Gebze Techinal University
Computer Engineering
CSE344 – Spring 2021
HW4

Mustafa Gurler
171044034

1.Problem Definition

It is a thread model that one supplier multi consumer. There are N types consumer which has been taken by from command line. There are main thread, detach thread and consumer threads. Consumer threads takes the 1 and 2 from supplier that produces one by one.

2.Solution

In main thread detach thread and Consumer threads have been created. Also two semaphores have been created with system 5 semaphore. Semaphores are used for controlling the 1 and 2. They are providing to a protection for operation of producer and consumers.

2.1.Design and Solutions

2.1.1 Detach Thread

I created detach thread with attribute of detach flag. It does not have to wait a join inside of main thread.

```
pthread_t tid_thread [C], tid_supplier;
pthread_attr_t attr;
int s;
if(sigtermInterrupt){
    write(1, "KILLED",strlen("KILLED"));
    semctl(sem_id, 1, IPC_RMID);
    exit(EXIT_SUCCESS);
}
s = pthread_attr_init(&attr);
if(sigtermInterrupt){
    write(1, "KILLED",strlen("KILLED"));
    semctl(sem_id, 1, IPC_RMID);
    exit(EXIT_SUCCESS);
}
if(s != 0){
    perror("pthread_attr_init");
    exit(EXIT_FAILURE);
}
if(sigtermInterrupt){
    write(1, "KILLED",strlen("KILLED"));
    semctl(sem_id, 1, IPC_RMID);
    exit(EXIT_SUCCESS);
}

s = pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_DETACHED);
if(s != 0){
    perror("pthread_attr_setdetachstate");
    exit(EXIT_FAILURE);
}
int thread_no[C];
int r;

if(sigtermInterrupt){
    write(1, "KILLED",strlen("KILLED"));
    semctl(sem_id, 1, IPC_RMID);
    exit(EXIT_SUCCESS);
}

//Create supplier
if((r = pthread_create(&tid_supplier, &attr ,supplier, NULL))  != 0){
    write(1, "Creating supplier thread failed",
                        sizeof("Creating supplier thread failed"));
    exit(EXIT_FAILURE);
}
```

2.1.2 Consumer Thread

Consumer threads have been created inside of a loop and ids has been forwarded via pthread_create fourth argument.

```
for(int i=0; i < C ; i++){
    thread_no[i] = i;
    if(sigtermInterrupt){
        write(1, "KILLED",strlen("KILLED"));
        semctl(sem_id, 1, IPC_RMID);
        exit(EXIT_SUCCESS);
    }
    if((r = pthread_create(&tid_thread[i],NULL, consumer, (void*) &thread_no[i])) !=0 ){
        write(1, "Creating consumer thread failed",
                        sizeof("Creating consumer thread failed"));
        exit(EXIT_FAILURE);
    }
    if(sigtermInterrupt){
        write(1, "KILLED",strlen("KILLED"));
        semctl(sem_id, 1, IPC_RMID);
        exit(EXIT_SUCCESS);
    }
}
```

2.1.3 Joining Thread

Threads are default needs to be joined. I did not declare any flag and it needs to be joined.

```
for(int i=0 ; i < C ; i++){
    if(sigtermInterrupt){
        write(1, "KILLED",strlen("KILLED"));
        semctl(sem_id, 1, IPC_RMID);
        exit(EXIT_SUCCESS);
    }
    if((r = pthread_join(tid_thread[i], NULL)) != 0){
        write(1, "Joining thread failed",
                            sizeof("Joining thread failed"));
        exit(EXIT_FAILURE);
    }
    if(sigtermInterrupt){
        write(1, "KILLED",strlen("KILLED"));
        semctl(sem_id, 1, IPC_RMID);
        exit(EXIT_SUCCESS);
    }
}
```

2.1.4 Detach Thread Function

Supplier needs to get the character by character from input file and post the semaphore. That is it. But also it needs to control which character has arrived.

2.1.5 Consumer Thread Function

It waits 1 and 2 at the same time from detach thread. It gets the value from semctl GETVALUE and prints out after that it operates -1 and again prints out the after values of semaphores.

2.1.6 SIGINT Signal

It controls all the possible signal and close the semaphores. You can control it via writing “ipcs -s”.

3. Which requirements I achieved?

I achieved all requirement from assignment file.