**CSE 212 – Programming with Data Structures**

**W05 Prove – Response Document**

|  |  |
| --- | --- |
| **Name:** | Jorge A Chavez |
| **Date:** | 14/10/2023 |
| **Teacher:** | Ephraim Kunz |

*It is a violation of BYU-Idaho Honor Code to post or share this document with others or to post it online. Storage into a personal and private repository (e.g. private GitHub repository, unshared Google Drive folder) is acceptable.*

**Question 1: From Part 1, how did you answer the interview question for the Set Operations problem (should be no more than 30 seconds if spoken aloud)?**

For the problem of Set Operations, I have worked it out in the following way. To find the intersection, we iterate through one set, check if each element is present in the other set, and add common elements to the intersection set. For the union, we iterate through both sets and add their elements to the union set. Thanks to set(), when we try to add an element that is already present in the union set, the set simply stays the same, without adding duplicates.

**Question 2: From Part 2, how did you answer the interview question for the Find Pairs problem (should be no more than 30 seconds if spoken aloud)?**

For the Find Pairs problem, I provided a solution that finds symmetric pairs of two-letter words in a list with O(n) time complexity using sets. The algorithm loops through the list once, stores the seen words in a set, and identifies symmetric pairs by checking whether the inverted word is in the set of seen words. Symmetrical pairs are printed in the specified format. This approach efficiently uses sets to identify and display symmetric pairs, ensuring optimal O(n) time complexity.

Remember: You need to submit the following code files in addition to this document:

* 05-prove\_set\_operations.py
* 05-prove\_find\_pairs.py