CSC108H Worksheet: Dictionaries

1. Consider this code:

```
name_to_binomial = {'human': 'Homo sapiens',
                    'dog': 'Canis familiaris',
                    'narwhal': 'Monodon monoceros'}
```

Circle the expressions that evaluate to `True`.

(a) `'dog' in name_to_binomial`

(b) `'Canis familiaris' in name_to_binomial`

(c) `name_to_binomial[0] == 'human'`

(d) `len(name_to_binomial) == 3`

(e)
```
name_to_binomial == {'dog': 'Canis familiaris',
                     'narwhal': 'Monodon monoceros',
                     'human': 'Homo sapiens'}
```

2. Consider this code:

```
animal_to_locomotion = {'fish' : ['swim'],
                        'kangaroo' : ['hop'],
                        'frog': ['swim', 'hop']}
```

Indicate whether each statement will cause an error and, if not, whether the statement will increase the number of key/value pairs in the dictionary:

Statement	**Error? (yes or no)**	**Increases length of dictionary? (yes or no)**
`animal_to_locomotion['human'] = ['swim', 'run', 'walk', 'airplane']`		
`animal_to_locomotion['orangutan'].append('brachiate')`		
`animal_to_locomotion['kangaroo'].append('airplane')`		
`animal_to_locomotion['frog'] = ['tapdance']`		
`animal_to_locomotion['dolphin'] = animal_to_locomotion['fish']`		

CSC108H Worksheet: Dictionaries

3. The express checkout is for grocery orders with 8 or fewer items. Complete the examples in the docstring and then complete the function body.

```
def express_checkout(product_to_quantity):
    """ (dict of {str: int}) -> bool

    Return True iff the grocery order in product_to_quantity qualifies for the
    express checkout.  product_to_quantity maps products to the numbers of those
    items in the grocery order.

    >>> express_checkout({'banana': 3, 'soy milk': 1, 'peanut butter': 1})
    
    >>> express_checkout({'banana': 3, 'soy milk': 1, 'twinkie': 5})
    
    """
```

4. As part of a study funded by a major shoe company, we crouched at the finish line of the Boston marathon and kept an ordered list of the shoes that we saw as they passed the finish line. Write a function that, given such a list, will return a dictionary mapping shoe companies to a list of the placements achieved by runners wearing shoes made by those companies.

```
def build_placements(shoes):
    """ (list of str) -> dict of {str: list of int}

    Return a dictionary where each key is a company and each value is a
    list of placements by people wearing shoes made by that company.

    >>> build_placements(['Saucony', 'Asics', 'Asics', 'NB', 'Saucony',
                          'Nike', 'Asics', 'Adidas', 'Saucony', 'Asics'])
    {'Saucony': [1, 5, 9], 'Asics': [2, 3, 7, 10], 'NB': [4], 'Nike': [6], 'Adidas': [8]}
    """
```