GEBZE TECHNICAL UNIVERSITY

CSE344
SYSTEM PROGRAMMING COURSE

HOMEWORK 4
REPORT

BARAN SOLMAZ
1801042601

Problem Defination:

- One supplier thread and multiple consumer threads,
- Supplier thread must be detached,
- Supplier reads inputfile one character at a time and posts that characters semaphore by one,
- Consumers waits for two semaphores at the same time,
- Using only System V Semaphores,
- No more than 2 semaphores.

Problem Solution Approach:

System V Semaphores:

Creating:

```
void create_sems(){
    key = ftok(FTOK_NAME, 1);
    if (key == -1)
        perror_call("ftok");

    semid = semget(key, 2, IPC_CREAT | IPC_EXCL | S_IRUSR | S_IWUSR);
    if (semid == -1)
        perror_call("semget");
}
```

2 Semaphore,
One for ‘1’, the other is for ‘2’

Initializing:

```
void initialize_sems(){
    union semun arg;
    unsigned short arr[2] = {0, 0};
    arg.array = arr;
    if (semctl(semid, 0, SETALL, arg) == -1)
        perror_call("semctl-initialize");
}
```

Removing:

```
void remove_sems(){
    if (semctl(semid, 0, IPC_RMID) == -1)
        perror_call("semctl-remove");
}
```

Post/Wait:

```
void semPost(int sem_index){
    struct sembuf buf={0};
    buf.sem_num=sem_index;
    buf.sem_op=1;
    buf.sem_flg=0;
    semop(semid,&buf,1);
}
```

```
void semWait_All(){
    struct sembuf buf[2] = {{0},{0}};
    for (int i = 0; i < 2; i++){
        buf[i].sem_num = i;
        buf[i].sem_op =-1;
        buf[i].sem_flg = 0;
    }
    semop(semid,buf,2);
}
```

- semPost is for Supplier, semWait_All is for Consumers,
- No need for semWait for only one semaphore,because it will not be used.

Threads:

```
void create_threads(int *arr,pthread_t *sup,pthread_t *threads){
    for (int i = 0; i < consumersNumber; i++){
        arr[i]=i;
        void *p=&arr[i];
        if (pthread_create(&threads[i], NULL, consumer, p) != 0)
            perror_call("pthread_create");
    }
    if (pthread_create(sup, NULL, supplier, NULL) != 0)
        perror_call("pthread_create");

    if (pthread_detach(*sup) != 0) To make Supplier thread detached
        perror_call("pthread_detach");
}
```

```
void join_threads(pthread_t *threads){
    for (int i = 0; i < consumersNumber; i++)
        if (pthread_join(threads[i], NULL) != 0)
            perror_call("pthread_join");
}
```

Only waiting for consumer threads.

Supplier:

```
void* supplier(){
    while (1){
        char buffer[3];
        memset(buffer, '\0', 3);
        if(sig_check_supplier()==1)
            break;
        if (read(inputFd, buffer, 1) != 1)//EOF
            break;

        if (sig_check_supplier() == 1)
            break;
        printf("%s Supplier: read from input a '%c'. Current amounts: %d x '1', %d x '2'.\n",
        switch (buffer[0]){
            case '1':
                semPost(0);
                break;
            case '2':
                semPost(1);
                break;
            default:
                break;
        }
        printf("%s Supplier: delivered a '%c'. Post-delivery amounts: %d x '1', %d x '2'.\n",
    }
    printf("%s The Supplier has left.\n", get_timestamp());
    return NULL;
}
```

Consumer:

```c
void *consumer(void* in){
    int id = *((int*)in);
    for (int i = 0; i <loopTime; i++){
        if (sig_check_consumer() == 1)
            break;
        printf("%s Consumer-%d at iteration %d (waiting).Current amounts: %d x '1', %d x '2'.\n",
              get_timestamp(), id, i, semctl(semid, 0, GETVAL, NULL),semctl(semid, 1, GETVAL, NULL));
        if (sig_check_consumer() == 1)
            break;
        semWait_All();

        if (sig_check_consumer() == 1)
            break;
        printf("%s Consumer-%d at iteration %d (consumed).Post-consumption amounts: %d x '1', %d x '2'.\n",
              get_timestamp(), id, i, semctl(semid, 0, GETVAL, NULL),semctl(semid, 1, GETVAL, NULL));
    }

    printf("%s Consumer-%d has left.\n", get_timestamp(), id);
    return NULL;
}
```

Tests:

Inputs:

```
./hw4 -C 5 -N 2 -F input5x2.txt
```

5 Consumer 2 Loop
10 ‘1’ 10 ‘2’

Outputs:

```
./hw4 -C 5 -N 2 -F input5x2.txt
Wed May 11 19:56:42 2022  Consumer-0 at iteration 0 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-3 at iteration 0 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-4 at iteration 0 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-1 at iteration 0 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-2 at iteration 0 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-3 at iteration 0 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-3 at iteration 1 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 1 x '2'.
```

```
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-2 at iteration 0 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-1 at iteration 0 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-1 at iteration 1 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-2 at iteration 1 (waiting).Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-3 at iteration 1 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-3 has left.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 0 x '1', 0 x '2'.
```

```
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 1 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 1 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Consumer-1 at iteration 1 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-1 has left.
Wed May 11 19:56:42 2022  Supplier: read from input a '1'. Current amounts: 1 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '1'. Post-delivery amounts: 2 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: read from input a '2'. Current amounts: 2 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Supplier: delivered a '2'. Post-delivery amounts: 2 x '1', 2 x '2'.
Wed May 11 19:56:42 2022  The Supplier has left.
Wed May 11 19:56:42 2022  Consumer-0 at iteration 0 (consumed).Post-consumption amounts: 2 x '1', 2 x '2'.
Wed May 11 19:56:42 2022  Consumer-0 at iteration 1 (waiting).Current amounts: 2 x '1', 2 x '2'.
Wed May 11 19:56:42 2022  Consumer-0 at iteration 1 (consumed).Post-consumption amounts: 1 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Consumer-0 has left.
Wed May 11 19:56:42 2022  Consumer-4 at iteration 0 (consumed).Post-consumption amounts: 2 x '1', 2 x '2'.
Wed May 11 19:56:42 2022  Consumer-4 at iteration 1 (waiting).Current amounts: 1 x '1', 1 x '2'.
Wed May 11 19:56:42 2022  Consumer-4 at iteration 1 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-4 has left.
Wed May 11 19:56:42 2022  Consumer-2 at iteration 1 (consumed).Post-consumption amounts: 0 x '1', 0 x '2'.
Wed May 11 19:56:42 2022  Consumer-2 has left.
```