

CS10 With-Computer Final (Fall 2018, Sec 1)

There are three questions, two Snap! ones and a Python one. Save your Snap! code into a Snap! file, and name it **FinalYourfirstnameYourlastname.xml** (e.g., **FinalAlanTuring.xml**). For the Python question, create a new Python file and name it **FinalYourfirstnameYourlastname.py** (e.g., **FinalAlanTuring.py**). Submit both files on bCourses under the “online” final assignment for your lab section. All questions are independent, and each worth 5.

Snap! Questions: (use this starter file: <https://bit.ly/2xFaabY>)

Palindrome words are those which read the same backward as forward, e.g., OTTO, I and ANA. A better name for these would be *PalindromeALL* words, because ALL letters have to match backward as forward. We’re interested in *PalindromeANY* words, in which ANY letters can match, e.g., OTTO, I, ANA, CAL, and REAR. You guessed it, STANFORD is not a *PalindromeANY* word, since no letters match.

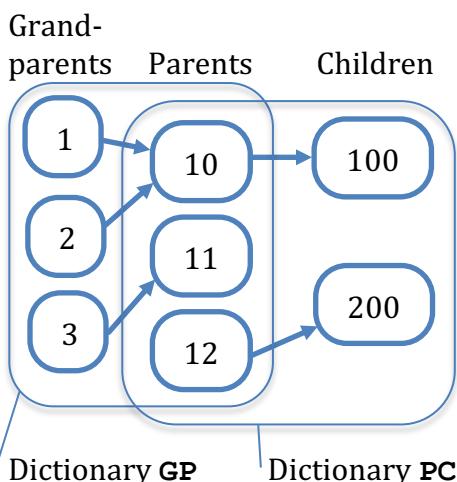


- Write it *recursively*. You may not use any iteration (`repeat`, `repeat until`, `for`, `for each`) or higher-order functions in this solution.
- Write *without using recursion*. You can earn +3 bonus points if you can do it with only higher-order functions (i.e., only `map`, `keep` and `combine` to drive the iteration). Here are three helpers you might find handy (note the `map` shown below is in addition to the built-in `map`).



Python Question:

Write a function that `find_GC` that takes in two dictionaries (**GP** capturing grandparents→parents, and **PC** capturing parents→children) and *returns a new dictionary of all grandparents→children it finds*. As an example, we have three grandparents: 1, 2 and 3; three parents: 10, 11 and 12; and two children: 100 and 200 with → connections as shown below. Your function would return the two grandparents→children: 1→100 and 2→100. By the way, more than 2 grandparents can → to the same parent; similarly for parents→children (sometimes family records get corrupted, it’s not our job to worry about that).



```
>>> GP = {1:10, 2:10, 3:11}
>>> PC = {10:100, 12:200}
>>> find_GC(GP,PC)
{1: 100, 2: 100}
```

If you forget any commands, remember there’s `help(type)` and `dir(type)`, as in `help(dict)` or `dir(str)`.