CSE344
SYSTEM PROGRAMMING

HOMEWORK 1 REPORT

Name : Göktuğ Ali
Surname : Akın
Student ID : 161044018

Representing complex number :

To represent a complex number, my approach is use basic struct named complex_num. By this way, manipulation of complex number is easy.

These structure defined in “complex.h” and implemented in “complex.c”. More detailed expression can be found in the header file function by function.

Synchronization of files

I used **<fcntl>** library to synchronize the processes through the files. Instance of programA locks the file output file by passing it to a *F_WRLCK*. By this way, other processes cannot write until programA finishes its file operation. So processes do not overwrite the files, they waits their orders. After program finishes its job on the file, it pass the fnctl method to *F_UNLCK* signal to unlock the file.

ProgramB also use the same file with the programA. ProgramB does the same locking mechanism, it locks before it operates and after workings of file done, it unlocks the file with the same way. I explained theese steps in my code step by step with comments.

Two instances of programB also use the same file for output. Same control mechanism used in this part too.

Interprocess communication between process (programA and programB)

Problem :

ProgramB should know that, if there is alive process (programA) or not. Because, if there is no processes that is programA, programB should not wait incoming inputs, since these inputs are written by programA. But if there is at least one programA running and programB sees that input file is empty, it should not terminate because there is one programA running.

Solution approach :

I did connection between processes (programA and programB) between simple txt file named “ipc.txt”. This file contains 4 bytes for communication. **<fcntl>** library used for this approach to lock or unlock the IPC file.

Example file format of ipc.txt file :

A1
B2

It means that, there are 1 programA running, and there are 2 programB running now. And **file_status** struct is used for represent the file status easily. Implementation can be found in “utils.c” file.

If A is 0 and B is 1 for example, programB will stop when he sees the A0 data in the ipc file.

If programB is started before programA, ipc file will start in this format ; After init_programB() function called ;

A/
B1

which means, there is no programA started yet, and it will wait for the running process of A.

Any other details of communication between process through ipc file can be found in the comments.

Calculating the Discrete Fourier Transform

FFT implementation separated into different file. Header file of FFT is “fft.h” and implementation is in the “fft.c” file.

Sources that I used to implement the FFT algorithm ;

- https://stackoverflow.com/questions/8801158/fft-in-a-single-c-file
- https://www.youtube.com/watch?v=htCj9exbGo0