코드 이미지 및 설명

(주석으로 작성)

Server

```c
mkfifo(FIFO_PATH, 0666); // server니까 FIFO만들기
fd = open(FIFO_PATH, O_RDWR); // 읽기 쓰기 모두 가능하게 옵션 지정

/* open semaphore */
sem_unlink("mysema");
if ((p_sem = sem_open("mysema", O_CREAT, 0600, 0)) == SEM_FAILED) // 세마포어 생성
{
  perror("sem_open");
  exit(1);
}

/* wait for client to join.
 * the client will call sem_post when starting */
sem_wait(p_sem);

/* game loop */
do {
    printf("[server] waiting for semaphore.. turn: %d\n", turn); // 로그 남기기
    sem_wait(p_sem);

    if (turn != 0) { // turn이 다 될때까지 반복
        printf("[server] reading data from buffer turn: %d\n", turn);
        /* read from fifo (opponet) */
        len = read(fd, buf, BUF_SIZE - 1);
        buf[len] = '\0'; /* assure null-termination of string */
        printf("[opponet] %s\n", buf);
    }

    printf("Your turn!\n");
    /* get user input */
    fgets(buf, BUF_SIZE, stdin);
    if (strcmp(buf, msg) != 0) {
      score -= 20;
      printf("wrong! -20\n");
    }
    write(fd, buf, strlen(buf));
    turn++;
    sem_post(p_sem);
    usleep(100);
} while (turn < 5);
```

Client

```c
  /* sem_unlink("mysema"); client shouldn't unlink semaphore */
  /* open semaphore */
  if ((p_sem = sem_open("mysema", O_CREAT, 0600, 0)) == SEM_FAILED) // 세마포어 생성
  {
    perror("sem_open");
    exit(1);
  }

  /* make server start game! */
  sem_post(p_sem);
  sem_post(p_sem);

  usleep(1000);
  /* game loop */
  do {
      printf("[client] waiting for semaphore.. turn: %d\n", turn);
      sem_wait(p_sem); // 세마포어 Lock

      printf("[client] acquired! turn: %d\n", turn);
      // 로그를 남겨서 몇번째 turn인지 확인

      /* read from fifo (opponet) */
      len = read(fd, buf, BUF_SIZE - 1);
      buf[len] = '\0'; /* assure null-termination of string */
      printf("[opponet] %s\n", buf);

      printf("Your turn!\n");
      /* get user input */
      fgets(buf, BUF_SIZE, stdin);
      if (strcmp(buf, msg) != 0) { // 입력이 잘못되면
        score -= 20;
        printf("wrong! -20\n");
      }

      write(fd, buf, strlen(buf));
      turn++;
      sem_post(p_sem); // 세마포이 unlock
      usleep(1000); // 기다려줘야지 세마포어가 혼자 입력받고 쓰고를 안함
      printf("[client] sem_post! turn: %d\n", turn); // 로그를 남겨서 디버깅함
  } while (turn < 5);

  printf("Done! Your score: %d\n", score);
  return 0;
}
```

예상결과 및 실제 결과 화면

Server & client (예상결과와 같았음)

```
u201600253@sejung-VirtualBox:~/Desktop/week08$ ./pp_server
[server] waiting for semaphore.. turn: 0
Your turn!
ping
[server] waiting for semaphore.. turn: 1
[server] reading data from buffer turn: 1
[opponet] ping

Your turn!
babo
wrong! -20
[server] waiting for semaphore.. turn: 2
[server] reading data from buffer turn: 2
[opponet] pong

Your turn!
pong
wrong! -20
[server] waiting for semaphore.. turn: 3
[server] reading data from buffer turn: 3
[opponet] pong

Your turn!
sejung
wrong! -20
[server] waiting for semaphore.. turn: 4
[server] reading data from buffer turn: 4
[opponet] pong

Your turn!
ping
Done! Your score: 40
```

```
u201600253@sejung-VirtualBox:~/Desktop/week08$ ./pp_client
[client] waiting for semaphore.. turn: 0
[client] acquired! turn: 0
[opponet] ping

Your turn!
ping
wrong! -20
[client] sem_post! turn: 1
[client] waiting for semaphore.. turn: 1
[client] acquired! turn: 1
[opponet] babo

Your turn!
pong
[client] sem_post! turn: 2
[client] waiting for semaphore.. turn: 2
[client] acquired! turn: 2
[opponet] pong

Your turn!
pong
[client] sem_post! turn: 3
[client] waiting for semaphore.. turn: 3
[client] acquired! turn: 3
[opponet] sejung

Your turn!
pong
[client] sem_post! turn: 4
[client] waiting for semaphore.. turn: 4
[client] acquired! turn: 4
[opponet] ping

Your turn!
ya
wrong! -20
[client] sem_post! turn: 5
Done! Your score: 60
```

느낀 점
너무 어려웠다. 사실 이해를 완벽하게 한 것 같지는 않다. 처음 post이후에 usleep을 해주지 않았는데 이 때문에 세마포어가 스스로 쓴 것을 fifo에서 읽어오는 상황이 발생했는데, usleep을 통해서 해결했다. 이유는 아직 잘 모르겠다.