CS 562

Student: Weiyu Lin

Student ID:932-479-491

Testing Project Proposal

- **Intro the test python library**

This proposal will be focused on testing classical algorithms and data structure. A third party Python library (Link: https://github.com/nryoung/algorithms) on Github named "Algorithms" will be tested by TSTL. This 1000 more stars Python library has 8 sections, which are data structures, dynamic programming factorization, math, random, searching, shuffling, sorting. All of the eight sections have been learned on CS261 and CS325 or other EECS courses. This proposal is only focusing on the data structure and searching algorithms.

- **What to test**

a. Data Structure

The data structure section contains 9 modules: binary_search_tree, digraph.py, queue.py,stack.py,undirected_graph.py,undirected_graph.py,union_find.py,union_find_by_rank.py,union_find_with_path_compression.py. These modules include inserting ,delete, searching functions. all the functions will be tested by TSTL. Six ways I will use to testing:

1. Check the correctness of data structure by passing one value into specific data structure to see if getting corresponding answer.

2.Check the robustness of the code.

3. Make sure the performance of data structures follow their properties. For example, push a few numbers into a Stack data structure, and pop them out to see whether the stack follow the FIFO (first in first out) property.

4. Check whether all the functions are work well. For instance, I will pop an empty stack to see if there still have values popped out.

5. All the data structure will be tested by different type of values to see if there

are any mistakes generate.

6. Any other test methods when need.

b. Searching algorithm

The searching section contain 6 modules: binary_search.py, bmh_search.py, breadth_first_search.py , depth_first_search.py, kmp_search.py, rabinkarp_search.py. All the function in above modules will be tested by TSTL. There are five ways I will use to testing:

1. Input series values and index, to see if the result fit the correct answer.
2. Check the robustness of the code.

3.check if the time complexity fit the corresponding algorithm. For example, the best case of binary search should be O(1), and the worst case should be O(log n). I can input specific values then plot the running times as function of input size and analysis if the running time fit the properties of binary search.

4. Check whether all the functions are work well. For instance, I will search an given index in an empty array by binary search tree to see if it still get results.

5. Insert different type to see if the error occurs.

6. Any other test methods when need.

- **Purposes and reasons**

I have learned the data structures and searching algorithms on CS261 and CS325. In order to well understand how to testing by TSTL and obtain useful testing skills, I decide to choose something I familiar. This is the reason I choose a Python library with data structures and searching algorithms. On the other side, by testing these classical data structure and algorithms, I can reinforce these useful data structure and algorithms knowledge, which I have learned on CS 261 and CS 325.