Week 08

Deadline: Saturday, November 18th 2017 at 3pm

Preparation before lab

1. In this lab, you will learn One-Level and Two-Level Scheduling with First Come First Serve (FCFS) and Round Robin in C program.
2. Login to your badak account.
3. Change your directory to “work” and create new directory named “work08” inside “work” directory.

```
$ cd work
$ mkdir work08
```

4. Move all the file attached at Scele to your work08 directory. Hint: use WinSCP or Tunnels. The files you need to move:
   (1) one-level.c
   (2) two-level.c
5. Change your directory to work08.

```
$ cd work08
```

Introduction to One-Level Scheduling with First Come and First Serve

6. You are given a program **one-level.c** contain first come first serve algorithm. As you can see, the first process that arrived will be executed compared with the later process. Example: There are three processes, where the sequence of the processes is Process 1, Process 2, and Process 3 with the following information:

Process	Burst Time
P1	4
P2	9
P3	3

Because the P1 was executed first, so the waiting time is 4, after that P2 with 13 and the last executed process was P2 with 16 as it waiting time.

Compile **one-level.c** with gcc.

```
$ gcc one-level.c -o one-level
```

7. Try to run the program with the test case at the previous step:

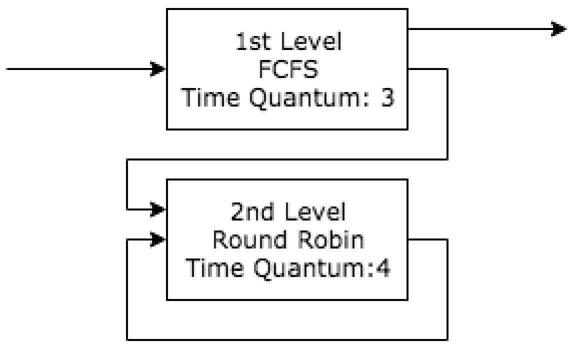
```
$ ./one-level
Enter Total Process: 3
Enter Burst Time for Process Number 1: 4
Enter Burst Time for Process Number 2: 9
Enter Burst Time for Process Number 3: 3
```

Analyze the output of the program.

Introduction to Two-Level Scheduling with First Come First Serve and Round Robin

8. The next algorithm is Round Robin. In Round Robin, there is a thing called time quantum. It's a time that will limit how long the process will be executed. If a process have executed past the time quantum, it will be returned to the last queue of processes and the job will be paused. The next process that will be executed is the one at the front of queue. The unfinished process will resume when it arrives in front of the queue and so on until the whole process is done.

   Your task is to create a program with two-level scheduling. The first level is implemented by first come first serve algorithm with 3 as the time quantum's value. The second level will be implemented by Round Robin algorithm with 4 as the time quantum's value. If there is an unfinished process in the first level, it will be thrown to the second level. Here's the illustration of the two-level.



Implement the Round Robin algorithm within **two-level.c** file in the space provided. The output is total of waiting time that each process has. Waiting time is time that is take for a process until it's done.

9. Here are some test cases that could help you out:
   a. **Input**

```
Enter Total Process: 2
Enter Arrival Time for Process Number 1: 0
Enter Burst Time for Process Number 1: 4
Enter Arrival Time for Process Number 2: 1
Enter Burst Time for Process Number 2: 3
```

Output:

```
P[1] : 7
P[2] : 5
```

   b. **Input**

```
Enter Total Process: 3
Enter Arrival Time for Process Number 1: 0
Enter Burst Time for Process Number 1: 9
Enter Arrival Time for Process Number 2: 1
Enter Burst Time for Process Number 2: 3
Enter Arrival Time for Process Number 3: 2
Enter Burst Time for Process Number 3: 8
```

Output:

```
P[1] : 19
P[2] : 5
P[3] : 18
```

Privacy Matters, Encryption and Digital Signature using GnuPG

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

```
$ sha1sum * > SHA1SUM

$ sha1sum -c SHA1SUM

$ gpg --sign --armor --detach SHA1SUM
```

11. Verify the works.

```
$ gpg --verify SHA1SUM.asc
```

12. Create a tar ball. Tar is a way to create an archive file. You can ask uncle G for more information.

```
$ cd ..

$ tar cvfj work08.tbj work08/
```

13. Encrypt your files (work08.tbj).

```
$ gpg --output work08.tbj.gpg --encrypt --recipient
OSTEAM --recipient your@email.com work08.tbj
```

*Use the same email as your Email input on GnuPG key generator.

14. Copy the file to your github account, under the file week08/

```
$ cp work08.tbj.gpg ~/os172/week08/work08.tbj.gpg
```

15. Change your directory to **os172/week08/**
16. Remove file named ”**dummy**”.
17. Check whether there is a file named ”**work08.tbj.gpg**” if you dont find it, do the copy once more.
18. Push the change to GitHub server.
19. Done!

Review Your Work

Dont forget to check your files/folders. After this lab, your current os172 folder should looks like:

```
os172
        key
                mypublickey1.txt
        log
                log01.txt
                log02.txt
                log03.txt
                log04.txt
                log05.txt
                log06.txt
                log08.txt
                log09.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.c
                        *99-myutils.o
                        *Makefile
                        *lab07.txt
                        *SHA1SUM
                        *SHA1SUM.asc
week08
      work08.tbj.gpg
            *work08
                  *one-level
                  *one-level.c
                  *two-level
                  *two-level.c
                  *SHA1SUM
                  *SHA1SUM.asc
week09
      dummy
week10
      dummy
xtra
      dummy
```

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

*means file that should be inside the archived file.

Note: "" means file should be inside the archived file.*