CS29206: Systems Programming Lab
Spring 2025
Assignment 3

In this assignment, you will write a C source program that computes all the factors of an input integer number, identify the ***working factors*** (which are in the ***working zone*** of a supplied program, see below point 5 for more details on working zone), and finally use the working factors to list which function in the supplied program is called how many times by how many other function(s) and is taking how much time (in percentage).

Please follow instructions carefully.

1. Create a directory named ***A3<your roll no>*** under your home directory (directory you are in when you log in). For example, if your roll no. is 23CS10002, the directory should be named ***A323CS10002***. All instructions after this in this assignment will use this directory name, ***A323CS10002***, as example, you should interpret it as your actual directory name you will use based on your own roll no.

2. Copy the assignment tar file (***Assignment3.tar***) in your directory. Open the tar file (***tar xvf Assignment3.tar***). Make sure you have the following files:

   (a) A file named ***checkZone*** (type: Executable file)
   (b) A file named ***workingZone*** (type: Executable file)
   (c) This assignment file (type: PDF file)
   (d) One template text file (TemplateDataSubmission.txt) that you need to fill up with your data.

   Report immediately if you do not find all of these files. Executable files ***checkZone*** and ***workingZone*** have been generated as follows from a main.c file (main.c code is not given to you).

   gcc -Wall -g main.c -o checkZone

   gcc -Wall -pg main.c -o workingZone

4. Write a C program (name it <your roll no>.c) that takes a positive integer as input and will do this. It will first generate and store all its factors in a list (array). If any factor is greater than 1000 (excluding the number itself), re-factor it and append its factors to the array. If the factor >1000 is prime then add one with it and then re-factor it, and append to your list. The program should print the final list excluding the number itself. Run the program with the last 5 digits of your roll no. as input. Note down the list printed somewhere.

   For example,
   (i) If your roll number is 23CS10001, then input to your C program will be 10001. It will print following factors:

   1, 73, 137

   (ii) If your roll number is 23CS10003, then input to your C program will be 10003. It will print following factors:

   1, 7, 2, 5, 10, 11, 13, 22, 26, 55, 65, 110, 130, 143, 286, 715, 1430

   (factors of 10003 are 1, 7, 1429, 10003, of which 1429 > 1000 is prime so we re-factor (1429+1) and append to the list)

5. The pseudo-code of the main() function in main.c, along with their actual line numbers in the file, is given below. Use this information in conjunction with appropriate executable files among the two given and

tools you know to determine which factors (exclude single digit numbers) are in the working zone. A factor is in the ***working zone*** if when input to the main function below in the scanf in line 75, it will generate a value of ***n*** (in line number 76) greater than 20 but less than 50. Please note that line number 80 onwards the program contains multiple functions with exponential time complexity. If the factor is not in the working zone then the program may end early or take long time to end.

```
-------------------------------------------------------------------------------------------
71 int main()
 72 {
 73        int n,m,num;
 74        printf("Enter your factor: ");
 75        scanf("%d",&num);
 76        n=func0(num);
 77            …..
 78            …..
 79            …..
 80            …..
 81            …..
 82        (n%2==1)? workZone1(n):workZone2(n);
 83        return 0;
 84 }
---------------------------------------------------------------------------------------------
```

6. Next, select last two factors that lie in the ***working zone*** to profile the flat file and call graphs.

7. Redirect only the flat file and the call graph (without any redundant explanation about them) corresponding to the factor and name the file as callg<factor value>.txt. For example, if you used the factor value 31, the file should be named callg31.txt. Use the appropriate executable file and other related commands that you need. Repeat the step for the other factor selected in the previous step.

8. Populate the TemplateDataSubmission.txt file with the call graph information.

9. Inside your current directory, create a subdirectory with the naming convention ***A3<your roll no>***. Copy your C code (as you have written in step 4), TemplateDataSubmission.txt file as you populated with relevant information, and the two callg*.txt files for the two factors generated in step 7. **Zip this subdirectory and upload in the moodle submission server under Assignment 3.**

Please note the executable file contains multiple functions whose complexity is exponential. If you input any value that does not belong to working zone, the computer system may hang/crash/shows some abnormalities.

Example Run:

```
==================Valid Run for factor 14========================
./checkZone
Enter your factor: 14
Func1: 8
Func2: 65535

/workingZone
Enter your factor: 14
Func1: 8
Func2: 65535
```