Search Function ---- Test Plan

1. ***Introduction***

   The function implements a linear search for an array of sorted numbers. The search type could be varied, such as find a number less than the input number, or find a number exactly same with the input number. The function will return the search result and the corresponding index. We will set the index to be -1 if the search result is "Not Found". See Readme file for the detail.

2. ***Required Resources***

   Test machines with multi-processor.

3. ***Test Plan***

   (1) Functional Test

   Wrote functions to generate an increasing/decreasing array, search key number, search type and search result, comparing the result with the function output.

   (2) Scalability Test

   Parallelize the function, and do the functional test on multi-processor cluster and distributed system.

   (3) Performance Test and Stability Test

   Use two machines, one is used as a server (system under test), provide the parallelized search function and store large search data. Another is used as a client, generate a large number of search request. We will test the performance and the stability of the function, such as the response time, memory consumption and CPU consumption.

   We also proposed an optimized method to use a hash table as a cache to memorize the tuple computation(See function FindTuple()). We will write a performance test function to see the time/memory/CPU improvement.

4. ***Will not Test***

   We have assumptions that: the items are sorted, items will be non-NULL, there are no duplicate items and number of items will be greater than 0. Thus, we will not test the illegal case and boundary case.

5. ***Expected Result***

   1. The parallelized function is correct and stable, the respond time is less than 1 second.

   2. Use hash table as a cache could improve the respond time. Will report the memory consumption to tune a proper size of hash table.