cs201 – sum 2020

# Assignment 4 – (40 points)

In this assignment, you will use the assign4.c file; Please download and modify this file. Upload the modified file when done to the d2l assignment folder before the due date. You may want to look at the example code in week 8 module for how to optimize this code.

**Please note the requirements and follow them for full credit.**

## *Optimize a loop program*

In this assignment you will optimize code and get it to run faster using techniques you have learned in this course. You will measure code execution time using the Linux time(1) command. This command prints out three numbers:

- **real**: total elapsed time in minutes and seconds
- **user**: total cpu usage of the application code in minutes and seconds
- **sys**: total cpu usage of the application's operating system calls in minutes and seconds

For the purposes of this assignment, you will use the **user** statistic returned by time command. To run it, just use 'time' followed by ./ and the name of the executable. For example:

```
time ./assign4
```

**You should do this work on syccuxas01.pcc.edu**. This is where I will be testing your code, and execution times on this machine are the standard that your code will be judged by. The fact that your code may have met the timing standard on some other machine does not count, and will not help you.

The code for this assignment has an outer loop and an inner loop. The inner loop computes the sum of the numbers in an array (note that the array is initialized by **calloc()** to all zeroes). The outer loop repeats the inner loop's computation the specified number of times. You should maintain this loop structure as you optimize this code. See Requirements below for what you can or cannot do.

The basic idea is to change the contents of the inner loop to make the computation run faster while having it still repeatedly compute the sum of the array elements. You can change the inner loop's header elements if needed. Also you can put other lines of code above or below the outer loop structure.

HINT: Look up 'loop unrolling' under the web page "Optimization" in week 8 module. This is what I have in mind for this assignment and you will get full credit for using this technique.

## <u>Grading</u>

Points will be awarded for the following criteria:

- For 40 out of 40, your code should have a **user run time less or equal to 27 seconds**.
- For 35 points out of 40 credit, your code should have a **user run time less or equal to 35 seconds**.
- For 25 points out of 40 credit, credit, your code should have a **user run time less or equal to 55 seconds**.
- **No points if for any code running over 55 seconds**.

### <u>Requirements</u>

**Your sum must be correct and match the given sum in the printf statement. Otherwise you will get no points.**

You need to abide by the following rules:

1. You should not alter the outer loop
2. You should not change the size of the array or its data type
3. The array should remain initialized the way it is - do not change any element of the array.
4. Your inner loop should still compute the sum of all the elements of the array. The printf statements for the printing the sum shouldn't be modified.
5. your code should be no longer than 100 lines
6. your code should not use the "register" keyword
7. Don't use any special mathematical formulas to do the sum; just do an accumulative sum. **All of the numbers in the array need to have been accessed and used at some time during the running of the inner loop.**
8. Do not write any assembly language for this assignment. You can achieve the speedup you need at the C code level.
9. **Use the lowest optimization level of gcc**. I will be compiling your code using the -O0 option. Use this for your test runs.

## <u>Example Output</u>

```
----------------------
EXAMPLE RUN #1:
This is a sample compiling and running of the code unmodified.  It takes about 1
minute to run.  If your unmodified file doesn't take this long email me about it.
----------------------
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln> gcc -O0 assign4.c -o
assign4
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln> time ./assign4
sum is 5000050000
sum should be 5000050000

real   1m1.631s
user   1m1.618s
sys     0m0.002s
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln>

----------------------
EXAMPLE RUN #2:
This is a sample compiling and running of the code with some modifications to the
code.
----------------------
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln> gcc -O0 assign4.c -o
assign4
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln> time ./assign4
sum is 5000050000
sum should be 5000050000

real   0m27.700s
user   0m27.688s
sys     0m0.005s
dona.hertel@syccuxas01:~/cs201_sumAss4/code/assign4_soln>
```