Bug/progress report

Part 1: Bugs

In this part, I have tested several different data structures and try to find out the bugs they may have. At the beginning, I tested the stack which is the easiest one. However, the stack runs well until now (actually the author used the python build-in function to build his push() function and pop() function() in his stack data structure). When I was testing the linked list. I try to test whether the remove function works well. When I delete a data existed in the linked list, I check whether the size of the link list equals to the previous size minus one (`<linklist,1>.size == pre<(<linklist,1>.size)>-1`) and the data deleted is still in the linked list (`<data,1> not in <linklist,1>.stack`) or not. The tstl reminds me there is a bug here. I thought I found a bug. Actually, after deep testing I think it is my test code’s fault. Because if the linked list includes two same data, which is legal, there will be one data still in the linked list after using remove() function to remove one of them. So I am still thinking a way to test if the linked list removes the data. So I don’t find any bug now. I will keep testing these data structure and finding if there are some bugs.

Part 2: Explain progress to date

I am still learning at tstl. Actually, I used to use python to just activate my C code. So I am not familiar with the python. But in some way, Python is similar to C language and most of time Python’s grammar is more concise. At first, I can just test the size of the data structure. When I am testing the push() function and pop() function in the stack data structure, I can just check if the size gets bigger or smaller to test the code is right or not. Now I know how to test if the data I added in the data structure. I have tested if these data structure can add the data in a right way. For the delete (remove or pop) functions, I can just test part of the data structures. Like the problem I mentioned, I cannot make sure the data is removed in linked list or some other data structure when there are two or more these data existed in the data structure. I still need to know whether the number of the data in the data structure reduces or not.

Part 3: Estimate your progress by end of term

By the end of term, I will test if the data structure can add or remove data in a right way. And for the AVL tree I will test if the tree rotates in a right way.

Part 4: Discuss quality of the SUT

In my opinion, the python library I tested is not completed. For example, in the AVL tree, there is no delete function. So it can just add data but it cannot delete data in the AVL tree. And the author used a lot of build-in data structure function to build his own data structure. For example, in the stack.py, the author used stack.pop(data) to pop the data. However, this python library did well in data structure building, especially building the rotate function in the AVL tree. And I cannot find any bug in his existed functions.

Part 5: code coverage

I have tested all the add functions for these data structures and parts of the delete functions, such as pop(). What I don’t cover is the remove functions for the linked list, queue and heap. For the AVL tree, I am still working at testing its rotate function.