Format as Currency

Write a function that takes a string as its only parameter and constructs a new string that is the same as the parameter except:

- A dollar sign has been added to the beginning of it
- The string contains a decimal point with exactly two characters after it. When the number of digits to the right of the decimal points has to be reduced the number is truncated to two decimal places (not rounded)
- Commas are inserted into the digits to the left of the decimal point so that digits are separated into groups of 3 in the usual manner for base 10 numbers

The function takes a string as its parameter instead of an integer so that placeholders such as XXXXXX can be formatted as $XXX,XXX.00 instead of only being able to format numbers.

Examples:

formatAsCurrency("0") should return "$0.00"

formatAsCurrency("1234567.89") should return "$1,234,567.89"

formatAsCurrency("1.23456789") should return "$1.23"

formatAsCurrency("12.34") should return "$12.34"

formatAsCurrency("abcde.fgh") should return "$ab,cde.fg"

Use the following lines shown on the next page to create your program. Some lines may not be required for a correct solution. Lines may be used multiple times.

1: after = left[0 : i]
2: after = left[i : len(left)]
3: before = left[0 : i]
4: before = left[i : len(left)]
5: dec_pos = s.find(".")
6: def formatAsCurrency(s):
7: else:
8: for i in range(len(left) - 3, 0, -3):
9: if dec_pos == 0:
12: if dec_pos == -1:
13: if len(right) < 2:
14: if len(right) == 2:
15: if len(right) > 2:
16: left = ""
17: left = before + "," + after
18: left = s
19: left = s[0 : dec_pos]
23: print "$" + left + "." + right
24: return "$" + left + "." + right
25: right = ""
26: right = right + zeros
27: right = right[0 : 2]
28: right = s
29: right = s[dec_pos + 1 : len(s)]
34: zeros = "00"
35: zeros = (2 - len(right)) * "0"