```python
# Some simple functions using lists as parameters.

# Print each individual element of a list on its own line.
def print_one_per_line(li):
    for val in li:
        print(val)

# Return the number of even integers in the list.
def count_evens(li):
    evens = 0
    for val in li:
        if val % 2 == 0:
            evens += 1
    return evens

# Return the number of strings of length 4 or greater in the list.
def long_strings(li):
    long = 0
    for st in li:
        if len(st) >= 4:
            long += 1
    return long

# Print the list in reverse order
def print_reverse(li):
    index = len(li) - 1
    while index >= 0:
        print(li[index])
        index -= 1
```

```python
# Lists are "mutable" -- they can be mutated, or changed, both by
# changing individual indices, or by adding or removing values.

faculty = ["Bo", "Frank", "Alvaro", "Birgit"]
print(faculty)

# We've already seen this:
faculty[0] = "Mehrdad"

# This is called "mutation". It changes one aspect of the variable faculty,
# but doesn't create a brand new object. faculty still points to the same
# object it used to point to, but that object has mutated.


# Other means of mutating lists:

# .append(x): given a new value x, add x to the end of the list.
faculty.append("Anthony")
print(faculty)

# faculty now has 5 elements, and has been mutated by appending the new value.


# This might seem similar to the following code:
x = 10
print(x)
x = 20
print(x)

# That might look like mutation, because x changed from 10 to 20... but it's
# not! The = means that we have *reassigned* x, not *mutated* it. x used to
# point to the object "10"; now it points to a different object "20".
# This is subtle, but important, and we'll cover it again later.


# .remove(x): find the first element equal to x, and remove it from the list
faculty.remove("Frank")
print(faculty)


# .pop(): remove the last item in the list
faculty.pop()
print(faculty)


# .pop(i): remove the item at index i
faculty.pop(0)
print(faculty)


# .insert(i, x): insert x into the list before index i
faculty.insert(1, "Neal")
```

```python
print(faculty)
```

```python
# doubleList: returns the same list as given, but with each element duplicated
# doubleList([1, 2, 3]) -> [1, 1, 2, 2, 3, 3]

def doubleList(li):
    answer = []
    for x in li:
        answer.append(x)
        answer.append(x)

    return answer



# listContains: true if the list contains a given value
# listContains([1, 2, 3], 2) -> True
# listContains([1, 2, 3], 5) -> False

def listContains(li, val):
    for x in li:
        if x == val:
            return True

    return False

# listEquals: true if the two lists have the same values in the same order.
# listEquals([1, 2, 3], [1, 2]) -> False
def listEquals(li1, li2):
    if len(li1) != len(li2):
        return False

    i = 0
    while i < len(li1):
        if li1[i] != li2[i]:
            return False
        i += 1

    return True

# isSorted: true if the list is sorted in nondecreasing order
# isSorted([1, 2, 3]) -> True
# isSorted([0, 0, 0, 0, 0]) -> True
# isSorted([1, 2, 4, 3, 5]) -> False

def isSorted(li):

    i = 0
    while i < len(li) - 1:
        if li[i] > li[i + 1]:
            return False

        i += 1
    return True
```

```python
print(isSorted(["hello", "jonathan", "long beach"]))
```