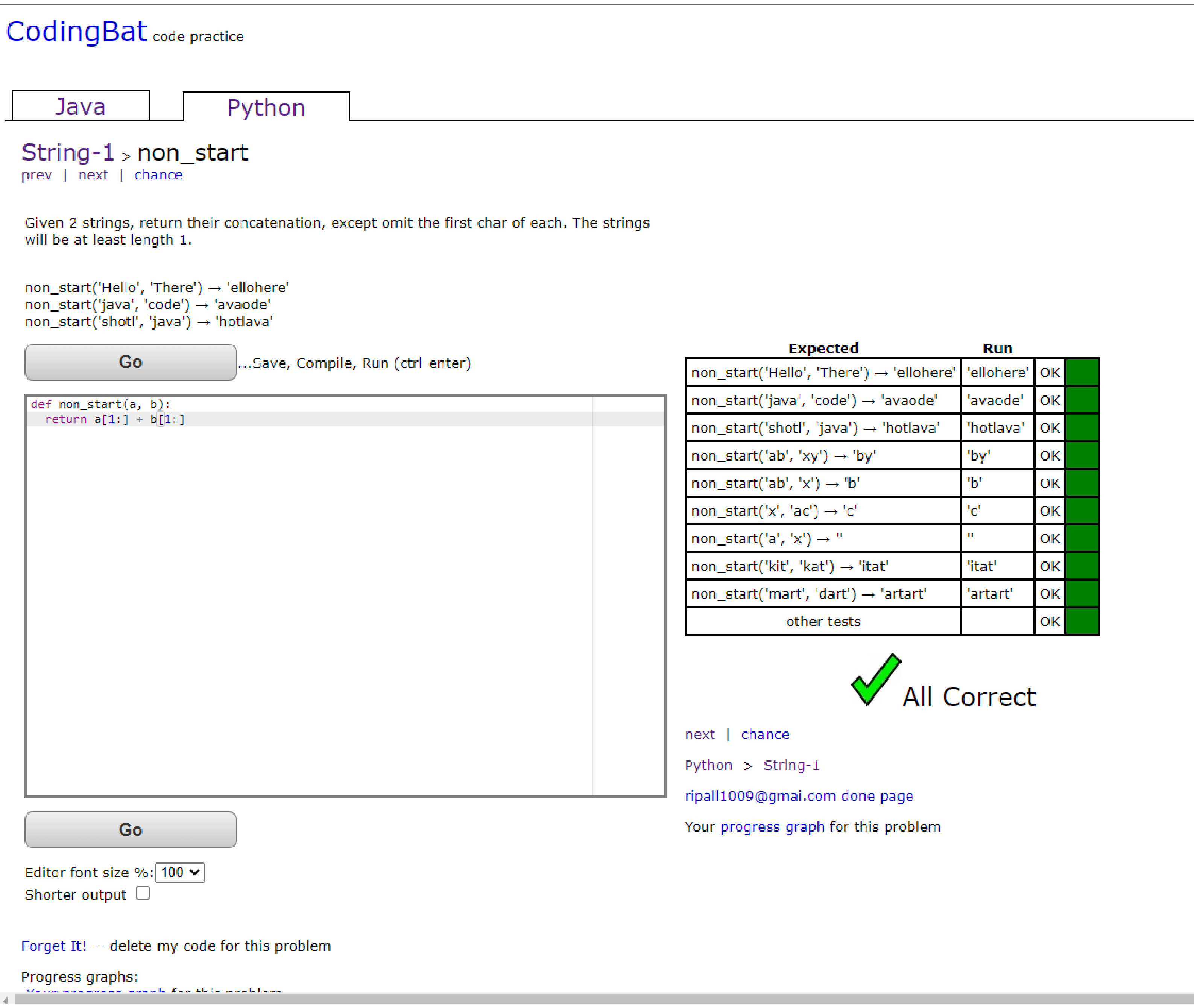
next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

Forget It! -- delete my code for this problem



CodingBat code practice

String-1 > non_start

prev | next | chance

Given 2 strings, return their concatenation, except omit the first char of each. The strings will be at least length 1.

non_start('Hello', 'There') → 'ellohere'
non_start('java', 'code') → 'avaode'
non_start('shotl', 'java') → 'hotlava'

...Save, Compile, Run (ctrl-enter)

```python
def non_start(a, b):
  return a[1:] + b[1:]
```

Expected	Run		
non_start('Hello', 'There') → 'ellohere'	'ellohere'	OK	🟩
non_start('java', 'code') → 'avaode'	'avaode'	OK	🟩
non_start('shotl', 'java') → 'hotlava'	'hotlava'	OK	🟩
non_start('ab', 'xy') → 'by'	'by'	OK	🟩
non_start('ab', 'x') → 'b'	'b'	OK	🟩
non_start('x', 'ac') → 'c'	'c'	OK	🟩
non_start('a', 'x') → ''	''	OK	🟩
non_start('kit', 'kat') → 'itat'	'itat'	OK	🟩
non_start('mart', 'dart') → 'artart'	'artart'	OK	🟩
other tests		OK	🟩

✓ All Correct

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem

Editor font size %: 100
Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:

Your progress graph for this problem

CodingBat code practice

Java | Python

String-1 > left2

prev | next | chance

Given a string, return a "rotated left 2" version where the first 2 chars are moved to the end. The string length will be at least 2.

left2('Hello') → 'lloHe'
left2('java') → 'vaja'
left2('Hi') → 'Hi'

Go ...Save, Compile, Run (ctrl-enter)

```
def left2(str):
  if len(str) >= 2:
      return str[2:] + str[:2]
  else:
      return str
```

Go

Editor font size %: 100
Shorter output ☐

Forget It! -- delete my code for this problem

Progress graphs:
Your progress graph for this problem

Expected	Run		
left2('Hello') → 'lloHe'	'lloHe'	OK	🟩
left2('java') → 'vaja'	'vaja'	OK	🟩
left2('Hi') → 'Hi'	'Hi'	OK	🟩
left2('code') → 'deco'	'deco'	OK	🟩
left2('cat') → 'tca'	'tca'	OK	🟩
left2('12345') → '34512'	'34512'	OK	🟩
left2('Chocolate') → 'ocolateCh'	'ocolateCh'	OK	🟩
left2('bricks') → 'icksbr'	'icksbr'	OK	🟩
other tests		OK	🟩

✓ All Correct

Good job -- problem solved. You can see our solution as an alternative.

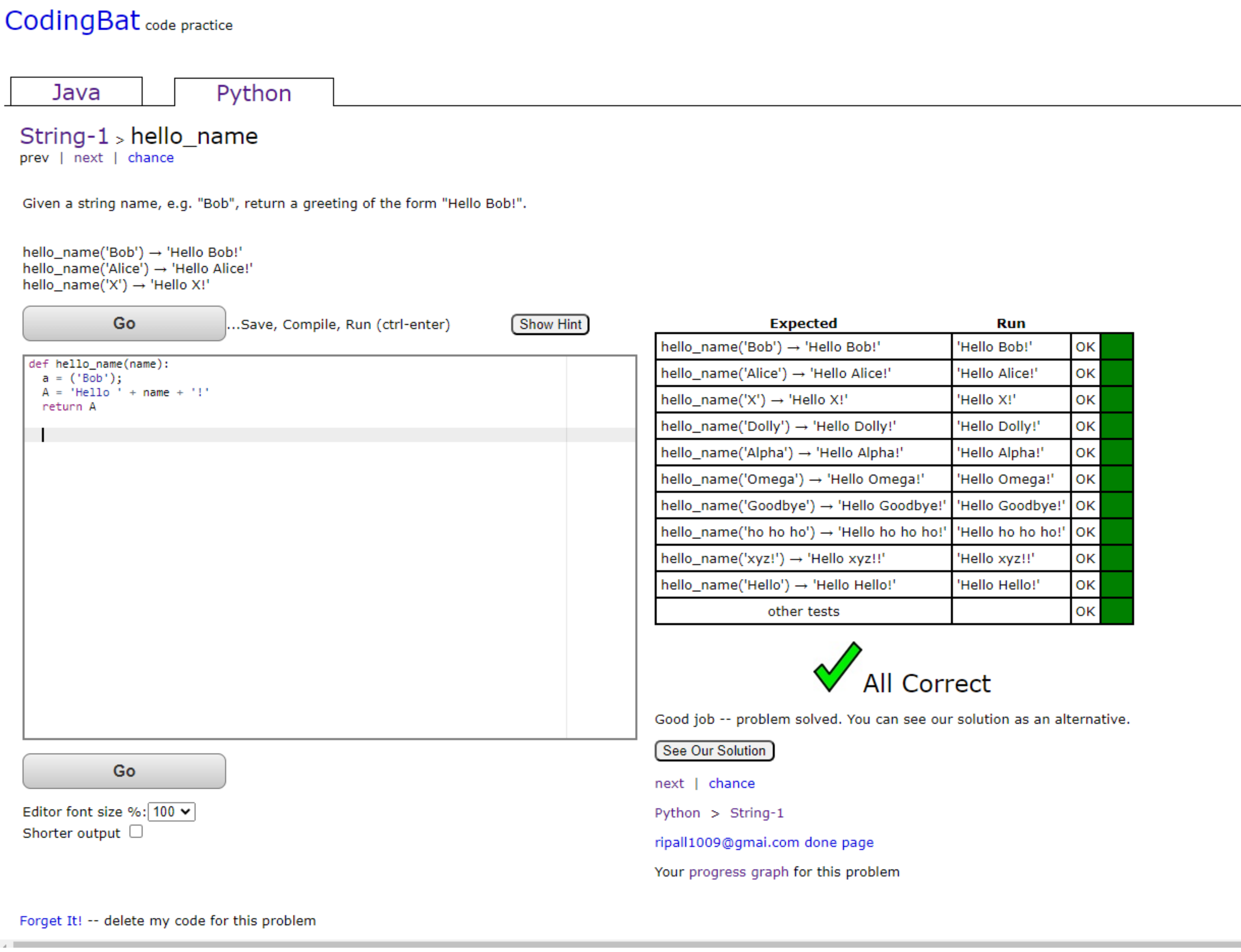
See Our Solution

next | chance

Python > String-1

ripall1009@gmai.com done page

Your progress graph for this problem



String-1 > hello_name

Given a string name, e.g. "Bob", return a greeting of the form "Hello Bob!".

hello_name('Bob') → 'Hello Bob!'
hello_name('Alice') → 'Hello Alice!'
hello_name('X') → 'Hello X!'

```
def hello_name(name):
  a = ('Bob');
  A = 'Hello ' + name + '!'
  return A
```

Expected	Run		
hello_name('Bob') → 'Hello Bob!'	'Hello Bob!'	OK	🟩
hello_name('Alice') → 'Hello Alice!'	'Hello Alice!'	OK	🟩
hello_name('X') → 'Hello X!'	'Hello X!'	OK	🟩
hello_name('Dolly') → 'Hello Dolly!'	'Hello Dolly!'	OK	🟩
hello_name('Alpha') → 'Hello Alpha!'	'Hello Alpha!'	OK	🟩
hello_name('Omega') → 'Hello Omega!'	'Hello Omega!'	OK	🟩
hello_name('Goodbye') → 'Hello Goodbye!'	'Hello Goodbye!'	OK	🟩
hello_name('ho ho ho') → 'Hello ho ho ho!'	'Hello ho ho ho!'	OK	🟩
hello_name('xyz!') → 'Hello xyz!!'	'Hello xyz!!'	OK	🟩
hello_name('Hello') → 'Hello Hello!'	'Hello Hello!'	OK	🟩
other tests		OK	🟩

✓ All Correct

Good job -- problem solved. You can see our solution as an alternative.

ripall1009@gmai.com done page

Your progress graph for this problem

Editor font size %: 100
Shorter output ☐

Forget It! -- delete my code for this problem