1. 서론

1. 프로젝트 목적 및 배경: 7주차까지 배운 내용에 대한 실습을 위해 진행

2. 목표: 간단한 Mud 게임 구현

2. 요구사항

1. 사용자 요구사항: 유저가 아이템과 적, 포션이 있는 필드를 이동하며 목적지에 도착하는 게임

2. 기능 계획:

① 사용자에게 "상", "하", "좌", "우", "지도", "종료" 중 하나를 입력 받기

- 상/하/좌/우 입력시 해당 방향으로 이동 후 지도 출력
- "지도"를 입력하면 전체 지도와 함께 현재 위치를 출력
- 이 중 다른 것을 입력하면 에러 메시지 출력 후 재 입력 요청

② 지도 밖으로 나가게 되면 에러 메시지 출력

③ 목적지에 도착하면 "성공"을 출력하고 종료

④ 유저는 체력 20을 가지고 게임 시작

⑤ 사용자가 이동할 때 마다 사용자 체력 1씩 감소

⑥ 처음 명령문을 입력 받을 때 마다 HP 함께 출력

⑦ HP가 0이 되면 "실패"를 출력하고 종료

⑧ 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력

- 예) {X}이/가 있습니다.
- 적을 만날 경우 HP가 2가 줄어들고 그에 대한 추가 메시지 출력
- 포션을 만날 경우 HP가 2가 늘어나고 그에 대한 추가 메시지 출력
- (적이나 포션 등은 사라지지 않음을 전제)

3. 함수 계획

① 메인 함수: 사용자에게 값을 계속 입력받고, 그에 대한 함수 호출

② 지도와 현재 위치 출력 함수: displayMap()

③ 사용자 위치 체크 함수: checkXY()

④ 목적지에 도착 체크 함수: checkGoal()

⑤ 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력하는 함수: checkState()

3. 설계 및 구현

1. 기능 별 구현 사항:

① 사용자에게 “상”, ”하”, ”좌”, ”우”, “지도”, “종료” 중 하나를 입력 받기

② 지도 밖으로 나가게 되면 에러 메시지 출력

④ 유저는 체력 20을 가지고 게임 시작

⑥ 처음 명령문을 입력 받을 때 마다 HP 함께 출력

1. 함수 스크린샷

```
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; // 유저 hp
```

```cpp
// 게임 시작
while (1) { // 사용자에게 계속 입력받기 위해 무한 루프

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

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

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

2. 입력

   • int map[][] = 전체 지도

   • int user_x, user_y, user_hp = 유저 x, y, hp 값

3. 반환값

4. 결과

   • 유저의 입력에 따라 상, 하, 좌, 우로 이동하는 방향으로 user_x, user_y값을 변경

   • 지도를 입력하면 지도 출력

   • 종료를 입력하면 종료

• 상하좌우 지도 종료 외의 단어가 입력되면 잘못된 입력이라는 알림 후 다시 입력을 받음

• 지도를 벗어나면 알림 후 다시 입력받음

• 이동 후 지도 출력

• 명령어 입력 전 현재 hp 출력

5. 설명

• 현재 hp 출력

• 유저의 입력을 받은 후 그에 따른 명령 수행

• 상, 하, 좌, 우의 경우 이동하는 방향으로 user_x, user_y 값 변경 후 현재 지도 출력, 변경된 값이 지도 밖이면 알림 후 다시 입력을 받음

• 지도 입력 시 현재 지도 출력

• 종료 입력 시 종료

• 이 이외의 명령이 입력 될 시 잘못된 입력이라는 알림 후 다시 입력을 받음

⑤ 사용자가 이동할 때 마다 사용자 체력 1씩 감소

⑦ HP가 0이 되면 “실패”를 출력하고 종료

1. 함수 스크린샷

```
user_hp -= 1;

checkState(map, user_x, user_y, user_hp);

// Hp가 0 이하인지 체크
if(user_hp <= 0){
    cout << "HP가 0 이하가 되었습니다. 실패했습니다." << endl;
    cout << "게임을 종료합니다." << endl;
    break;
}
```

2. 입력

• int user_hp = 유저 hp 값

3. 반환 값

4. 결과

- 유저 hp의 값을 1 감소시키고 user_hp의 값이 0 이하이면 알림 후 게임 종료

5. 설명
   - 유저 hp의 값을 1 감소시킨다.
   - user_hp의 값이 0 이하인지 체크하고 0 이하이면 알림 후 게임 종료

지도와 현재 위치 출력 함수

1. 함수 스크린샷

```
// 지도와 사용자 위치 출력하는 함수
void displayMap(int map[][mapX], int user_x, int user_y) {
    for (int i = 0; i < mapY; i++) {
        for (int j = 0; j < mapX; j++) {
            if (i == user_y && j == user_x) {
                cout << " USER |"; // 양 옆 1칸 공백
            }
            else {
                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;
    }
}
```

2. 입력

- int map[][] = 전체 지도
- user_x, user_y = 유저 x, y값

3. 반환 값
   - 없음
4. 결과
   - 전체 지도를 출력
   - 사용자 위치를 출력
5. 설명
   - 2차원 배열에 있는 맵을 출력
   - 사용자 위치와 동일한 좌표를 발견할 경우 사용자 정보 출력

사용자 위치 체크 함수

1. 함수 스크린샷

```
// 이동하려는 곳이 유효한 좌표인지 체크하는 함수
bool checkXY(int user_x, int mapX, int user_y, int mapY) {
    bool checkFlag = false;
    if (user_x >= 0 && user_x < mapX && user_y >= 0 && user_y < mapY) {
        checkFlag = true;
    }
    return checkFlag;
}
```

2. 입력
   - user_x, user_y = 유저 x값, 유저 y값
   - mapX, mapY = map의 X, Y 크기
3. 반환 값
   - 유효하면 true, 유효하지 않으면 false
4. 결과
   - 유저의 좌표를 검사하여 true 또는 false 출력
5. 설명
   - 유저의 좌표가 map 안에 있으면 true 이외에는 false 출력

목적지에 도착 체크 함수

1. 함수 스크린샷

```
// 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;
}
```

2. 입력
   - int map[][] = 전체 지도
   - user_x, user_,y = 유저 x, y값
3. 반환 값
   - true or false
4. 결과
   - 목적지에 도착하면 true 아니면 false
5. 설명
   - 현재 유저의 좌표가 map의 4(목적지)가 있는 좌표면 true, 아니면 false

무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력하는 함수

1. 함수 스크린샷

```
void checkState(int map[][mapX], int user_x, int user_y, int& user_hp){
    //아이템
    if(map[user_y][user_x] == 1){
        cout << "아이템이 있습니다." << endl;
    }
    //적
    else if(map[user_y][user_x] == 2){
        user_hp -= 1;
        cout << "적이 있습니다. HP가 2 감소합니다." << endl;
    }
    //포션
    else if(map[user_y][user_x] == 3){
        user_hp += 3;
        cout << "포션이 있습니다. HP가 2 증가합니다." << endl;
    }
}
```

2. 입력

   • int map[][] = 전체 지도

   • user_x, user_y = 유저 x, y값

   • user_hp = 유저 hp값

3. 반환 값

   • 없음

4. 결과

   • 유저의 x, y값에 아이템이 있으면 알림, 적이 있으면 알림 후 hp에 1 추가 감소, 포션이 있으면 hp 3 증가

5. 설명

   • 유저의 x, y값에 1이 있으면 아이템, 2가 있으면 적, 3이 있으면 포션

4. 테스트

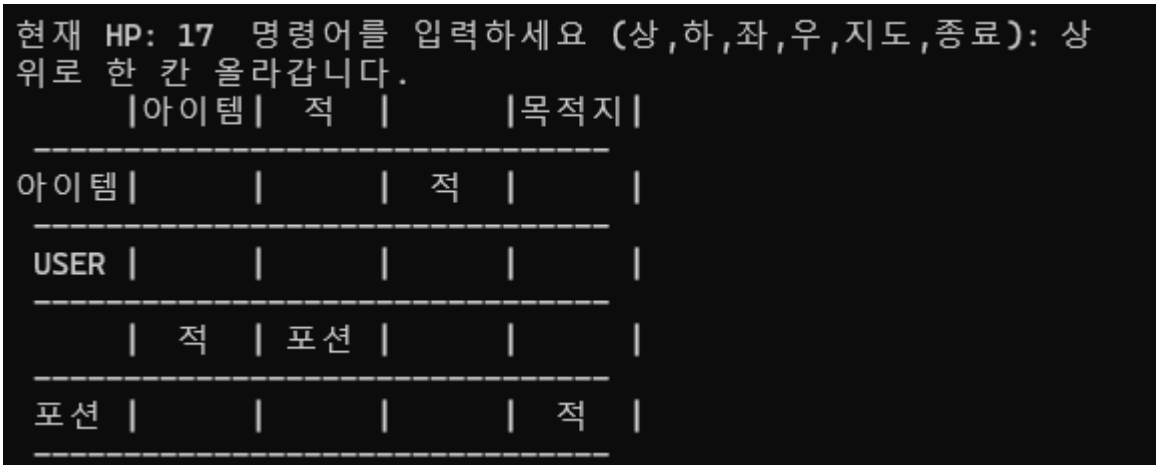
1. 기능 별 테스트 결과:

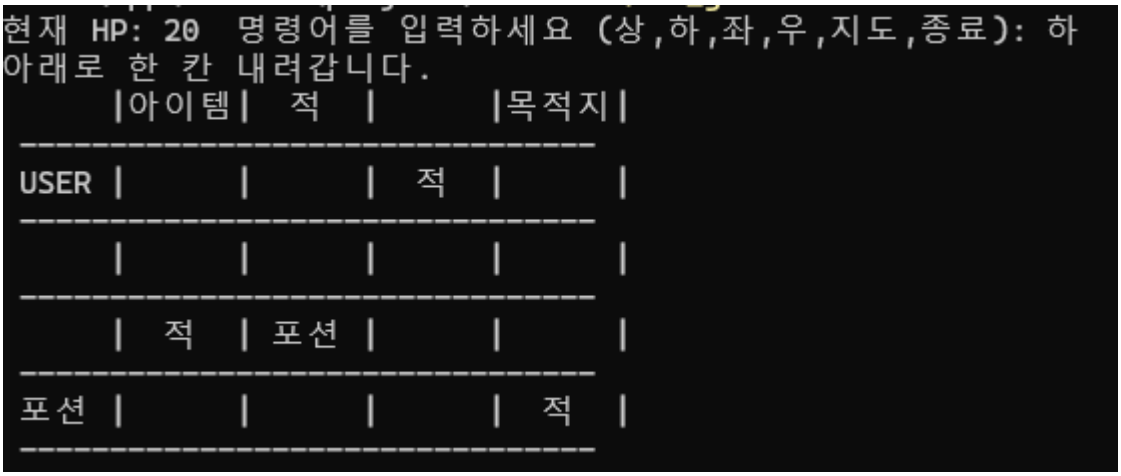
① 사용자에게 “상”, ”하”, ”좌”, ”우”, “지도”, “종료” 중 하나를 입력 받기

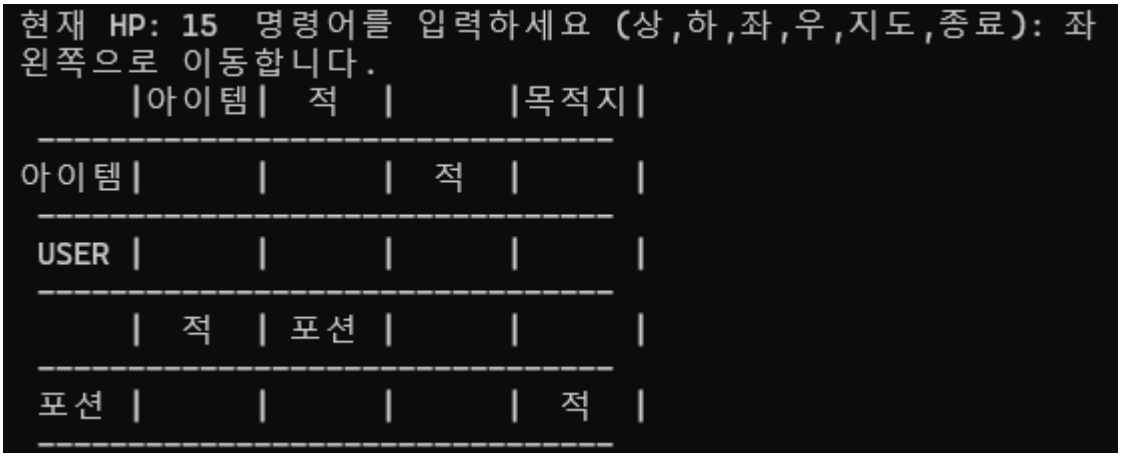
• 상/하/좌/우 입력시 해당 방향으로 이동 후 지도 출력

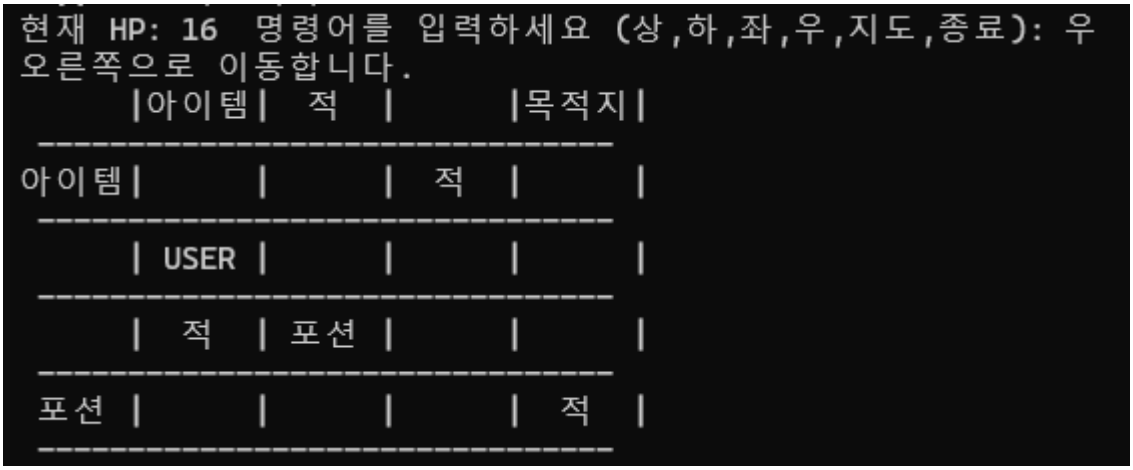
• “지도”를 입력하면 전체 지도와 함께 현재 위치를 출력

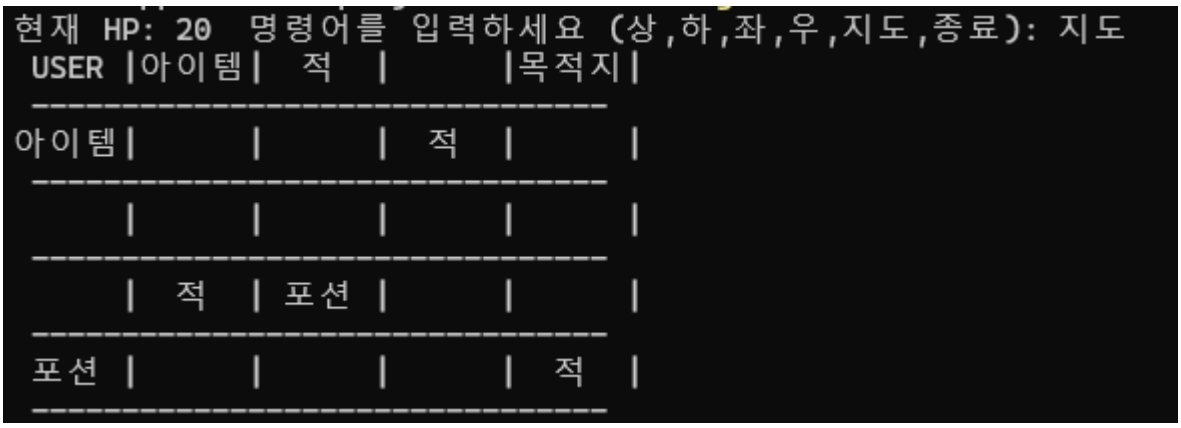
• 이 중 다른 것을 입력하면 에러 메시지 출력 후 재 입력 요청

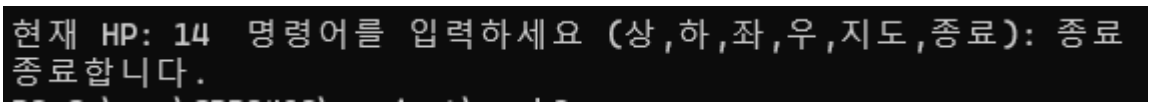


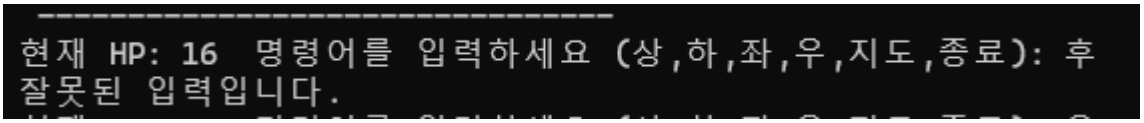




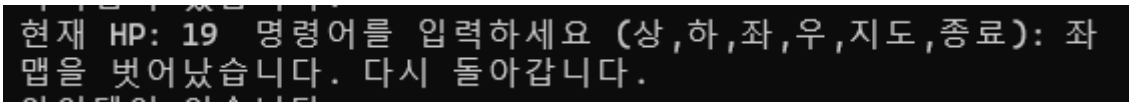




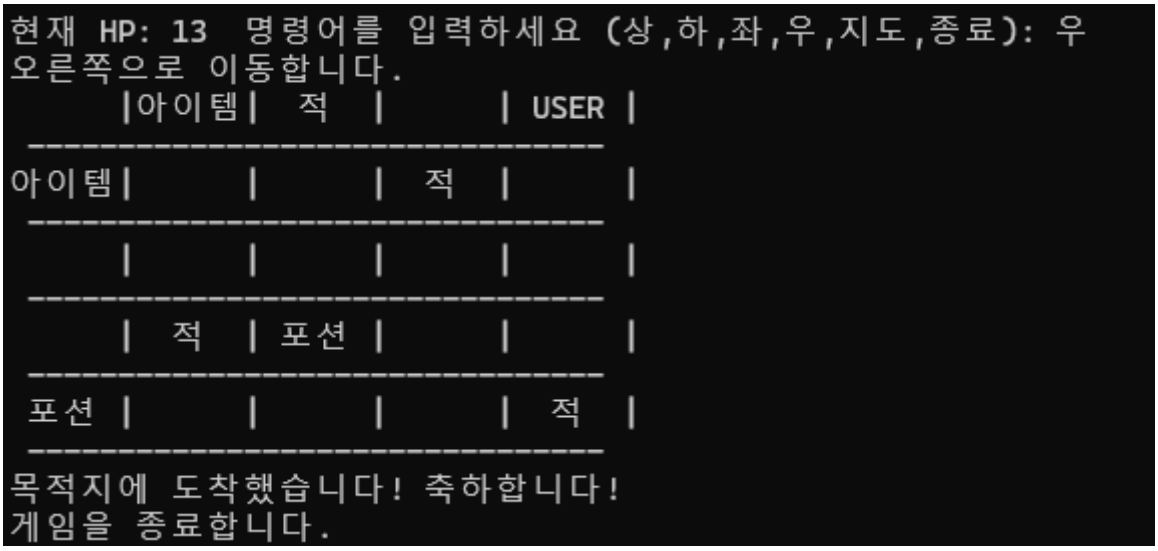




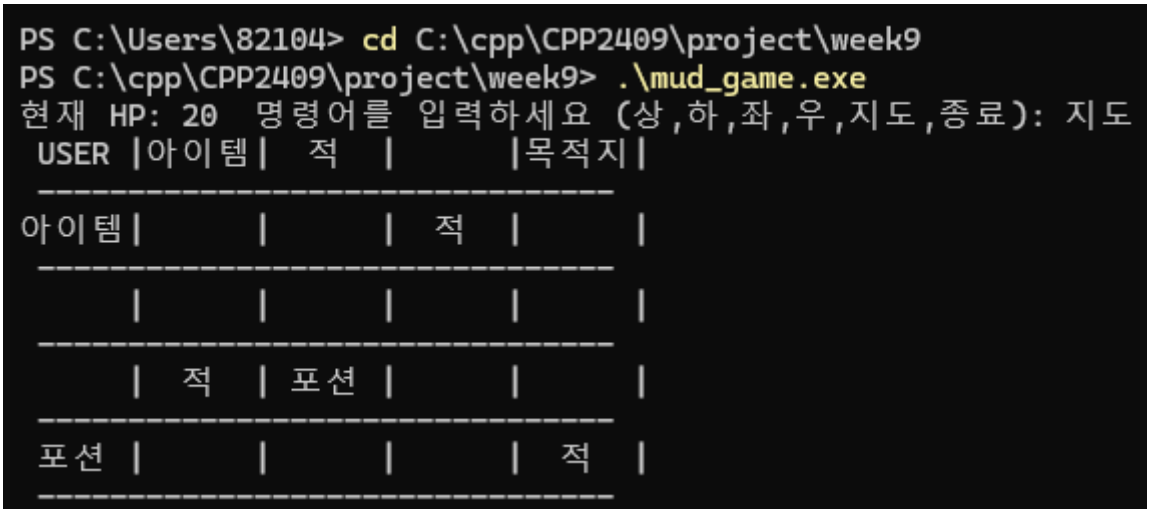
② 지도 밖으로 나가게 되면 에러 메시지 출력



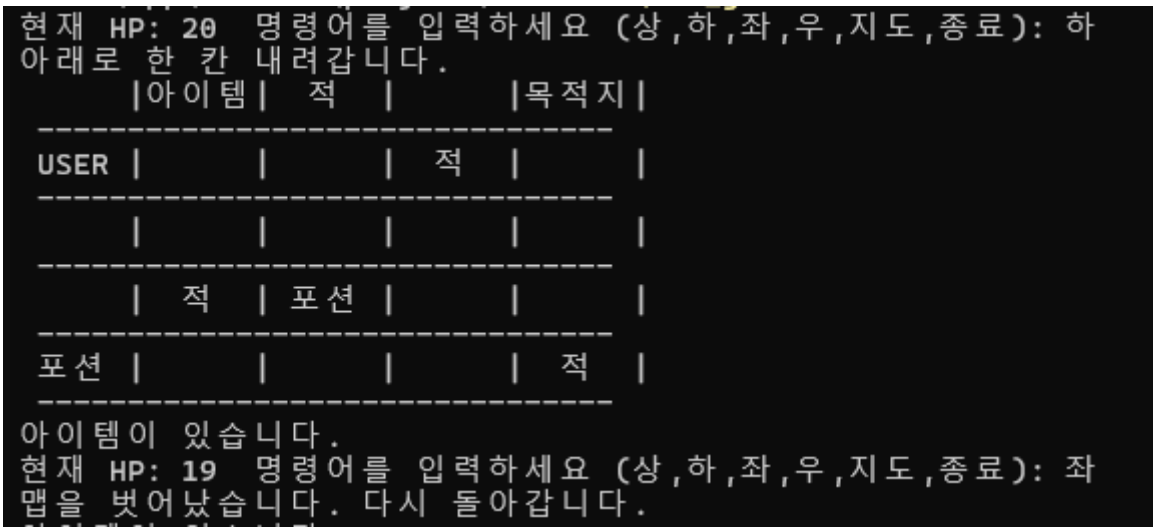
③ 목적지에 도착하면 “성공”을 출력하고 종료



④ 유저는 체력 20을 가지고 게임 시작

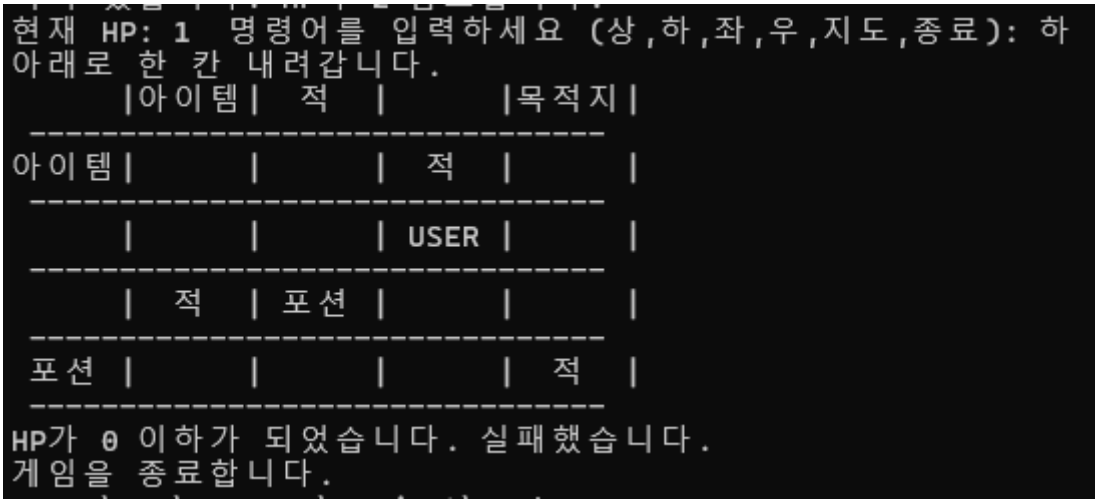


⑤ 사용자가 이동할 때 마다 사용자 체력 1씩 감소

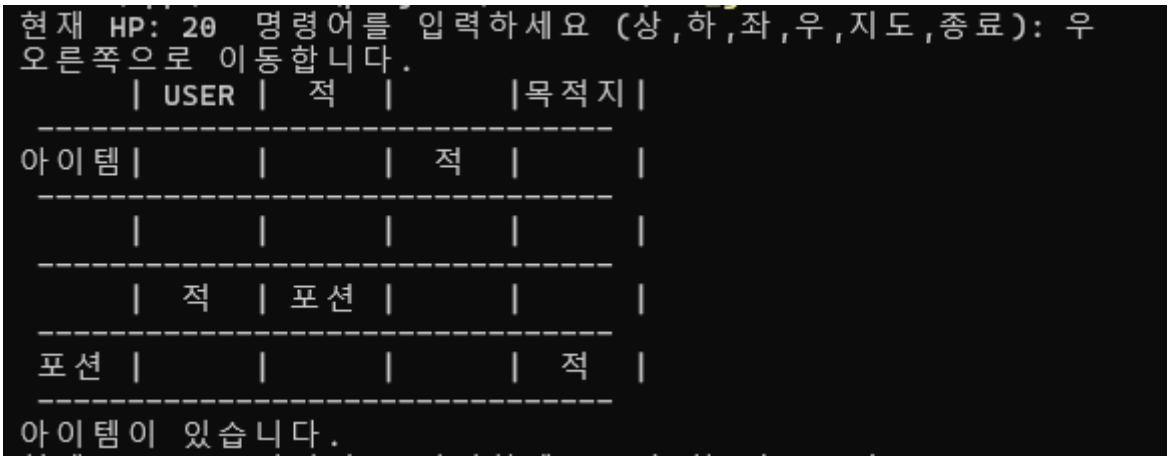


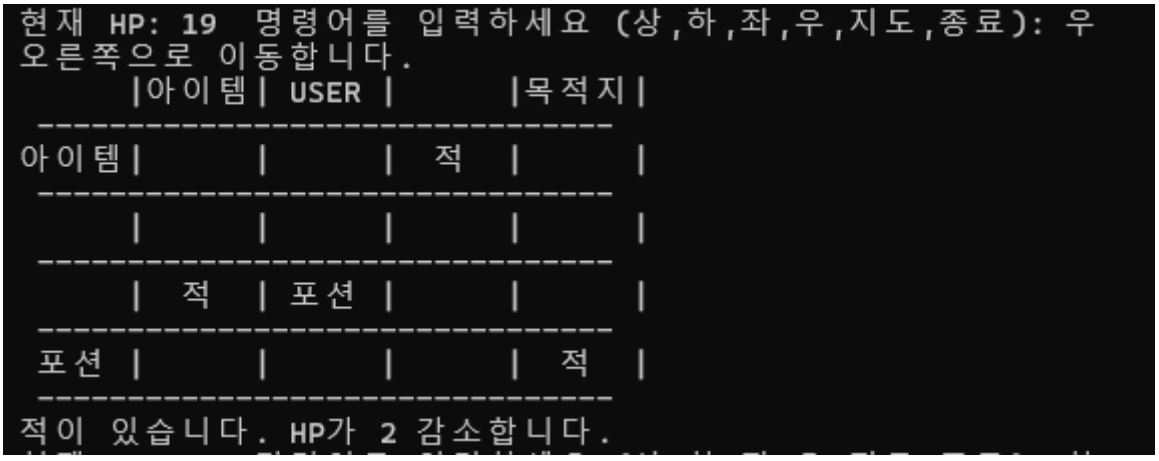
⑥ 처음 명령문을 입력 받을 때 마다 HP 함께 출력

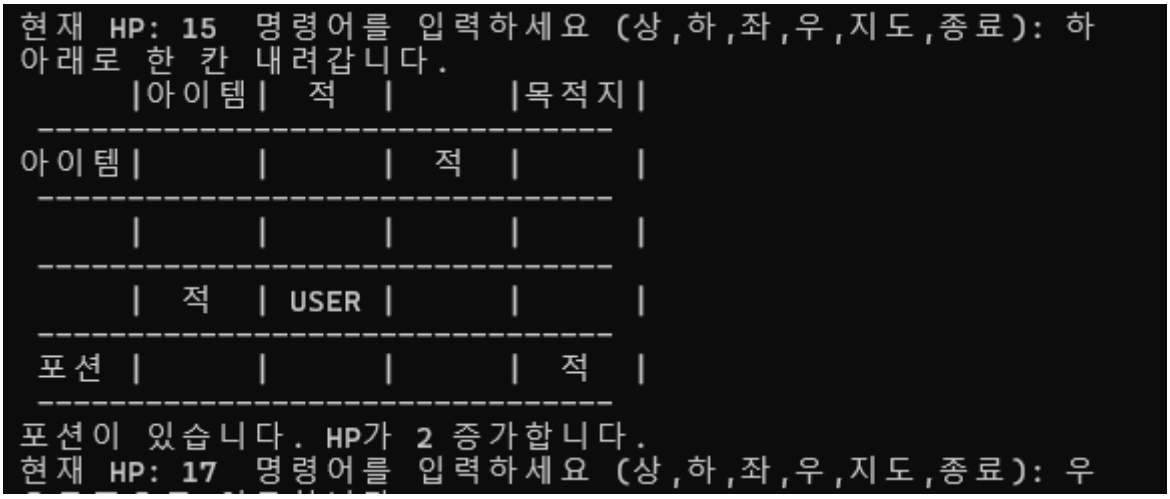
⑦ HP가 0이 되면 “실패”를 출력하고 종료



⑧ 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력






현재 HP: 15 명령어를 입력하세요 (상,하,좌,우,지도,종료): 하
아래로 한 칸 내려갑니다.
|아이템| 적 | |목적지|
아이템| | | 적 | |
| | | | |
| 적 | USER | | |
포션 | | | | 적 |
포션이 있습니다. HP가 2 증가합니다.
현재 HP: 17 명령어를 입력하세요 (상,하,좌,우,지도,종료): 우

2. 최종 테스트 결과:

```
현재 HP: 20  명령어를 입력하세요 (상,하,좌,우,지도,종료): 우
오른쪽으로 이동합니다.
      | USER |  적  |      |목적지|
 ------------------------------
아이템|      |      |  적  |      |
 ------------------------------
      |      |      |      |      |
 ------------------------------
      |  적  | 포션 |      |      |
 ------------------------------
 포션 |      |      |      |  적  |
 ------------------------------
아이템이 있습니다.
현재 HP: 19  명령어를 입력하세요 (상,하,좌,우,지도,종료): 우
오른쪽으로 이동합니다.
      |아이템| USER |      |목적지|
 ------------------------------
아이템|      |      |  적  |      |
 ------------------------------
      |      |      |      |      |
 ------------------------------
      |  적  | 포션 |      |      |
 ------------------------------
 포션 |      |      |      |  적  |
 ------------------------------
적이 있습니다. HP가 2 감소합니다.
현재 HP: 17  명령어를 입력하세요 (상,하,좌,우,지도,종료): 우
오른쪽으로 이동합니다.
      |아이템|  적  | USER |목적지|
 ------------------------------
아이템|      |      |  적  |      |
 ------------------------------
      |      |      |      |      |
 ------------------------------
      |  적  | 포션 |      |      |
 ------------------------------
 포션 |      |      |      |  적  |
 ------------------------------
현재 HP: 16  명령어를 입력하세요 (상,하,좌,우,지도,종료): 우
오른쪽으로 이동합니다.
      |아이템|  적  |      | USER |
 ------------------------------
아이템|      |      |  적  |      |
 ------------------------------
      |      |      |      |      |
 ------------------------------
      |  적  | 포션 |      |      |
 ------------------------------
 포션 |      |      |      |  적  |
 ------------------------------
목적지에 도착했습니다! 축하합니다!
게임을 종료합니다.
```

5. 결과 및 결론

1. 프로젝트 결과: 상하좌우로 이동할 수 있고 지도에 존재하는 적, 포션, 아이템과 상호작용이 되는 간단한 mud게임을 구현했다.

2. 느낀 점: 이런 mud 게임은 내가 게임을 접하기 전에 유행했던 유형의 게임이어서 이런 게임을 구현하는 것이 색다른 경험이었다.