Passing functions as arguments

Exercise:

Functions hello and ciao have already been defined (see below). Write a new function, greeting. Greeting has two parameters, f, a function, and s, a string. Function greeting should call function f, for s. Function greeting will return None.

For example, given

```python
def hello(s):
    """
    print('Hello, ' + s + '.')
    return None

def ciao(s):
    """
    print('Ciao, ' + s + '.')
    return None
```

```python
>>> greeting(hello, 'World')
Hello, World.
```

```python
>>> greeting(ciao, 'World')
Ciao, World.
```

Next, add code to greeting to report the name of the function that is being called:

```python
>>> greeting(hello, 'World')
Calling hello
Hello, World.
```

```python
>>> greeting(ciao, 'World')
Calling ciao
Ciao, World.
```
Exercises: Dictionaries

(1) Given the following lists of dates and temperatures:

days = ['Mo', 'Tu', 'We', 'Th']
temps = [55, 23, 42, 44]

Write a function, createTempD, that takes the two lists and returns a dictionary with the days as keys and the temperatures as the values. Assign the result of executing createTempD to dd, which will be used in the exercises that follow.

(2) Use dd to find the temperature for 'We'.

(3) Add a temperature of 32 for 'Fr'.

(4) Create a sorted list of all of the temperatures in dd.
(5) Add a temperature of 60 for 'Saturday'.

(6) Delete the temperature for 'Saturday' and add a temperature of 60 for 'Sa'.

(7) Add a temperature of Saturday's temperature plus 10 degrees for 'Su'.

EXERCISES: Lists

Although Python provides us with many list methods, it is good practice and very instructive to think about how they are implemented. Implement a Python function that works like the following list methods (of course, don’t use the Python methods in your implementation):

(a) count
(b) Does the `my_in` function implement Python in i.e., return `True` if an item is in the list, and `False` otherwise? (If not, rewrite `my_in` so it implements Python in - without using Python in, of course.)

```python
def my_in(li, i):
    for item in li:
        if item == i:
            return True
    return False
```

(c) `index` – return `-1` if not in the list

(d) `reverse`