MUD_Game

200791 박지성

1. 서론

저번 Tic_Tac_Toe 게임에 이어서, 7주차까지 배웠던 내용을 복습, 실습하기 위한 프로젝트다. 기본적으로 사용자에게 입력받는 기능, 전체 지도와 함께 현재 위치를 출력하는 기능, 지도 밖으로 나가면 에러 메시지를 출력 후 재입력 기능, 목적지에 도착하면 종료하는 기능 등이 제공되고 (displayMap(), checkXY(), checkGoal() 함수 제공), 추가적으로 사용자에게 체력을 설정하여, 특정한 상황을 만났을 때 유저의 Hp가 어떻게 증감하는 지, 상호작용 이벤트 발생, 이동 시 Hp를 일정하게 감소하는 기능 등을 작성하도록 설계되어있다.

2. 요구사항

사용자 요구사항: 텍스트를 입력받아, 명령어(상하좌우 이동)를 수행하며, 목적지에 도착하는 게임

기능 계획

1) 사용자에게 “상”,”하”,”좌”,”우”, “지도”, “종료” 중 하나를 입력 받기
    1. 상하좌우 이동 후, 지도 자동 출력
    2. “지도”를 입력하면 전체 지도와 함께 현재 USER의 위치 출력
    3. 주어진 단어를 입력하지 않으면 에러 메시지 출력
2) 지도 밖으로 나가면 에러 메시지 출력 후 재입력
3) 목적지에 도착하면 성공 메시지 출력 후 종료
4) USER는 hp를 기본적으로 20을 가지고 시작
    1. 사용자가 명령문을 받을 때마다 사용자의 hp 1씩 감소
    2. hp 가 0이 되면 실패
5) 특정 이벤트가 발생하면 (아이템 , 적, 포션), 그에 대한 메시지 출력
    1. 적을 만날 경우 HP가 2가 줄어들고 추가 메시지 출력
    2. 포션을 만날 경우 HP가 2가 늘어나고 추가 메시지 출력

※ 단, USER가 움직이면서 감소하는 hp와 이벤트가 발생하여 조절되는 hp 는 독립적인 사건이다.

함수 계획

1) 메인 함수 : 사용자에게 값을 계속 입력받고, 그에 대한 함수 호출
2) displayMap() : 지도와 현재 위치 출력 함수
3) checkXY() : 사용자 위치 체크 함수
4) checkGoal() : 목적지에 도착 체크 함수
5) checkState() :현재 발생한 상호작용을 체크해줄 함수

3. 설계 및 구현

1. 기능 별 구현 사항

```
int main() {
    // 0은 빈 공간, 1은 아이템, 2는 적, 3은 포션, 4는 목적지
    int map[mapY][mapX] = { {0, 1, 2, 0, 4},
                            {1, 0, 0, 2, 0},
                            {0, 0, 0, 0, 0},
                            {0, 2, 3, 0, 0},
                            {3, 0, 0, 0, 2} };


    // 유저의 위치를 저장할 변수
    int user_x = 0; // 가로 번호
    int user_y = 0; // 세로 번호
    int user_hp = 20; // 유저의 체력  (기능 4번)

    // 게임 시작
    while (user_hp > 0) { // 사용자의 체력이 hp가 0보다 크면 무한 반복  (추가한 부분 )

        cout << "현재 HP: " << user_hp << endl; // 현재 체력 출력 (기능 6번)

        // 사용자의 입력을 저장할 변수
        string user_input = "";

        cout << "명령어를 입력하세요 (상,하,좌,우,지도,종료): ";
        cin >> user_input;
```

1. 기본적인 main 함수의 틀을 잡는 코드 블록
2. 입력
   - int map[][] : 전체 지도
   - int user_x : 사용자의 현재 x 위치값
   - int user_y : 사용자의 현재 y 위치값
   - int user_hp = 20 : 사용자의 체력 (20)
3. 결과
   - 사용자의 체력이 0보다 크면 무한 반복
   - 사용자의 현재 hp 출력
   - 사용자의 입력을 받아 저장

4. 설명
   - while문의 조건을 hp > 0 으로 설정하여 hp에 따라 무한 반복 설정
   - cin >> user_input 으로 사용자 입력을 받아 저장

```
if (user_input == "상") {
    // 위로 한 칸 올라가기
    user_y -= 1;
    bool inMap = checkXY(user_x, mapX, user_y, mapY); // bool 변수로 유저 위치, 맵 위치 확인
    if (inMap == false) { // 기능 2번 ( 지도 밖으로 나가면 에러 )
        cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
        user_y += 1;
    }
    else { // 기능 1번 (상하좌우 이동 )
        cout << "위로 한 칸 올라갑니다." << endl;
        displayMap(map, user_x, user_y);
        user_hp -= 1;// (기능 5번)
    }
}
else if (user_input == "하") {
    // TODO: 아래로 한 칸 내려가기
    user_y += 1;
    bool inMap = checkXY(user_x, mapX, user_y, mapY);
    if (inMap == false) {
        cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
        user_y -= 1;
    }
    else {
        cout << "아래로 한 칸 내려갑니다." << endl;
        displayMap(map, user_x, user_y);
        user_hp -= 1;
    }
}
else if (user_input == "좌") {
    // TODO: 왼쪽으로 이동하기
    user_x -= 1;
    bool inMap = checkXY(user_x, mapX, user_y, mapY);

    if (inMap == false) {
        cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
        user_x += 1;
    }
    else {
        cout << "왼쪽으로 이동합니다." << endl;
        displayMap(map, user_x, user_y);
        user_hp -= 1;
    }
}
else if (user_input == "우") {
    // TODO: 오른쪽으로 이동하기
    user_x += 1;
    bool inMap = checkXY(user_x, mapX, user_y, mapY);
    if (inMap == false) {
        cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
        user_x -= 1;
    }
    else {
        cout << "오른쪽으로 이동합니다." << endl;
        displayMap(map, user_x, user_y);
        user_hp -= 1;
    }
}
```

1. 상하좌우 이동 코드 블록
2. 입력
    - user_input : 사용자에게 입력받은 문자열
    - user_x , user_y : 사용자의 현재 x,y 좌표
    - checkXY() : 사용자의 위치 유효 검사 함수
    - displayMap() : 지도 출력 함수
    - user_hp : 사용자의 현재 체력

3. 결과
    - 사용자 입력
    - 먼저 사용자의 좌표를 바꿔보고, 좌표 유효 검사
    - 좌표가 유효하지 않으면 다시 좌표 원상복구
    - 좌표가 유효하면, 좌표를 바꾸고 지도 출력
    - 이동했으면 hp를 1 감소

4. 설명
    - if 조건에 사용자의 입력에 따라 달라지는 실행문
    - checkXY() 함수 호출값을 부울 변수 inMap에 대입
    - if로 inMap이 false면 에러메세지 출력 후 좌표 되돌리기
    - false가 아니면 정상 이동후, displayMap() 호출로 지도 출력

```
else if (user_input == "지도") {
    // TODO: 지도 보여주기 함수 호출
    displayMap(map, user_x, user_y);
}
else if (user_input == "종료") {
    cout << "종료합니다.";
    break;
}
else {
    cout << "잘못된 입력입니다." << endl;
    continue;
}
```

1. 사용자가 "지도" , "종료" , 그 외 다른 것을 입력한 경우의 코드블록

2. 입력
   - user_input : 사용자에게 입력받는 문자열
   - displayMap() : 지도 출력 함수

3. 결과
   - 지도 입력시 지도 출력
   - 종료 입력시 종료
   - 그 외에 다른 것을 입력한 경우 에러 메세지 출력

4. 결과
   - "지도" 를 입력하면 displayMap() 함수 호출
   - "종료" 를 입력하면 break로 반복 종료
   - 그 외의 것을 입력하면 에러 메세지 출력 후 continue

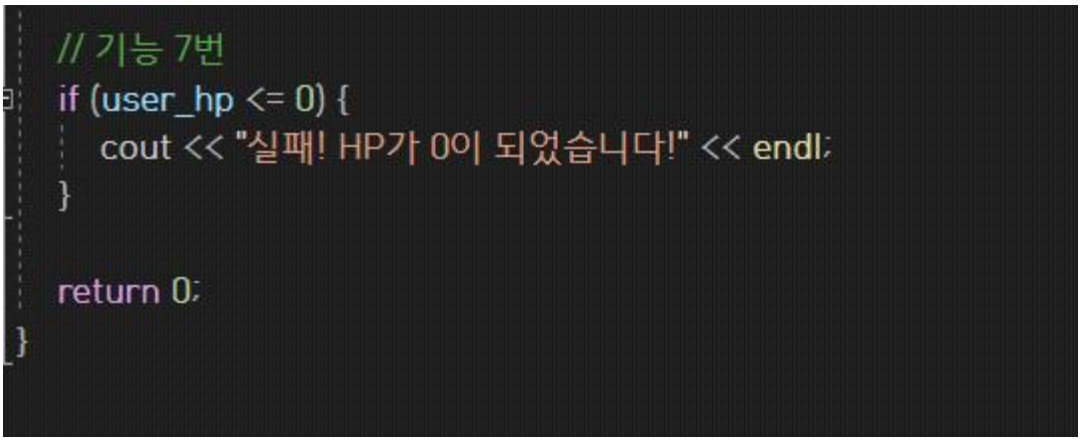
```
    // 기능 3번 연계. 목적지에 도달했는지 체크
    bool finish = checkGoal(map, user_x, user_y);
    if (finish == true) {
        cout << "목적지에 도착했습니다! 축하합니다!" << endl;
        cout << "게임을 종료합니다." << endl;
        break;
    }

    checkState(map, user_x, user_y, user_hp);


}
```

1. 목적지에 잘 도착했는 지 체크 + 상호작용 함수 호출 블럭

2. 입력
   - user_x , user_y : 사용자의 x, y 좌표
   - map[][] : 아이템, 적, 목적지 등의 위치를 인덱스로 가지고 있는 배열

3. 결과
   - 사용자가 목적지에 도착하면 축하 메세지 출력후 종료

4. 설명
    - 부울변수에 checkGoal() 함수의 반환값 대입
    - true면 축하메세지 출력 후 break
    - 반복문 바깥에서 checkState() 함수 호출



1. hp가 0이 되면 메세지 출력
2. 입력
    - user_hp : 사용자의 hp

3. 결과
    - 사용자의 hp가 0이 되거나, 그보다 더 작아지면 종료 메세지 출력
4. 설명
    - user_hp가 <= 0 이 되면 실패 메세지 출력

```cpp
// 기능 8번. 지도와 사용자 위치 출력하는 함수
void displayMap(int map[][mapX], int user_x, int user_y) {
    for (int i = 0; i < mapY; i++) { // 맵 크기만큼 2차원 배열 생성
        for (int j = 0; j < mapX; j++) {
            if (i == user_y && j == user_x) { // 만약 유저의 위치가 배열의 인덱스 x,y 위치가 같으면
                cout << " USER |"; // 양 옆 1칸 공백
            }
            else { // 그게 아니면 USER가 아닌 위치
                int posState = map[i][j];
                switch (posState) {
                case 0:
                    cout << "      |"; // 6칸 공백
                    break;
                case 1:
                    cout << "아이템|";
                    break;
                case 2:
                    cout << "  적  |"; // 양 옆 2칸 공백
                    break;
                case 3:
                    cout << " 포션 |"; // 양 옆 1칸 공백
                    break;
                case 4:
                    cout << "목적지|";
                    break;
                }
            }
        }
        cout << endl;
        cout << " ------------------------------ " << endl;
    }
}
```

1. 지도와 사용자의 위치 출력 함수 코드 블록

2. 입력
   - mapX, mapY : 맵의 가로, 세로 크기 상수
   - user_x, user_y : 사용자의 x, y 좌표

3. 결과
   - 3*3 크기만큼의 격자 무늬 지도
   - 사용자의 위치가 지도 상에서 어디인지 출력
   - 아이템, 적, 목적지, 포션등의 위치를 지도 상에서 출력
   - 아무것도 없는 곳은 공백으로 출력

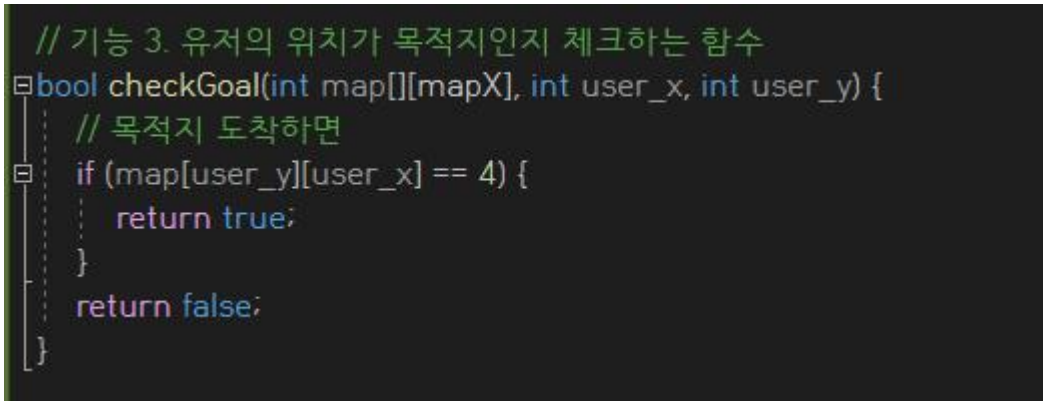
4. 설명
    - 이중 반복문으로 3*3 지도 크기 생성
    - if 조건으로 지도 상의 x,y위치가 사용자의 x,y 좌표가 같으면 그 위치를 "USER" 라고 지도상에 표시
    - else문으로 사용자가 없는 위치에 아이템, 적, 포션, 목적지등을 배치
    - postate 변수값을 기준으로 switch문이 돌아가며, 각 case마다 출력할 텍스트가 달라짐

```
// 기능 2. 이동하려는 곳이 유효한 좌표인지 체크하는 함수
bool checkXY(int user_x, int mapX, int user_y, int mapY) {
    bool checkFlag = false; // 먼저 bool 변수 하나를 false 값으로 지정
    if (user_x >= 0 && user_x < mapX && user_y >= 0 && user_y < mapY) { // 유저가 맵 안에 있으면
        checkFlag = true; // 플래그 변수를 true 값으로 바꿔서 유저를 움직이고, 그게 아니면 에러 메세지 출력으로 연결
    }
    return checkFlag;
}
```

1. 유효 좌표 체크 코드 블록
2. 입력
    - user_x, user_y : 사용자의 x,y좌표값
    - mapX, mapY : 맵 크기만큼의 상수

3. 반환값
    - checkFlag 부울 변수
4. 결과
    - 사용자가 맵 안에 있으면 움직이도록 설정
    - 사용자가 맵 밖으로 나가려고 하면 에러 메세지 출력으로 연결

5. 설명
    - checkFlag 부울 변수를 false로 먼저 지정
    - if 조건으로 유저가 맵안에 있으면 부울 변수를 true로 설정
    - checkFlag 부울 변수를 반환값으로 설정



```
// 기능 3. 유저의 위치가 목적지인지 체크하는 함수
bool checkGoal(int map[][mapX], int user_x, int user_y) {
    // 목적지 도착하면
    if (map[user_y][user_x] == 4) {
        return true;
    }
    return false;
}
```

1. 목적지 체크 함수 코드 블록

2. 입력
   - user_x , user_y : 사용자의 x, y 좌표
   - map[][] : 아이템, 적, 목적지 등의 위치를 인덱스로 가지고 있는 배열

3. 결과
   - 사용자가 목적지에 도달 했는지 안했는 지를 체크해줌
4. 설명
   - 부울 함수로 설정된 checkGoal
   - if 조건으로 지도 상에 사용자의 x,y위치가 목적지면, true 값 반환
   - 그게 아니면 무조건 false 반환

```
// 이동하려는 곳에서의 상호작용
void checkState(int map[][mapX], int user_x, int user_y, int& user_hp) {
    int posState = map[user_y][user_x];
    switch (posState) {
    case 0:
        cout << "아무 것도 없습니다." << endl;
        break;
    case 1:
        cout << "아이템이 있습니다." << endl;
        break;
    case 2:
        user_hp -= 2; // 적을 만났을 때 HP 감소
        cout << "적이 있습니다! HP가 -2 감소합니다 !" << endl;
        break;
    case 3:
        user_hp += 2; // 포션을 사용하여 HP 회복
        cout << "포션을 사용하여 HP가 + 2 증가합니다 !" << endl;
        break;
    case 4:
        // 목적지에 도착한 경우 아무것도 하지 않음
        break;
    }
}
```

1. 상호작용 이벤트 함수 코드 블록

2. 입력
   - map[][] : 아이템, 적, 목적지 등의 위치를 인덱스로 가지고 있는 배열
   - user_x, user_y : 사용자의 x, y 좌표
   - user_hp : 사용자의 현재 체력

3. 결과
   - 지도 상에서 사용자가 공백 칸으로 이동하면 , 공백 메세지 출력
   - 이하 사용자가 아이템이 있는 칸으로 이동하면, 아이템 메세지 출력
   - 이하 사용자가 적이 있는 곳으로 이동하면, hp를 2 감소시키고 메세지 출력
   - 이하 사용자가 포션이 있는 곳으로 이동하면, hp를 2 증가시키고 메세지 출력
   - 사용자가 목적지에 도달하면 아무것도 하지 않음

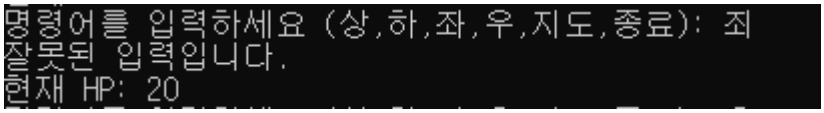
4. 설명
   - 지도 출력 코드블록을 이용해서 동일한 방식으로 이벤트 함수 작성
   - posState 변수를 switch문에 조건으로 넣어서 각 case 별로 이벤트 메세지 출력
   - 특별하게 hp를 증감할일이 있는 경우엔 user_hp 조절

4. 테스트

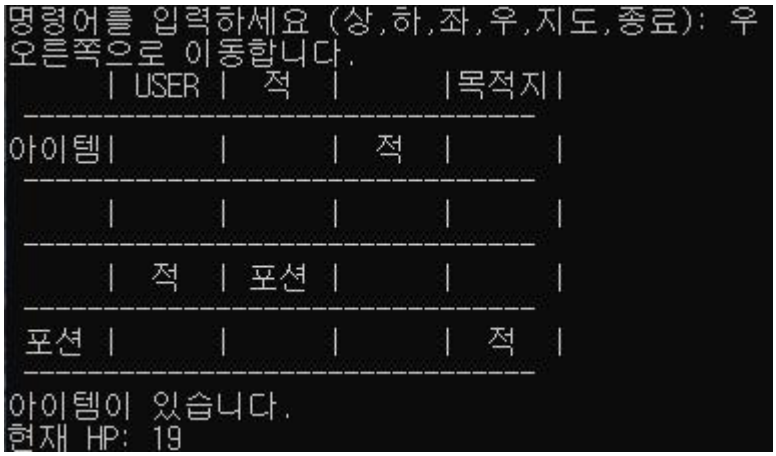
1. 기능 별 테스트 결과: (요구사항 별 스크린샷)

① 지도 밖으로 나가게 되면 에러 메세지 출력

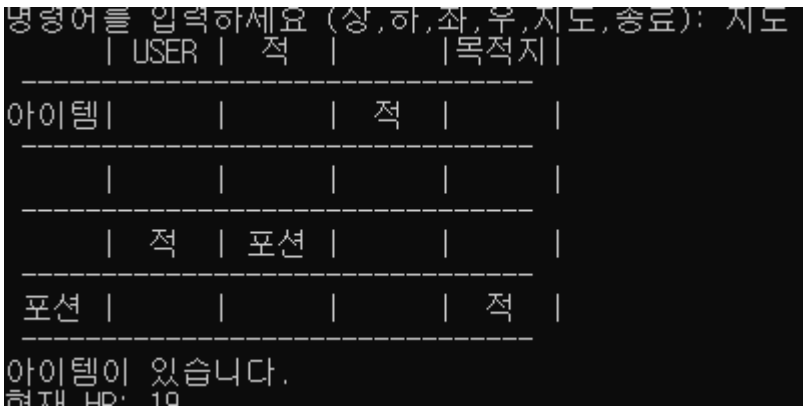
② 잘못된 문자열을 입력할 경우 에러메세지 출력



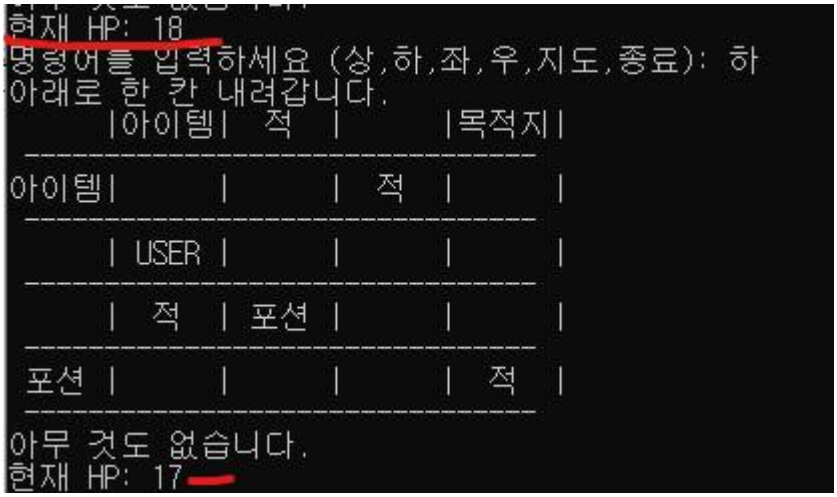
③ 상하좌우 입력시 지도 자동출력



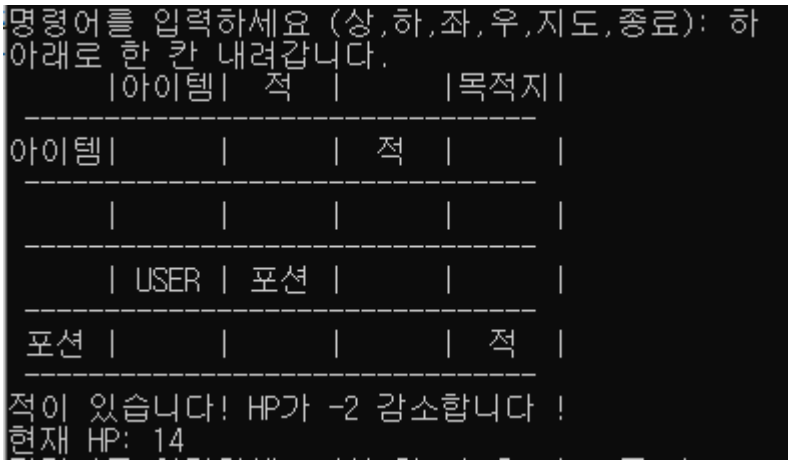
④ 지도 입력 시 지도 출력



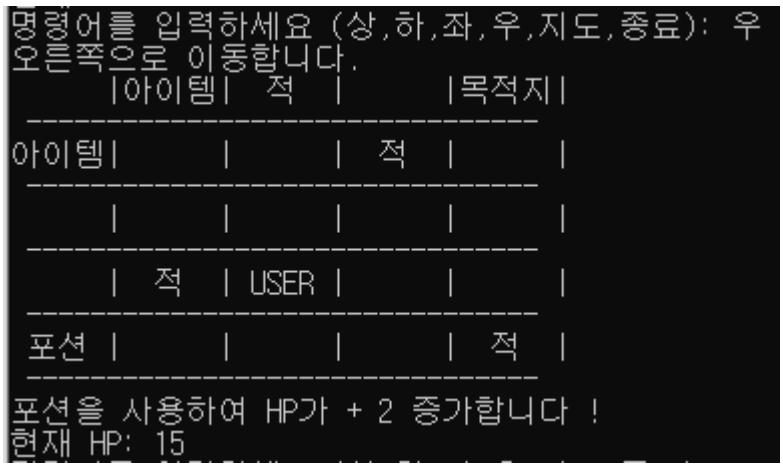
⑤ hp는 사용자가 움직일때마다 1씩 감소



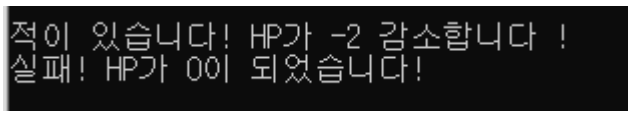
⑥ 적을 만나면 hp 2씩 감소



⑦ 포션을 만나면 hp 2 증가

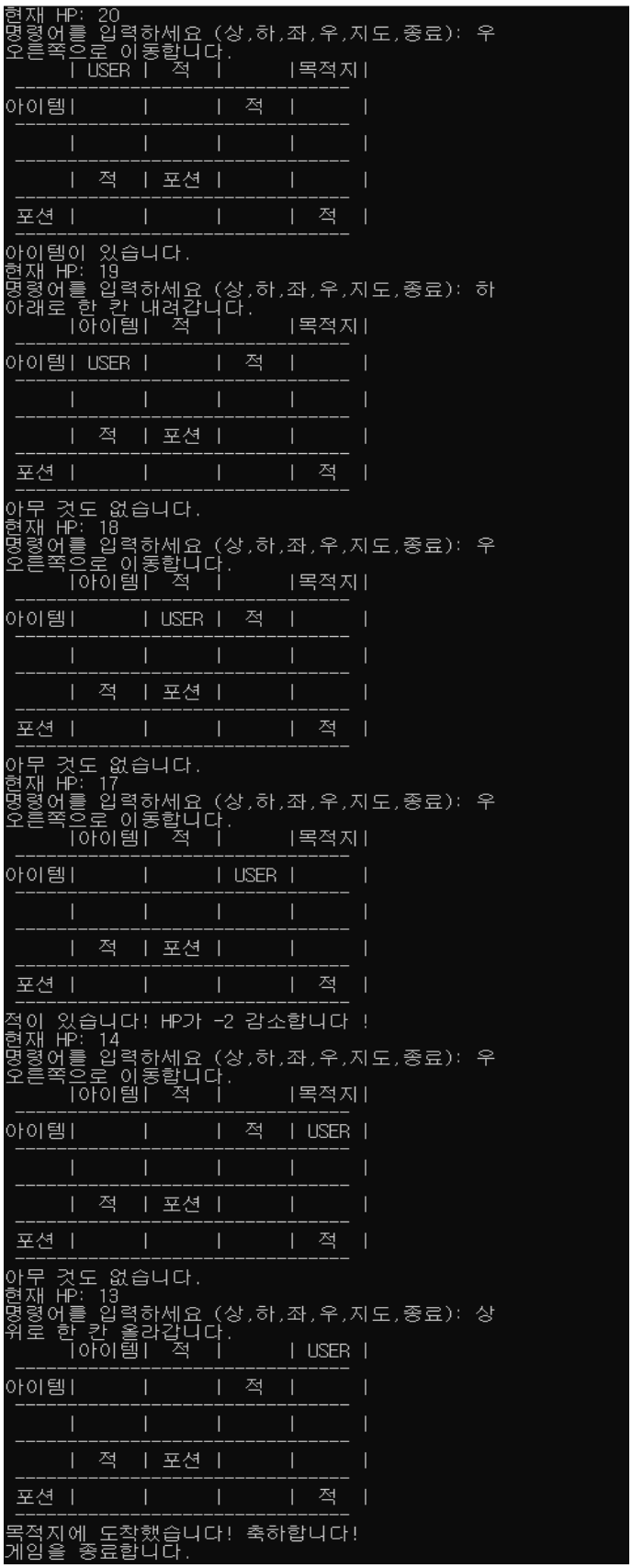


⑧ hp가 0이 되면 게임 종료



⑨ 종료를 입력하면 게임 종료

2. 최종 테스트 스크린샷: (프로그램 전체 동작 스크린샷



5. 결과 및 결론

MUD를 만들면서, 함수의 편리성은 확실히 깨달을 수 있었다.
함수를 이용한 호출, 반환 방식이 편리하다는 것도 알기 쉬웠지만, 다른것보다 switch문에 posState 변수를 넣어서 case마다 위치에 특정 이벤트를 보이게 하고, 발생시키는 것도 굉장히 새로웠다.

지도를 출력하는 함수를 살펴보면, 배열 + 이중 반복 + switch 문 + if 문등
수업시간에 다뤘던 다양한 문법들이 한 번에 사용된 함수였는데, 배열의 인덱스에 사용자의 위치를 비교해서 USER의 위치를 표시하는 법이나, 그 외에 else 문을 switch 문으로 처리하는 방식이 확실히 이론적으로 지식만 알아서는 코드를 짤 수 없겠다는 생각이 들었다.

그래서 이런 실습적인 과제를 하는 것도 좋고, 처음부터 끝까지 무리하게 짜는 것보다 어느 정도 틀을 잡아주신 코드에 코드 블럭만 작성하는 등, 개인적으로는 참 마음에 드는 수업, 과제라고 생각한다.