

COMP 690 Data Fusion Fall 2008 Quiz 1—Solutions

20 points total

1 (9 pts.). Write a function

```
function add_ds(d1, d2)
```

where `d1` and `d2` are dictionaries whose keys are names (strings) and values are integers. We'd expect `d1` and `d2` to have some keys in common. For each key in `d1`, if that key also exists in `d2`, then add the associated value in `d2` to the associated value in `d1`. There is no explicit return; rather, the caller finds the first argument updated. Be sure not to try to access a dictionary with a non-existent key.

The following is an example run.

```
>>> d1 = {'Fred': 2, 'Al': 4, 'Ed': 6}
>>> d2 = {'Bill': 1, 'Fred': 3, 'Bob': 5, 'Ed': 7}
>>> add_ds(d1, d2)
>>> d1
{'Ed': 13, 'Al': 4, 'Fred': 5}
```

Answer

```
def add_ds(d1, d2):
    for n in d1.keys():
        if d2.has_key(n):
            d1[n] += d2[n]
```

2 (9 pts.)The listing (with gaps) below shows the definition of a function

```
def trunc_lists(lists):
```

that, given a list of lists of various lengths, returns a list of lists all of the same length, namely, the length of the shortest original list. Furthermore, each list in the result consists of an initial slice of the corresponding list in the input (which is the entire list for those of minimum length).

The following is an example run.

```
>>> ls = [[1, 2, 3], [4, 5], [6, 7, 8, 9]]
>>> trunc_lists(ls)
[[1, 2], [4, 5], [6, 7]]
```

The gaps are labeled with Greek letters. After the listing, the Greek letters are repeated with descriptions of what goes into the corresponding gaps. You then supply the missing code. The lists returned are copies of (parts of) the original lists. Note that a slice of a list is a copy of (part of) the list, not an alternative reference to it. In particular, if `a` is a list, then `a[:]` is a copy of all of `a`.

```
def trunc_lists(lists):
    import sys
    min_len = sys.maxint

    # Set min_len to the length of the shortest list
    for x in range(len(lists)):
        α
        _____

    new_lists = []

    # Copy initial slices of length min_len of successive
    # lists in 'lists' into 'new_lists'
    for x in range(len(lists)):
        β
        _____

    return new_lists
```

α: If `list[x]` is shorter than any list checked so far, then update `min_len` accordingly

Answer

```
if len(lists[x]) < min_len:
    min_len = len(lists[x])
```

β: Add a new list to `new_lists`, namely, the initial slice of `list[x]` of length `min_len`.

Answer

```
new_lists.append(lists[x][:min_len])
```

3 (2 pts.). Write a map expression that, given a list of integers, returns an identical list except every odd integer is replaced by an integer greater than it by 1

Answer

Where variable `a` is bound to a list of integers,

```
map(lambda x: x+1 if x % 2 == 1 else x, a)
```

Another possibility is

```
map(lambda x: x + x % 2, a)
```