Project 1: Performance Measurement

Given a list of ordered N integers, numbered from 0 to N–1, checking to see that N is not in this list provides a worst case for many search algorithms.

Consider two algorithms: one is called "*sequential search*" which scans through the list from left to right; and the other is "*binary search*" which is given on page 23 of your textbook. Your tasks are:

(1) Implement an iterative version and a recursive version of binary search, together with an iterative version of sequential search;

(2) Analyze the worst case complexities of the above three versions of searching methods;

(3) Measure and compare the worst case performances of the above three functions for N = 100, 500, 1000, 2000, 4000, 6000, 8000, 10000.

To measure the performance of a function, we may use C's standard library **time.h** as the following:

```
#include <time.h>
clock_t   start, stop; /* clock_t is a built-in type for processor time (ticks) */
double   duration;   /* records the run time (seconds) of a function */

int main ( )
{     … …
      /* clock() returns the amount of processor time (ticks) that has elapsed
         since the program began running */
      start = clock();     /* records the ticks at the beginning of the function call */
      function();          /* run your function here */
      stop = clock();      /* records the ticks at the end of the function call */
      duration = ((double)(stop - start))/CLK_TCK;
      /* CLK_TCK is a built-in constant = ticks per second */
      … …
      return 1;
}
```

Note: If a function runs so quickly that it takes less than a tick to finish, we may repeat the function calls for K times to obtain a total run time, and then divide the total time by K to obtain a more accurate duration for a single run of the function. The repetition factor must be large enough so that the number of elapsed ticks is at least 10 if we want an accuracy of at least 10%.

The test results must be listed in the following table:

	N	100	500	1000	2000	4000	6000	8000	10000
Binary Search (iterative version)	Iterations (*K*)								
	Ticks								
	Total Time (sec)								
	Duration (sec)								
Binary Search (recursive version)	Iterations (*K*)								
	Ticks								
	Total Time (sec)								
	Duration (sec)								
Sequential Search	Iterations (*K*)								
	Ticks								
	Total Time (sec)								
	Duration (sec)								

The performances of the three functions must be **plotted** in the **same** *N*–run_time coordinate system for illustration.

Grading Policy:

This assignment is due Friday, September 25th, 2009 at 10:00pm.

- **Programmer:** Implement the three functions **(30 pts.)** and a testing program **(20 pts.) with sufficient comments**.
- **Tester:** Decide the iteration number *K* for each test case and fill in the table of results **(8 pts.)**. Plot the run times of the functions **(12 pts.)**. Write analysis and comments **(10 pts.)**.
- **Report Writer:** Write Chapter 1 **(6 pts.)**, Chapter 2 **(12 pts.)**, and finally a complete report **(2 pts. for overall style of documentation)**.