# CS153/453 Fall 2017

HW 8

Due: Monday Oct 30, 2017 11:59pm (before midnight)

**Task 1**

Consider the solution to Task 2 of HW 7.

(a) For the function `hidden_word`, rewrite the codes using list comprehension and the join operation of string.

(b) Rewrite the function `success` by checking to see if the character `'-'` appears in `hidden_word`. Another way is to use the `all` built-in function for list.

Note: for both functions, you should implement the logic in *one line* of code.

**Task 2**

Consider the solution to Task 4 of HW 7.

(a) Write a function `valid_key( key )` that returns True if `key` is a valid key that is of length 27, and its content is permutation of the string `alphabet = "abcdefghijklmnopqrstuvwxyz "`, and False otherwise.

(b) Rewrite the function `codedText` using list comprehension and the join operation of string.

(c) Rewrite the function `substitutionEncrypt` using dictionary comprehension.

(d) Rewrite the function `substitutionDecrypt` using dictionary comprehension.

Note: for each of part of this question, you are asked to write the codes in *one single line*. Try to express your ideas in a high level manner using concepts like list comprehension, set comprehension, dictionary comprehension, and function call.

**Task 3** (for CS453 only) (Problem is taken from TopCoder Inc.)

Problem Statement:

A health club chain allows its members to visit any of its many health club locations an unlimited number of times per day. The only constraining rule is, a customer can only visit one health club location per day, even though he or she may return to that location an unlimited number of times for the rest of that day.

Although the honor system has always worked quite well, the club wants to run some tests to see how many people really follow the rules. You are to write a program that takes the entrance log files from three different clubs (all logging the same day) and return a sorted list of the people who are not honest and went to more than one health club location in the same day.

The log files are represented as lists of String's where each element is the member name of a customer who entered that day. For example, if a customer showed up three times to one of the club locations that day, the member's name would appear three times in the corresponding list of Strings.

Notes:

- club1, club2, and club3 may contain a different number of elements.
- The same member name can appear multiple times in a single log file. The elements of the returned String list should be sorted in lexicographic order (the order they would appear in a dictionary).
- Assume that two people with the same name are in fact the same person.

Constraints:

- club1, club2, and club3 each have between 1 and 50 elements, inclusive.
- Each element of club1, club2, and club3 contains between 1 and 50 characters, inclusive.
- Each element of club1, club2, and club3 consists only of uppercase letters ('A'-'Z').

Examples:

1. ```
   ["JOHN","JOHN","FRED","PEG"]
   ["PEG","GEORGE"]
   ["GEORGE","DAVID"]

   Returns: [ "GEORGE", "PEG" ]
   ```
   "PEG" went to club1 and club2, and "GEORGE" went to club2 and club3.

2. ```
   ["DOUG","DOUG","DOUG","DOUG","DOUG"]
   ["BOBBY","BOBBY"]
   ["JAMES"]

   Returns: [ ]
   ```
   Here, no one went to more than one club location.

3. ```
   ["BOBBY"]
   ["BOB","BOBBY"]
   ["BOB"]

   Returns: [ "BOB", "BOBBY" ]
   ```
   Note that "BOB" is sorted before "BOBBY"

4. ```
   ["BOBBY","HUGH","LIZ","GEORGE"]
   ["ELIZABETH","WILL"]
   ["BOB","BOBBY","BOBBY","PAM","LIZ","BOBBY","BOBBY","WILL"]

   Returns: [ "BOBBY", "LIZ", "WILL" ]
   ```

5. ```
   ["JAMES","HUGH","HUGH","GEORGE","ELIZABETH","ELIZABETH","HUGH",
   "DAVID","ROBERT","DAVID","BOB","BOBBY","PAM","JAMES","JAMES"]
   ["BOBBY","ROBERT","GEORGE","JAMES","PEG","JAMES","DAVID","JOHN","LIZ",
   "SANDRA","GEORGE","JOHN","GEORGE","ELIZABETH","LIZ","JAMES"]
   ["ROBERT","ROBERT","ROBERT","SANDRA","PAM","BOB","LIZ","GEORGE"]

   Returns: ["BOB","BOBBY","DAVID","ELIZABETH","GEORGE","JAMES","LIZ","PAM",
             "ROBERT", "SANDRA" ]
   ```

6. ```
   ["LIZ","WILL","JAMES"]
   ["JOHN","ROBERT","GEORGE","LIZ","PEG","HUGH","BOB","BOBBY","ROBERT","ELIZABETH","DAVID"]
   ["PAM","DAVID","SANDRA","GEORGE","JOHN","ROBERT","SANDRA","GEORGE"]

   Returns: [ "DAVID", "GEORGE", "JOHN", "LIZ", "ROBERT" ]
   ```

7. ```
   ["PEG","ROBERT","PAM","JOHN","DAVID","JOHN","ROBERT",
   "GEORGE","HUGH","WILL","JAMES","JAMES","BOBBY","BOBBY","SANDRA"]

   ["SANDRA","BOB","PAM","JAMES","WILL","DAVID","BOBBY","GEORGE",
   "WILL","LIZ","BOBBY","ROBERT","WILL","BOB","BOBBY","ELIZABETH","HUGH"]

   ["WILL","PEG","ELIZABETH","DAVID","HUGH","BOBBY","JOHN","SANDRA","ELIZABETH",
   "ELIZABETH","SANDRA","GEORGE","PAM","ELIZABETH","BOBBY","DAVID","PAM"]

   Returns: ["BOBBY","DAVID","ELIZABETH","GEORGE","HUGH","JAMES","JOHN","PAM",
             "PEG","ROBERT","SANDRA","WILL"]
   ```
   ```

Submit a program using the following format:

```
filename: MemberCheck.py

def whosDishonest(club1, club2, club3):
    """
    return sorted list of Strings, the names
    that appear in more than one of the
    String lists club1, club2, club3
    """

    # write code here
```

You are asked to write the program using concepts of set, list, list comprehension, and union operation of sets, and `sorted` function.

Your codes should be as clean as possible. You can write the answers in four lines of codes.