Project – Final report

게스트하우스

과 목	임베디드시스템S/W설계
담당교수	김태석 교수님
학 과	컴퓨터정보공학부
학 번	2017202087
성 명	홍세정
날 짜	2021. 05. 25 (화)

광운대학교 KwangWoon University

1. 요약

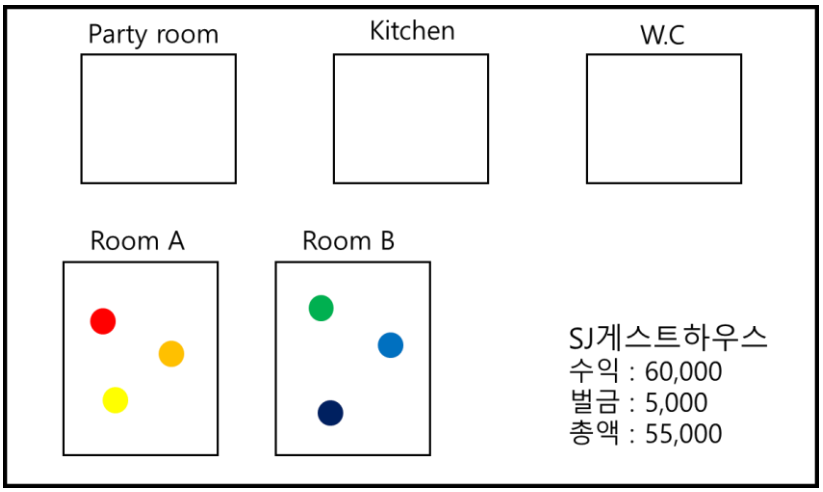
uCoS를 이용하여 임베디드 응용프로그램을 구현한다. 구현하고자 하는 프로그램은 게스트하우스 시스템이다.

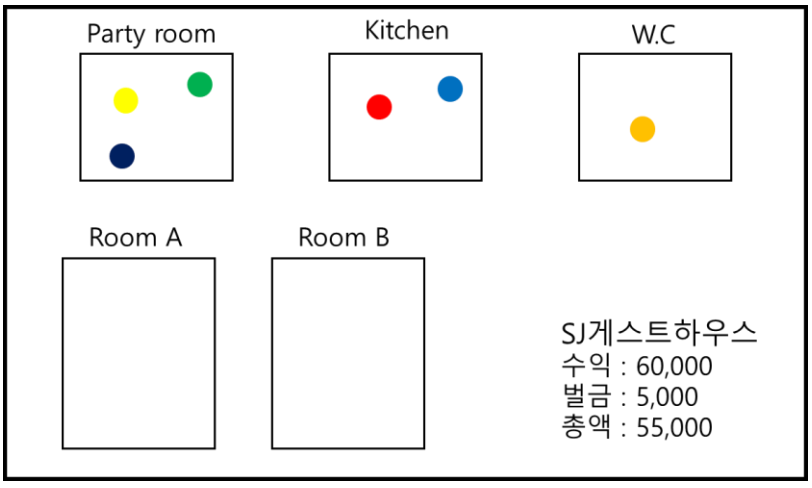
게스트 하우스에는 방이 총 2개가 존재하고 각 방에는 3명이 사용할 수 있다. 만실일 때는 총 6명의 게스트가 숙박하게 된다. 게스트 하우스 입실한 게스트들은 공용 시설인 파티룸, 주방, 화장실을 사용할 수 있다. 하지만 공용 시설들은 파티룸: 6명, 주방: 4명, 화장실:1명 들어갈 수 있는 인원에 제한이 있다. 게스트들은 공용시설들을 사용할 수 있지만 인원이 모두 차있다면 사용하지 못하고 기다려야한다. 이때 게스트들은 체크인을 한 순서대로 움직이게 된다. 또한 체크인할 때 숙박비 10000씩 내야한다.

프로젝트는 다음과 같은 요구사항을 가지고 있다. 게스트들이 체크인을 할 수 있어야한다. 요금 지불, 게스트 공용 시설 이동, 이동을 마쳤다면 체크 아웃을 해야 한다.

2. 본문

A) 프로젝트 개요



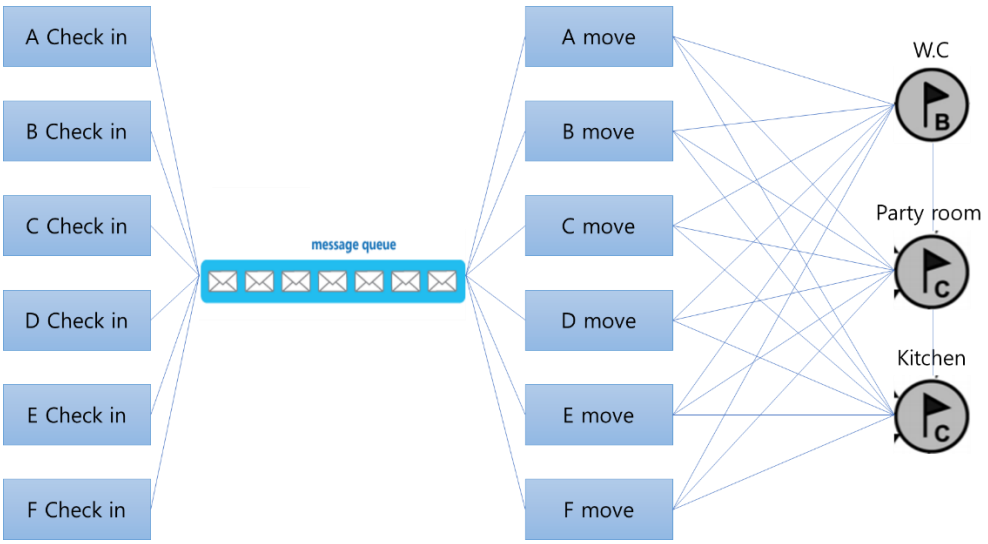


다음과 같이 게스트 하우스를 설계 해보았다.

첫번째 그림과 같이 6명의 사람이 체크인하여 각 방에 들어가 있다. 각 사람들은 이동하는 순서와 시간을 랜덤으로 입력받는다. 입력받은 후 message queue를 이용하여 사람들은 각 원하는 위치로 이동이 가능하다. 이때 이동하는 순서는 체크인을 한 순서이다. 순서를 알기 위해서 빨,주,노,초,파,남 순으로 우선순위를 정하였다. 공용 시설들은 파티룸: 6명, 주방: 2명, 화장실:1명으로 들어갈 수 있는 인원에 제한이 있다. 따라서 이동하려고 할 때 다음 장소에 인원이 모두 차 있으면 들어가지 못하고 누군가가 나올 때까지 기다려야한다. 파티룸, 주방, 화장실 3개의 Semaphore를 사용하여 구현할 수 있다.

게스트 하우스의 하루 숙박비는 10000원이다. 하지만 원하는 공용 시설을 이용하지 못했을 때는 게스트들의 불만으로 숙박비 500원이 감소된다. 또한 코로나로 인하여 파티룸에 5명 이상 들어가게 된다면 5인이상 집합 금지이므로 사장님은 1000원의 벌금을 내야한다.

사장님의 수익과 벌금을 총액을 표시해줄 수 있다.



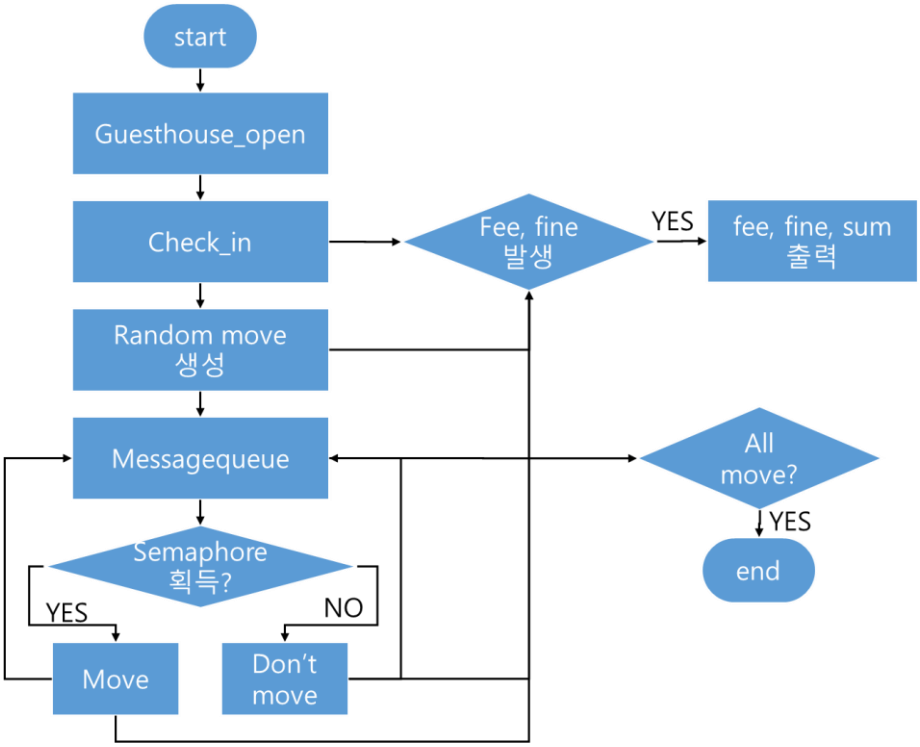
다음과 같이 semaphore와 message queue를 이용할 수 있다.

게스트 6명의 task를 생성한 후 각 랜덤으로 사용하는 공용시설과 시간을 입력 받아 메시지 큐에 저장할 수 있다. 그 후 우선순위대로 이동하고 semaphore인 공용시설을 이용할 수 있다.

이후 모든 사람들이 모든 사람의 메시지 큐가 사라지면 새로운 6명의 게스트들의 task를 받아올 수 있다.

B) 프로젝트의 구조

➢ Flow chart



게스트 하우스가 오픈해서 초기 화면이 출력된다. 총 6명의 사람이 체크인 한 후 각 사람마다 랜덤으로 이동하는 곳이 정해진다. A부터 차례대로 메세지 큐에 이동을 push 한다. 메시지 큐에서 pop하여 이동한다. 이동할 위치의 semaphore를 획득할 수 있다면 이동할 수 있고, 획득하지 못한다면 이동하지 못한다. 모든 사람들이 10번 움직일 때까지 반복하고 모두 움직였다면 종료한다. 체크인 할 때 5명의 이상이 모여있을 때, 움직이지 못하였을 때 fee, fine이 발생한다. 이때 sum을 출력해줄 수 있다.

➢ pseudo code

- main

```
OSInit(); // uC/OS-II 초기화

//semaphore 생성
sem_partyroom = OSSemCreate(6);
sem_kitchen = OSSemCreate(4);
sem_wc = OSSemCreate(1);

//message queue 생성
msg_q = OSQCreate(msg_array, (INT16U)N_MSG);

//task 생성
OSTaskCreate()

OSStart(); // 멀티 테스킹 시작
```

- check_in

```
 INT8U ary[11] = {1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3};

for(int j=A; j<=F; j++){
        for(int i=0; i<10; i++){
                j_move[i] = arr[rand()%11];
        }
        fee +=10000;
}
```

arr배열에 다음과 같이 저장한 후 0~10까지 랜덤함수를 발생하여 나온 수의 배열의 위치에 값을 저장한다.

A, B, C, D, E, F 모두 이동을 랜덤으로 받아오고 fee 10000을 지불한다.

- Task_Messagequeue_Post

```
for(int i=0; j<10; i++){
        for(int j=0; j<6; j++){
                sprintf(msg, " %d move₩n", (A~F)_move[i])
                QSQPost(msg_q, msg);
                OSTimeDly(1);
        }
}
```

6명 10번 이동 총 60번을 반복하여 이동을 queue에 저장한다.

- Task_Move

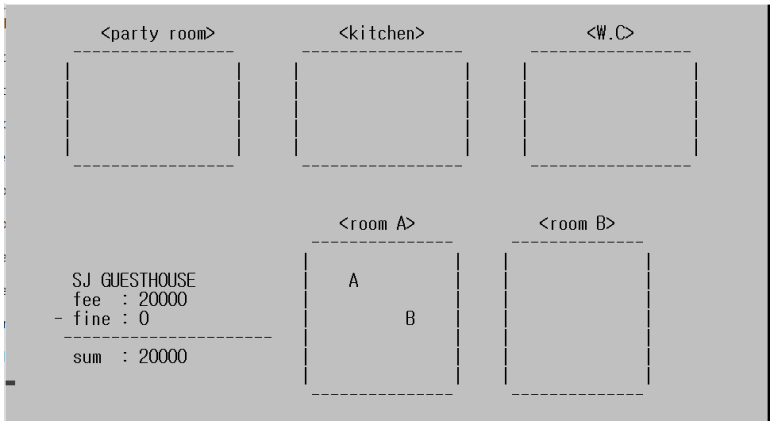
```
for(int i=0; j<10; i++){
        for(int j=0; j<6; j++){
                msg_pend = OSQPend(msg_q, 0, &err_q);
                fprintf(moveLog, "%s", msg_pend);
                fflush(moveLog); // fflush
                Remove();
                Move_arr();
        }
}
```

queue에서 메시지를 꺼내 log.txt에 저장한다. 이미 위치해 있던 곳을 지우고 이동할 곳의 위치를 표시해준다.

C) 각 task의 정의

- void Task_gusethouse_open(void *data)

  시작 task이다. 시작하면 다음과 같은 화면을 띄우고 순차적으로 다른 task들을 생성한다.



- void Task_Checkin(void *data)

  게스트 6명이 체크인하는 함수이다. A, B, C, D, E, F 가 순서대로 체크인 한다. A, B, C는 room A, D, E, F는 room B로 들어가는 것을 확인할 수 있다. 게스트하우스의 요금은 10,000원이므로 fee는 체크인 할 때마다 10000씩 증가한다. 체크인 할 때 각 사람은 총 열 번의 이동을 랜덤으로 이동하는 정보를 생성할 수 있다. 파티룸, 주방, 화장실을 이동할 수 있다. 화장실은 하나이므로 한사람이 너무 오래 쓰지 않도록 1/11의 확률로 나오도록, 주방은 4/11의 확률, 파티룸은 6/11의 확률을 가지도록 설정하였다.

- void Task_Messagequeue_Post(void * data)

  메세지 큐에 A부터 B의 이동을 저장하는 task이다. 이때 메시지 log.txt에 저장할 수 있다. 하나를 저장할 때마다 1초의 딜레이로 Task_Move로 이동하여 움직일 수 있다.

- void Task_Move(void * data)

  큐의 메시지에 따라서 파티룸, 주방, 화장실을 이동한다. 파티룸은 최대 정원이 6명, 주방은 최대 정원이 4명, 화장실은 1명이다. 이동할 위치에 정원이 다 차있으면 이동하지 못하고 room으로 돌아간다. 여기서 파티룸과 주방 화장실은 각자의 세마포어를 가지고 있어 인원을 제한할 수 있다. 한 사람이 P에서 P로 이동할 때

는 이미 그 곳에 있다고 판단하여 아무 일도 일어나지 않는다.

➢ void Task_Pay(void* data)

요금, 벌금, 합계를 나타내는 task이다. 숙박비는 10000원 사람들이 체크인할 때마다 사장님은 10000원을 벌게 된다. 사람들이 공용시설을 사용할 때 인원제한이 있어 사용하지 못한다면 만족도 감소로 벌금을 500원 낸다. 5명 이상 모여 있을 때 사장님은 벌금을 내게 된다. 사람이 체크인을 하거나, 이동하지 못할 때, 5인 이상 모여 있을 때 다음 task가 수행된다.

D) Task간 semaphore와 message queue의 활용방안

- Semaphore

파티룸, 주방, 화장시설은 공용시설이다. 모든 사람이 공유할 수 있지만 들어갈 수 있는 인원에 제한이 있어서 semaphore를 이용해서 동기화 시켜주고 관리해야 한다. 파티룸, 주방은 conting semaphore를 사용하고 화장실은 binary semaphore를 사용하여 구현할 수 있다.

sem_partyroom: 파티룸 세마포어, sem_kitchen: 주방 세마포어, sem_wc: 화장실 세마포어로 만들어 주었다.

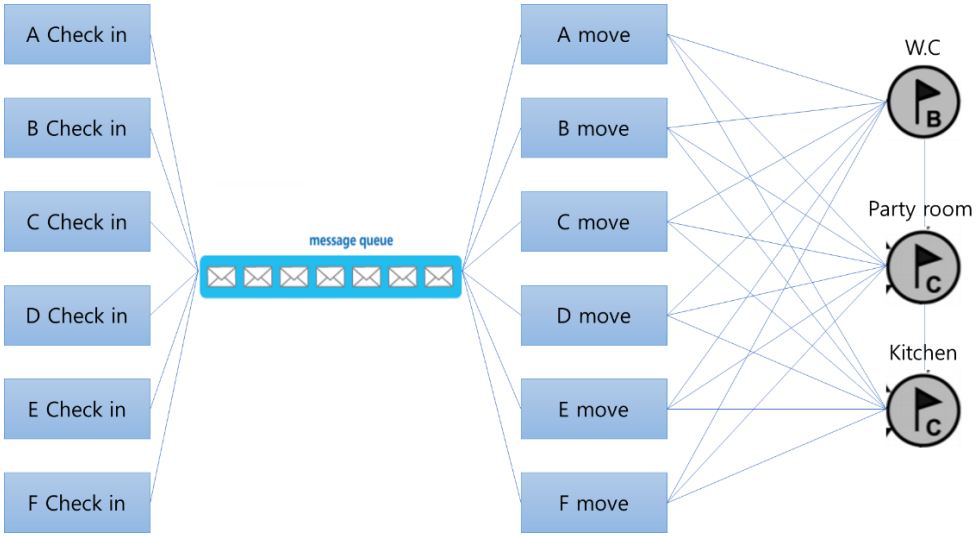
파티룸은 최대 정원이 6명, 주방은 최대 정원이 4명, 화장실은 1명이다. 그래서 파티룸은 6으로 초기화 주방은 4로 초기화 화장실은 1로 초기화 해주었다.

- queue

각 6명이 이동해야 할 메세지를 수신해야 한다. 메시지 큐를 사용하여 한 사람

씩 이동이 가능하다. 한사람이 이동하지 못한다면 방에서 대기하고 있는다.

6명의 사람이 이동할 위치를 랜덤으로 입력 받는다. A부터 F까지 메세지 큐에 넣은 후 메시지 큐에 맞게 사람이 이동할 수 있다. 한사람당 열 번씩 총 60번의 이동을 확인할 수 있다.



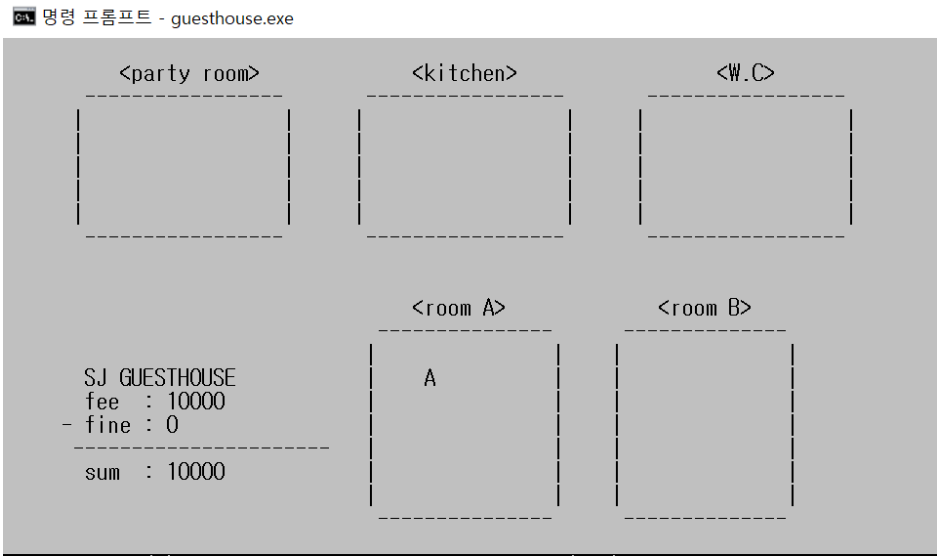
E) 제안서 대비 변경사항

- Task 변경

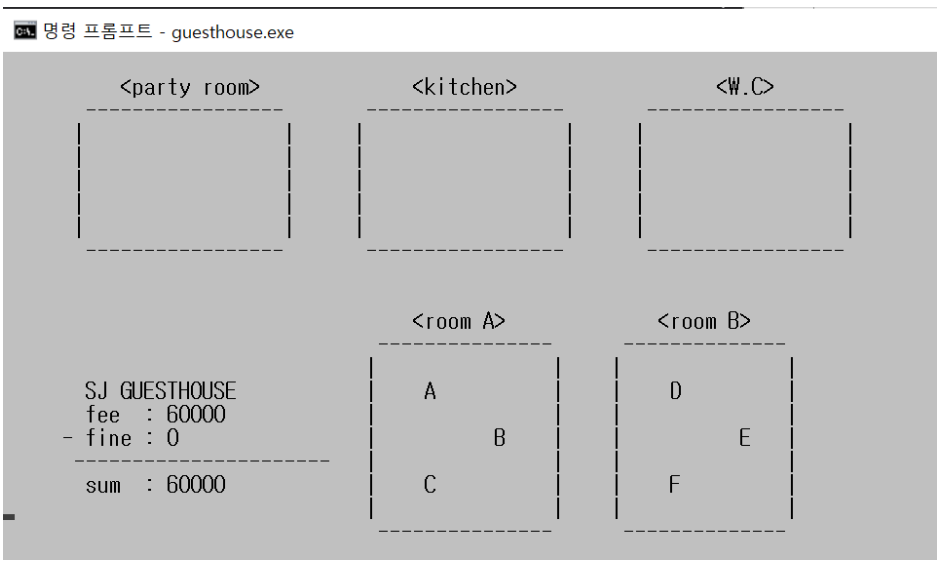
  A, B, C, D, E, F 각 각의 task를 만든 후 우선순위에 따라 이동하도록 설계하였지만, 반복되므로 하나의 task에서 delay를 주어 이동하도록 변경하였다.

- 주방에 들어갈 수 있는 인원이 2명이었지만 테스트를 해본 결과 이동하는 경우가 많지 않고, 벌금이 수익보다 더 커지는 경우가 발생하여 주방에 최대한으로 들어갈 수 있는 인원의 수를 4명으로 늘렸다.

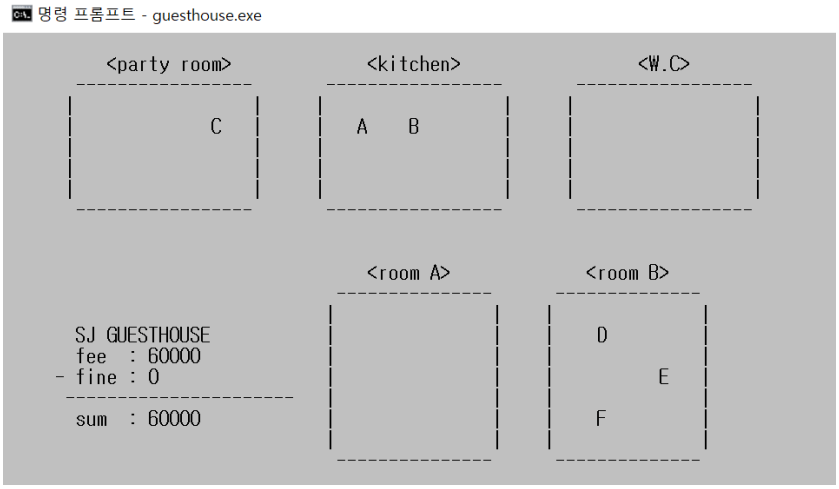
F) 동작 예시



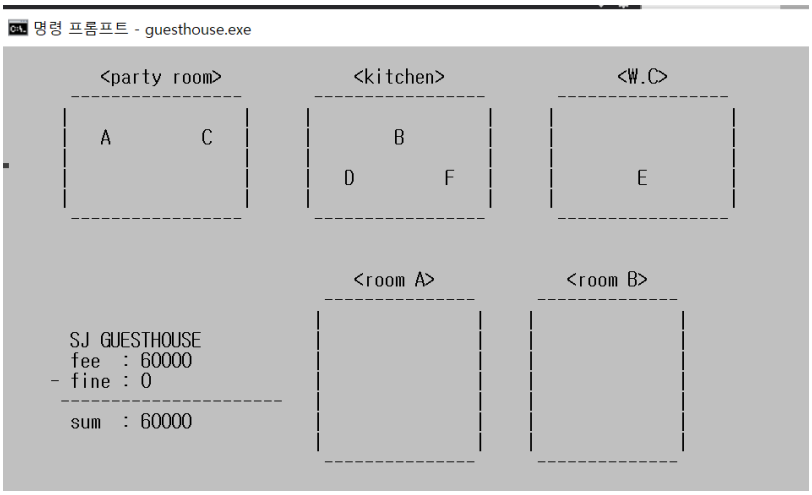
초기화면은 다음과 같다. A가 체크인하여 room A에 위치해 있고 요금을 10000을 지불하였다.



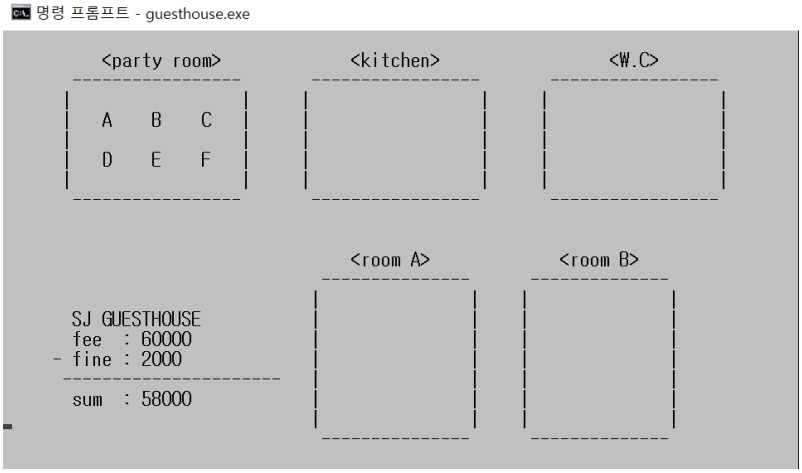
A, B, C, D, E, F가 모두 체크인해서 방에 모두 위치해 있고 요금을 각 10000원씩 지불하여 fee가 60000원이 되었다.



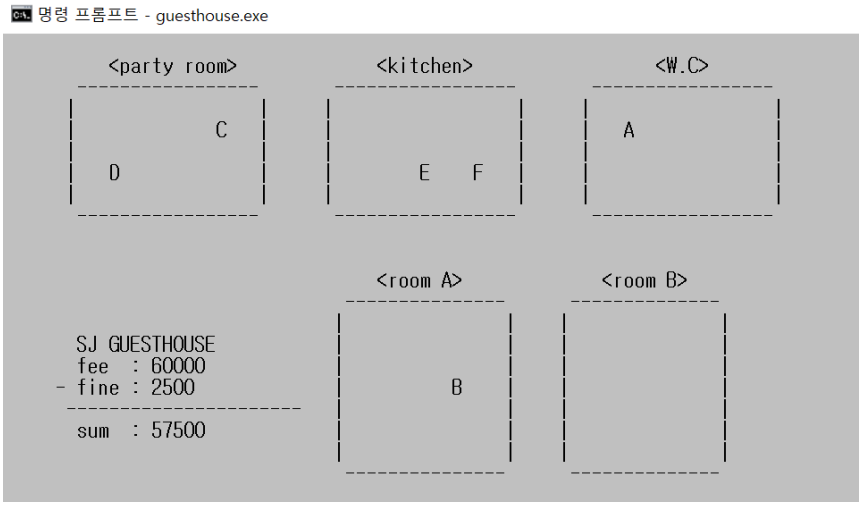
A부터 순차적으로 이동한다.



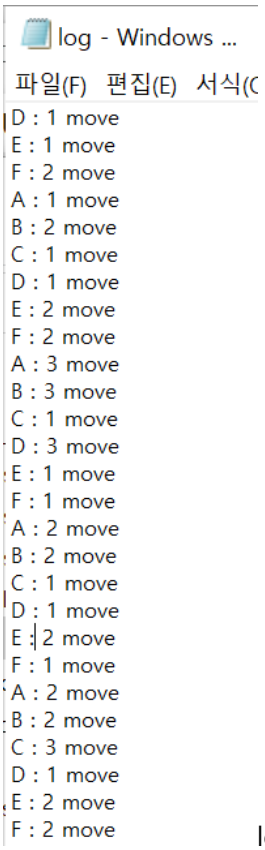
A, B, C, D, E, F 모두 이동하였다.



Party room에 5명 이상 모여 있어서 벌금이 발생하였다.



B가 원하는 곳으로 이동하지 못해서 room으로 돌아오고 벌금이 500원 발생한다.



log.txt에 다음과 같이 A부터 F까지 움직임이 저장되어 있다.

3. 고찰

이번 프로젝트는 message queue와 semaphore를 사용하여 실시간 임베디드 시스템을 구현하는 것이다. 어떤 임베디드 시스템을 설계해야할지 막막했다. 예시를 보고 보통 단순한 게임을 임베디드 시스템으로 구현하는 것을 보고 아이디어를 얻을 수 있었다.

설계할 때는 task를 이렇게 구현하면 되겠다고 생각했지만 구현을 실제로 해본 결과 뜻하는대로 되지 않았다. 우선순위가 매우 중요하고 우선 순위가 높은 task가 먼저 수행되고 우선 순위가 낮은 task가 수행된다. 하지만 이 우선 순위가 구현하는데 꽤 어려웠다. 원하는 task가 실행되게 하고 싶지만 우선 순위가 있기 때문에 쉽지 않았다. 그럴 때 suspend와 resume을 적절하게 사용하여 구현할 수 있었다.

직접 임베디드 시스템을 구현해서 task의 우선순위 흐름을 잘 알 수 있었다. 중간 고사 때 task가 이동하는 문제에서는 어떻게 이동하는지 알 수 없어서 많이 어려웠지만 직접 구현해보니 어떤 식으로 이동하는 지 알 수 있었다.