1. **Example** Write some examples of calls to your function\(^1\) and the expected returned values. Include an example of a *standard* case (as opposed to a tricky or corner case.) Put the examples inside an indented triple-quoted string.

```python
>>> length_is_multiple("two", 2)
False
>>> length_is_multiple("two", 3)
True
```

2. **Type Contract** Write a type contract that identifies name and type of each parameter. Choose a meaningful name for each parameter. Also identify the return type of the function. Put the type contract above the examples.

```python
@param str string: a string
@param int num: a whole number
@rtype: bool
>>> length_is_multiple("two", 2)
False
>>> length_is_multiple("two", 3)
True
```

3. **Header** Write the function header above the docstring and outdent it.

```python
def length_is_multiple(string, num):
    
    @param str string: a string
    @param int num: a whole number
    @rtype: bool
    >>> length_is_multiple("two", 2)
    False
    >>> length_is_multiple("two", 3)
    True
```

4. **Description** In the same line as the opening triple-quote mark, put a one-line summary of what the function does. If necessary, you can put an optional, longer description above the type contract. Mention each parameter by name.

```python
def length_is_multiple(string, num):
    
    """Return whether num evenly divides length of string."
    
    @param str string: a string
    @param int num: a whole number
    @rtype: bool
    >>> length_is_multiple("two", 2)
    False
    >>> length_is_multiple("two", 3)
    True
```

---

\(^1\) Do not include examples for functions that involve randomness or user input.
5. **Body** Write the body of the function by remembering to indent it to match the docstring. To help yourself write the body, review your example cases from step 1 and how you determined the return values. You may find it helpful to write a few more example calls in the docstring.

```python
def length_is_multiple(string, num):
    """Return whether num evenly divides length of string.
    @param str string: a string
    @param int num: a whole number
    @rtype: bool
    >>> length_is_multiple("two", 2)
    False
    >>> length_is_multiple("two", 3)
    True
    """
    return len(string) % num == 0
```

6. **Test Your Function** Test your function on all your example cases including any additional cases you created in step 5. Additionally try it on extra tricky or corner cases. In order to automatically test your docstring examples you can include the following at the end of the file:

```python
if __name__ == '__main__':
    import doctest
doctest.testmod()
```