# Lab 5: 1D Array

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Write a program to generate ***M*** random integers and put them in an array, then check how random your random number generator is! Generate another sequence of random numbers ***N*** and count how many times it occurs in the array – using a sequential search.
2. Run your program for M= 10, 100, 1000 and N= 10, 100, 1000, timing the result using clock()
3. Change the logic of the above code to work recursively.
4. Implement recursive Fibonacci code from lecture pseudocode, print out 50th term.
5. **Timing**

Time how long your algorithm takes to run. Use the C timing library to record the start and end of your program’s run time – the actual run time is the difference between the end and the start.

1. Include the <time.h> library,
2. Declare two variables – for the start and end time values
3. Start the clock – assign the value of the clock() function to the start
4. End the clock – capture the clock() return
5. Calculate the difference between them for the elapsed time.
6. Display the result. Note the result is cast from time\_t to double.
7. The result is in clock ticks – to display seconds, divide by the architecture’s value for clock ticks per second – CLOCKS\_PER\_SEC. (this is a #define in the time.h file)

int main()

{

time\_t pStart,pEnd; //time\_t is a variable type

pStart = clock(); //Record start time

/\* do your code \*/

pEnd = clock(); //record end time

printf (

"Elapsed time is %1.7lf seconds\n",

(double)(pEnd-pStart)/CLOCKS\_PER\_SEC);

}