Version Saturday, November 4th 2017

Week 07

Deadline : Saturday, November 11th 2017 15:00

Synchronization

1. In this tutorial, we will learning about synchronization with some C program
2. Login to your badak account
3. Change your directory to “work” and create new directory named “work07” inside “work” directory

```
# cd work
# mkdir work07
```

4. Download the tarball file required for this task on scele, and **move** it to your work07 directory.

   The tarball file named `lab-materials.tbj`.

5. Change your directory to `work07` directory
6. Un-tar the tarball file named `lab-materials.tbj`

```
$ tar xvfj lab-materials.tbj
```

7. **Move** all the files inside folder `lab-materials` **to** your work07 directory.

   Example of the command:

```
$ mv [file_name] ../[file_name]
```

   *The files you need to move:

(1) 99-myutils.h

(2) 99-myutils.c

(3) 01-thread.c

(4) 03-readwrite.c

(5) 05-balap.c

(6) 50-readwrite.c

(7) Makefile

8. Remove the folder and the tarball file.

```
$ rm -rf lab-materials
$ rm -rf lab-materials.tbj
```

9. Make file lab07.txt

```
$ vi lab07.txt
```

10. Fill the first row of file lab07 with your github username like

```
# Github Account: myusername
```

    Don’t forget to save the file and exit the text editor

11. Compile the files in the `work07` directory

```
$ make
```

Learning Synchronization

1. Run the `01-thread` and take a look for the output

```
# ./01-thread
THREAD2: I am first!
THREAD1: I am second!
THREAD3: I am last!
Bye Bye Main...
```

2. If you see the source code (`01-thread.c`), even though the threads registered by sequence `thread1-thread2-thread3`, the output always by sequence `thread2-thread1-thread3`
3. It occurred because we using `semaphore` to control the program sequential execution.
4. In this program, we use two semaphores (`generik` and `generik2`) to do that.
5. We made the `thread3` to wait the `generik2` semaphore that given by `thread1` that waiting the `generik` semaphore that given by `thread2`. By doing this, we make sure that `thread3` is executed after `thread1`, and `thread1` executed after `thread2`.
6. Now, run the `03-readwrite` and `05-balap` program.

```
# ./[program_name]
```

7. Understand the output by looking at `03-readwrite.c` and `05-balap.c`

```
# vi [file_name].c
```

8. Write on `lab07.txt` that contains **for each program** (`03-readwrite and 05-balap`):

- brief summary of the program.
- critical sections.
- purpose of using mutex and the outcome if not using mutex.

  *you can also see 99-myutils.c and 99-myutils.h to understand the program.

  *warning : write the answer by your own, plagiarism rule applies.

Exercise

1. Modify `50-readwrite.c` so that **all writers must write once before the readers can read it and the writer only can write again after all readers read.**
2. Program will accept input with format “<symbol><characacter>”. Symbol is consist of either ‘*’ and ‘#’. ‘*’ denotes that the character will be written in lowercase and ‘#’ denotes that the character will be written in uppercase. As an example, input “#f*a*s*i*l*k*o*m” will be written as “Fasilkom”. Please note that all the characters in input is lowercase.
3. Each reader will be read two characters from the input alternately. If all the characters has been read then next reader will read input from the beginning of the string.
4. Each writer will be write one character alternately. If all character has been write then the next writer will write string from the beginning. Please note that if reader has not read any character then writer will output “=TIDAK==NULIS”.
5. Example: if there are 2 writers and 3 readers

```
Enter input : #o#s*e*z
                        READER 0: SIAP  ******
                        READER 1: SIAP  ******
                        READER 2: SIAP  ******
WRITER 0: SIAP  *******
WRITER 0: REHAT *******
WRITER 1: SIAP  *******
WRITER 1: REHAT *******
WRITER 0: MAU   MENULIS
WRITER 0:=TIDAK==NULIS
WRITER 1: MAU   MENULIS
WRITER 0: SELESAI NULIS
WRITER 1:=TIDAK==NULIS
WRITER 1: SELESAI NULIS
                        READER 0: REHAT ******
                        READER 2: REHAT ******
                        READER 1: REHAT ******
                        READER 1: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 1:=SEDANG==BACA [#o]
                        READER 2: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 2
                        READER 2:=SEDANG==BACA [#s]
                        READER 2: SELESAI BACA
                        ***** SISA PEMBACA 1
                        READER 1: SELESAI BACA
                        ***** SISA PEMBACA 0
                        READER 0: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 0:=SEDANG==BACA [*e]
                        READER 0: SELESAI BACA
                        ***** SISA PEMBACA 0
WRITER 1: REHAT *******
WRITER 0: REHAT *******
WRITER 0: MAU   MENULIS
WRITER 0:=SEDANG==NULIS [O]
WRITER 1: MAU   MENULIS
WRITER 0: SELESAI NULIS
WRITER 1:=SEDANG==NULIS [S]
WRITER 1: SELESAI NULIS
                        READER 0: REHAT ******
                        READER 2: REHAT ******
                        READER 1: REHAT ******
                        READER 1: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 1:=SEDANG==BACA [*z]
                        READER 0: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 2
                        READER 0:=SEDANG==BACA [#o]
                        READER 1: SELESAI BACA
                        ***** SISA PEMBACA 1
                        READER 0: SELESAI BACA
                        ***** SISA PEMBACA 0
                        READER 2: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 2:=SEDANG==BACA [#s]
                        READER 2: SELESAI BACA
                        ***** SISA PEMBACA 0
WRITER 0: REHAT *******
WRITER 1: REHAT *******
WRITER 0: MAU   MENULIS
WRITER 0:=SEDANG==NULIS [e]
WRITER 0: SELESAI NULIS
WRITER 1: MAU   MENULIS
WRITER 1:=SEDANG==NULIS [z]
WRITER 1: SELESAI NULIS
                        READER 2: REHAT ******
                        READER 0: REHAT ******
                        READER 1: REHAT ******
                        READER 2: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 1
                        READER 2:=SEDANG==BACA [*e]
                        READER 0: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 2
                        READER 0:=SEDANG==BACA [*z]
                        READER 1: MAU  MEMBACA
                        ***** JUMLAH PEMBACA 3
                        READER 1:=SEDANG==BACA [#o]
                        READER 2: SELESAI BACA
                        ***** SISA PEMBACA 2
                        READER 0: SELESAI BACA
                        ***** SISA PEMBACA 1
                        READER 1: SELESAI BACA
                        ***** SISA PEMBACA 0
WRITER 0: REHAT *******
WRITER 1: REHAT *******
WRITER 0: MAU   MENULIS
WRITER 0:=SEDANG==NULIS [O]
WRITER 1: MAU   MENULIS
WRITER 0: SELESAI NULIS
WRITER 1:=SEDANG==NULIS [S]
WRITER 1: SELESAI NULIS
        …
        …
```

hint: one of the ways to do this is that you can make the readers waiting for signal from last writer and vice versa. You can also create some counter to check whether all writers/readers have finished writing/reading.

note: You may only add some line of code to do this task, do not edit or delete the code that already exist in 50-readwrite.c, except for the code that stated to be modified.

6. Don't forget to compile the files in the `work07` directory after change the program

```
$ make
```

Privacy Matters, Encryption and Digital Signature using GnuPG

1. Hash and sign your works so the other know it truly your works

```
$ sha1sum * > SHA1SUM
$ sha1sum -c SHA1SUM
$ gpg --sign --armor --detach SHA1SUM
```

2. verify the works

```
$ gpg —-verify SHA1SUM.asc
```

3. Change your directory to "work", and create a tar ball. Tar is a way to create an archive file (like zip). You can google it for more information

```
$ cd ..
$ tar cvfj work07.tbj work07/
```

4. encrypt the tar file

```
$ gpg --output work07.tbj.gpg --encrypt --recipient OSTEAM work07.tbj
```

5. Copy the file to your os172 directory in folder week07/

```
$ cp work07.tbj.gpg ~/os172/week07/work07.tbj.gpg
```

6. change your directory to os172/week07/

7. remove file named `"dummy"`
8. Check also whether your copy of “work07.tbj.gpg” exists or not. If you don’t find it, copy it once again
9. push the change to GitHub server
10. Week07 is done.

Review your Work

After this week task completed, please don’t forget to check your files/folders. The structure of files/folders should be like:

```
os172
      key
            mypublickey1.txt
      log
            log01.txt
            log02.txt
            log03.txt
            log04.txt
            log05.txt
            log06.txt
            log07.txt
      SandBox
            <some_random_name>
      week00
            report.txt
      week01
            lab01.txt
            report.txt
            whyStudyOS.txt
            what-time-script.sh
      week02
            work02.tbj.gpg
                  *work02
                        *00-toc.txt
                        *01-public-osteam.txt
                        *02-ls-al.txt
                        *03-list-keys1.txt
                        *04-list-keys2.txt
```

```
                    *hello.c
                    *hello
                    *status.c
                    *status
                    *loop.c
                    *loop
                    *exercise.c
                    *exercise
                    *SHA1SUM
                    *SHA1SUM.asc
week03
      work03.tbj.gpg
            *work03
                    *.profile
                    *sudo-explanation.txt
                    *what-is-boot.txt
                    *SHA1SUM
                    *SHA1SUM.asc
week04
      work04.tbj.gpg
            *work04
                    *01-public-osteam.txt
                    *lab04.txt
                    *global-char.c
                    *global-char
                    *local-char.c
                    *local-char
                    *open-close.c
                    *open-close
                    *write.c
                    *write
                    *result1.txt
                    *result2.txt
                    *demo-file1.txt

                    *demo-file2.txt

                    *demo-file3.txt

                    *demo-file5.txt

                    *00-pointer-basic.c

                    *00-step-1

                    *00-step-2

                    *00-step-3

                    *00-step-4

                    *SHA1SUM

                    *SHA1SUM.asc

week05

      Work05.tbj.gpg

            *vm-to-memory.c

            *vm-to-memory

week06
      work06.tbj.gpg
            *04-fork.c
            *04-fork
            *05-fork.c
            *05-fork
            *06-fork.c
            *06-fork
            *10-fork.c
            *10-fork
            *11-fork.c
            *11-fork
            *cascafork.c
            *cascafork
            *cascafork2.c
            *cascafork2
            *Makefile
            *result.txt
            *lab06.txt
            *SHA1SUM
            *SHA1SUM.asc
```

week07

```
      work07.tbj.gpg
```

```
        *work07
              *01-thread
              *01-thread.c
              *01-thread.o
              *03-readwrite
              *03-readwrite.c
              *03-readwrite.o
              *05-balap
              *05-balap.c
              *05-balap.o
              *50-readwrite
              *50-readwrite.c
              *50-readwrite.o
              *99-myutils.h
              *99-myutils.c
              *99-myutils.o
              *Makefile
              *lab07.txt
              *SHA1SUM
              *SHA1SUM.asc
  week08
        dummy
  week09
        dummy
  week10
        dummy
  xtra
        dummy
```

keep in mind for every files/directories with wrong name/content, you will get penalty point.

note: "" means file that should be inside the archived file.*