In [1]:

```
def hello_name(name):
  """
  Given a string name, e.g. "Bob", return a greeting of the form "Hello Bob!".
  """
  return "Hello " + name + "!"
```

In [2]:

```
def make_abba(a, b):
  """
  Given two strings, a and b, return the result of putting them together
  in the order abba, e.g. "Hi" and "Bye" returns "HiByeByeHi".
  """
  return a+2*b+a
```

In [5]:

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

In [6]:

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

In [7]:

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

In [8]:

```
def first_two(str):
  """
  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 "".
  """
  return str if len(str)<=2 else str[:2]
```

In [9]:

```
def first_half(str):
  """
  Given a string of even length, return the first half. So the string "WooHoo"
  yields "Woo".
  """
  return str[:len(str)/2]
```

In [10]:

```
def without_end(str):
  """
  Given a string, return a version without the first and last char,
  so "Hello" yields "ell". The string length will be at least 2.
  """
  return str[1:-1]
```

In [11]:

```
def combo_string(a, b):
  """
  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).
  """
  return a+b+a if len(a)<len(b) else b+a+b
```

In [12]:

```
def non_start(a, b):
  """
  Given 2 strings, return their concatenation, except omit the first char
  of each. The strings will be at least length 1.
  """
  return a[1:]+b[1:]
```

In [ ]:

```
def left2(str):
  """
```

```
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.
"""
return str[2:]+str[:2]
```