Bug/progress report

Brief Description:
As I have introduced in the proposal, my python file has fourteen functions. Linked list is a basic data structure. So I am familiar with how does linked list work and what part of the list can find tricks. From now, I have tested a half of them. During my tstl test, I haven't found any bug in the python file. To the code I have tested, I think the python file has strong robustness.

Explain progress:
At the beginning of the project, I just learned what is tstl. Then I tried to figure out what is the tstl's syntax. In the part two of this project, I learned that I could defined my own functions to test the python bracketed by "<@ @>". Then if I wanted to test a function in the python file, I could call the function and build a property checker to prove all the actions that the function has done are correct. For example: I wanted to test insert() function in the LinkedList.py. I should get random numbers from pool and pass them to the insert() function (~%LINKEDLIST%.insert(%INT%)). Then I should create a property checker to prove the insert() function. So my logic was after inserting, the length of the list equals to the length of list before inserting +1 and the data I had inserted into the list was at the front of the list. This logic could be changed to a tstl syntax:

```
(length(%LINKEDLIST,1%) == PRE%(length(%LINKEDLIST,1%))%+1) and (checkInsert(%LINKEDLIST,1%,%INT%) == True )
```

So from starting to now, I have tested insert, append, returnIndex, updateIndex, deleteIndex, insertBeforeIndex and insertAfterIndex function. During testing, I found that only use the functions which are provided by the python file is not enough. I have to write some helping functions to help me to test the python. So I have write length, empty, checkInsert, checkAppend, and returnvalue functions to help me to build the property checkers. For instance, the checkInsert helping function helps me to check that after inserting, the first element equals the data I have inserted in.

Estimate Future Progress:
My python file has fourteen functions. I have already tested a half of them. So in the rest of the term, I will still finish testing work. Also I find the logic of the tstl code which I have already written is not accurate. I will add more testing conditions to consummate the property checkers. For example, when I tested the deleteIndex function, I just check that after deleting a number, the length of the list equals to the length of list before inserting -1. I will add one testing condition:

1. Initial two lists in the pool
2. Delete a number in a specific index by deleteIndex function in one list
3. Delete the same number in the same index by hand in the other list
4. Compare two lists

Quality of The SUT

I think this python file has a strong robustness. I didn't find any bug so far. Besides the testing result, when I read the code at the first time, I think the author of the python considered many special conditions. So I feel confident to this python file.

Code Coverage

```
Coverage.py warning: No data was collected.
36.8794326241 PERCENT COVERED
4.60739612579 TOTAL RUNTIME
200 EXECUTED
20000 TOTAL TEST OPERATIONS
4.0852060318 TIME SPENT EXECUTING TEST OPERATIONS
0.31797337532 TIME SPENT EVALUATING GUARDS AND CHOOSING ACTIONS
0.0147712230682 TIME SPENT CHECKING PROPERTIES
4.09997725487 TOTAL TIME SPENT RUNNING SUT
0.0871620178223 TIME SPENT RESTARTING
0.0 TIME SPENT REDUCING TEST CASES
95 BRANCHES COVERED
69 STATEMENTS COVERED
```

So far, my code coverage is about 36.87 percent. I have tested a half of the functions. So my conservative estimation of code coverage will be close to 80 percent. To reach this perspective, I should go on testing the rest of the code and refining the property checkers. I hope to find some bug of the python in the rest of the term which can give me more information about how to use tstl to test a code.