DATA STRUCTURES & ALGORITHMS 1

BATCH – A [WEDNESDAY MARCH 06, 2019: 3:30 PM – 6:30 PM]

LAB ASSIGNMENT – 7 CODE:assign07

NOTES:

1. Please carefully read all assignments and there is **no choice**.
2. **Use the template for this assignment**
3. Each problem in this assignment has to be answered in the **same c file**.
4. Create a .c file following the **file name convention**:
    a. If your roll number is 'abc' and assignment code is 'assignXX'. Then use the following file name convention as follows: 'abc-assignXX.c'
    b. For example, if the roll number is 92 and assignment code is assign01, then the file name should be 092-assign01.c
    c. Strictly follow the file name convention. When you are ready, submit the solution via google classroom.
5. **Follow variable and function naming conventions**
    a. except for variables in for-loop, none of the other variables should be a single character.
    b. The variable names and function names should indicate what they are storing/computing. For this assignment, we have given you some of the variable names and function names to use. They are highlighted as function_name or variable_name
    c. All global variable should start with 'g_'
6. Indentation improves readability. Please pick an indentation style and **indent your code** appropriately.
7. Follow constants and type naming
    a. All constants should be defined using IFNDEF and DEFINE
    b. All structures should have a TYPEDEF to a simpler name
8. When in doubt about naming or style conventions, consult the following link: https://users.ece.cmu.edu/~eno/coding/CCodingStandard.html

---

```
#include <iostream>
#include <time.h>
using namespace std;

timespec diff(timespec start, timespec end);

int main()
{
        timespec time1, time2;
        int temp;
        clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &time1);
        for (int i = 0; i< 242000000; i++)
                temp+=temp;
        clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &time2);
        cout<<diff(time1,time2).tv_sec<<":"<<diff(time1,time2).tv_nsec<<en
        return 0;
}

timespec diff(timespec start, timespec end)
{
        timespec temp;
        if ((end.tv_nsec-start.tv_nsec)<0) {
                temp.tv_sec = end.tv_sec-start.tv_sec-1;
                temp.tv_nsec = 1000000000+end.tv_nsec-start.tv_nsec;
        } else {
                temp.tv_sec = end.tv_sec-start.tv_sec;
                temp.tv_nsec = end.tv_nsec-start.tv_nsec;
        }
        return temp;
}
```

PROBLEMS [Total Marks: 20]:

The above code snippet shows an example of how you can profile your code using time.h Even though, the above code is in C++, its the same process for C.

1. You should implement multiple ways of adding a sum of ‘n’ numbers. You can assume a max_size (say 50). This means that you can store the numbers in an array. [6 Marks]
    a. Using loops (for or while)
    b. Recursive
    c. Gauss technique (only work for a continuous sequence, but that's ok)
    d. Using a formula [hint: n(n+1)/2] (only work for a continuous sequence, but that's ok)
    e. Using multiple pointers (like we saw in class today)
2. Please profile your code as shown above. [6 Marks]
3. Please improve your code to take in command line arguments (‘n’ and the numbers themselves) [4 Marks]

4. Please improve your code further to take ‘n’,min, max as command line arguments and generate n random numbers within (min-max) range and use that as input for algorithms [4 Marks]