mud 게임 구현 보고서

223445 빅데이터융합학과 김채흔

1. 서론

1-1. 프로젝트 목적 및 배경

- C++의 배열, 반복문, 조건문, 변수 등 배운 개념을 응용해보기 위한 목적
- 응용하여 요구 코드를 구현해보면서 C++ 실력 향상을 위한 목적

1-2. 목표

- mud 게임을 C++를 이용해 구현

2. 요구 사항

2-1. 사용자 요구사항

- 목적지까지 갈 수 있도록 상/하/좌/우 입력하여 움직이기

2-2. 1. 기능 요구사항

(1) 사용자에게 상/하/좌/우/지도/종료 중 하나 입력 받기

- 상/하/좌/우를 입력 받으면 그에 맞게 사용자가 이동
- 지도를 입력 받으면 지도와 현재 사용자 위치 출력
- 종료 입력 받으면 프로그램 종료
- 그 외를 입력 받으면 에러 메시지 출력 후 재입력 요청

(2) 정해져 있는 지도 밖으로 나가게 되면 에러 메시지 출력 후 재입력 요청

(3) 목적지에 도착하면 메시지 출력하고 프로그램 종료

2 . 추가 기능 요구 사항

(1) 사용자가 이동할 때마다 체력 1씩 감소

- 사용자의 기본체력은 20
- 사용자에게 입력 받을 때마다 체력 출력

(2) 체력이 0이 되면 '실패' 출력하고 종료

(3) 아이템/포션/적을 만났을 때 해당 메시지 출력

- 아이템 만났을 때는 아이템을 만났다는 메시지 출력
- 포션을 만났을 때는 포션을 만났다는 메시지와 체력이 2 회복된다는 메시지와 현재 체력 출력
- 적을 만났을 때는 적을 만났다는 메시지와 체력이 2 깎인다는 메시지와 현재 체력 출력

2-3. 함수 계획

- Main 함수 사용자에게 입력 받고 기능을 실행할 수 있도록 함수 호출
- checkXY( ) : 이동하려는 곳이 유효한 좌표인지 체크하는 함수
- checkGoal( ): 사용자의 위치가 현재 목적지인지 체크하는 함수
- checkhp( ): 체력이 0이 되었는 지 체크하는 함수
- checkState( ): 포션/아이템/적을 만났는 지 체크하는 함수
- displayMap( ): 지도와 현재 사용자 위치 출력 함수

3. 설계 및 구현

(1) 사용자에게 상/하/좌/우/지도/종료 중 하나 입력 받기

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

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

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

- 변수
  - user_input = 사용자 입력 저장 변수
- 설명
  - 사용자에게 입력 받은 것을 user_input에 저장

```
if (user_input == "상") {
    // 위로 한 칸 올라가기
    user_y -= 1;
}
else if (user_input == "하") {
    // 아래로 한 칸 내려가기
    user_y += 1;
}
else if (user_input == "좌") {
    // 왼쪽으로 이동하기
    user_x -= 1;
}
else if (user_input == "우") {
    // 오른쪽으로 이동하기
    user_x += 1;
}
```

- 변수
  - user_y = 현재 유저의 세로 위치
  - user_x = 현재 유저의 가로 위치
- 설명
  - 상/하/좌/우를 입력 받으면 그에 맞게 사용자가 이동

```
else if (user_input == "지도") {
    // 지도 보여주기 함수 호출
    displayMap(map, user_x, user_y);
}
else if (user_input == "종료") {
    cout << "종료합니다.";
    break;  // 종료를 입력받으면 종료
}
else {
    cout << "잘못된 입력입니다." << endl;
    continue;
}   // 사용자 입력이 상/하/좌/우/지도/종료 외에 다른 것이라면 잘못된 입력 출력
```

- 변수
  - Map = 지도 변수
- 설명
  - 지도 입력받으면 displaymap( ) 함수 호출
- 결과
  - 지도를 입력 받으면 지도와 현재 사용자 위치 출력
  - 종료 입력 받으면 프로그램 종료
  - 그 외를 입력 받으면 에러 메시지 출력

- 사용 함수 displayMap( )

```
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;
```

- 변수
  - posState = 지도 위치 변수
- 설명
  - 지도 행렬에서 모든 행렬에 대해 반복하며 0은 공백, 1은 아이템, 2는 적, 3은 포션, 4는 목적지를 출력하도록 한다.
- 결과
  - 현재 지도 출력

(2) 정해져 있는 지도 밖으로 나가게 되면 에러 메시지 출력 후 재입력 요청

```
bool inMap = checkXY(user_x, mapX, user_y, mapY);
if (inMap == false) {
    cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
```

- 변수
  - Inmap = 이동하려는 곳이 유효한 좌표인지 체크한 결과 값 변수
- 설명

- checkXY 함수로 지도 밖으로 나갔는 지 체크

- 결과

- 지도 밖으로 나갔다면 에러 메시지 출력하고 사용자 원래대로 복구

- 사용 함수 checkXY( )

```
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;
}
```

- 변수

- checkFlag = 지도 밖으로 나갔는지 여부 체크 변수

- 설명

- checkFlag를 false로 기본 설정을 해놓고, 만약 현재 유저 위치가 정해진 지도 이상으로 벗어났다면 true 값을 반환

(3) 목적지에 도착하면 메시지 출력하고 프로그램 종료

```
// 목적지에 도달했는지 체크
bool finish = checkGoal(map, user_x, user_y);
if (finish == true) {
    cout << "목적지에 도착했습니다! 축하합니다!" << endl;
    cout << "게임을 종료합니다." << endl;  //목적지에 도달했다면 축하메시지 출력 후 종료
    break;
}
```

- 변수

- finish = 목적지에 도달했는지 여부 결과 값 변수

- 설명

- checkGoal 함수로 목적지에 도달했는지 체크

- 결과

- 목적지에 도착하면 축하메시지와 종료 메시지 출력 뒤 프로그램 종료

- 사용 함수 checkGoal( )

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

- 설명
  - 현재 유저의 위치가 목적지(4) 라면 true 값 반환

추가 기능 요구 사항

(1) 사용자가 이동할 때마다 체력 1씩 감소

```
int hp= 20; // 사용자 체력 20 설정
```

```
cout << "체력은 " << hp << "입니다"<<endl; //남은 체력 출력
```

- 변수
  - hp = 사용자 현재 체력 변수
- 설명
  - 사용자의 기본체력을 20으로 설정(전역변수)
- 결과
  - 사용자에게 입력 받을 때마다 현재 체력 출력

(2) 체력이 0이 되면 '실패' 출력하고 종료

```
//체력이 0이 되었는 지 체크
bool hp_check = checkhp(hp);
if (hp_check == true) {
    cout << "실패" << endl;
    cout << "게임을 종료합니다" << endl; //체력이 0이 되면 실패 출력 후 종료
    break;
}
```

- 변수
  - hp_check = 체력이 0이 되었는 지 여부 결과값 변수
- 설명
  - checkhp( ) 함수 활용하여 체력이 0이 되었는 지 체크

- 결과

  - 체력이 0이 되었다면 실패와 종료 메시지 출력 후 프로그램 종료

- 사용 함수 checkhp( )

```
// 체력을 다 했는 지 체크하는 함수
bool checkhp(int hp) {
    if (hp <= 0) {  //체력이 0이히면 true 반환
        return true;
    }
    return false;
}
```

- 설명

  - 체력이 0 이하면 true 반환, 그렇지 않으면 false 반환

(3) 아이템/포션/적을 만났을 때 해당 메시지 출력

```
checkState(map, user_x, user_y); //적, 포션, 아이템 만났을 때 함수
```

- 설명

  - checkState( ) 함수 활용해서 적, 포션, 아이템 만났는 지 체크

- 사용 함수 checkState( )

```
void checkState(int map[][mapX], int user_x, int user_y) {
            int posState = map[user_y][user_x];
                switch (posState) {
                case 1:
                    cout << "아이템이 있습니다"<<endl;
                    break;
                case 2:
                    cout << "적이 있습니다. 체력이 2만큼 깎입니다."<<endl;
                    hp = hp - 2;  // 체력 2 깎기
                    cout << "체력은 " << hp <<"입니다."<<  endl;
                    break;
                case 3:
                    cout << " 포션이 있습니다. 체력이 2만큼 회복됩니다."<<endl;
                    hp = hp + 2; //체력 2 회복
                    cout << "체력은 " << hp << "입니다."<< endl;
                    break;
                }  // 유저 위치에 아이템, 적, 포션 이 있으면 이를 출력
        }
```

- 변수

  - posState = 현재 유저 위치의 값(0,1,2,3,4,)

- 설명

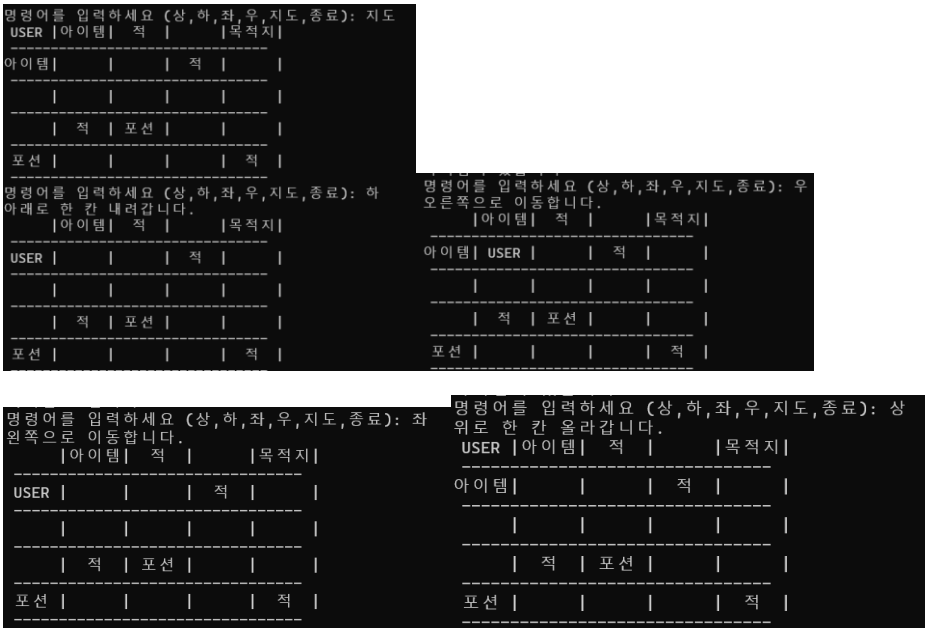
- 적을 만나면 체력 2 감소
- 포션 만나면 체력 2 증가

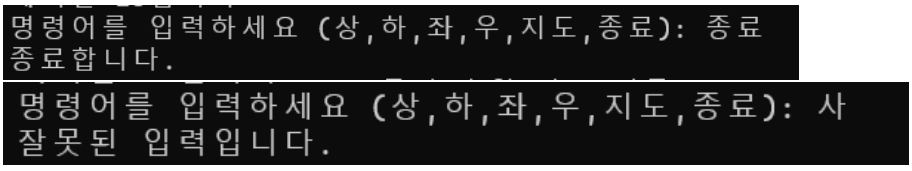
- 결과
  - 아이템 만났을 때는 아이템을 만났다는 메시지 출력
  - 포션을 만났을 때는 포션 만났다는 메시지와 체력이 2 회복된다는 메시지와 함께 현재 체력 출력
  - 적을 만났을 때는 적을 만났다는 메시지와 체력이 2 깎인다는 메시지와 함께 현재 체력 출력

4. 테스트

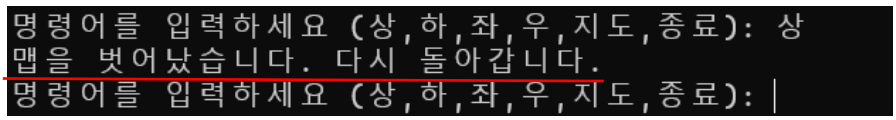
4-1. 기능별 테스트 결과

(1) 사용자에게 상/하/좌/우/지도/종료 중 하나 입력 받기

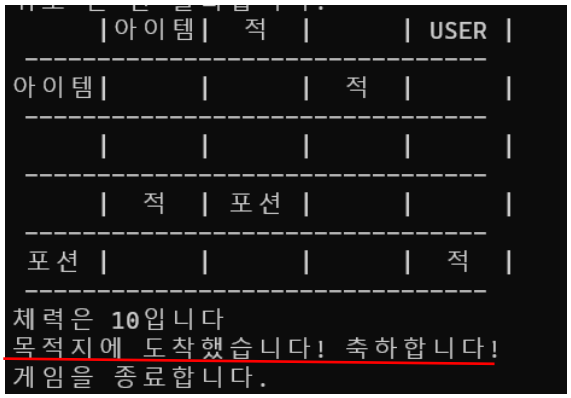




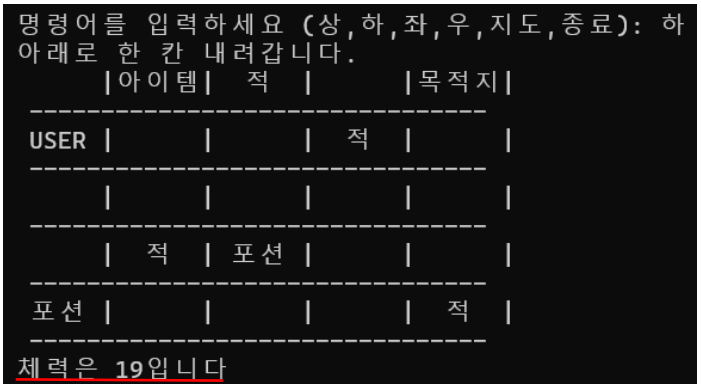
(2) 정해져 있는 지도 밖으로 나가게 되면 에러 메시지 출력 후 재입력 요청



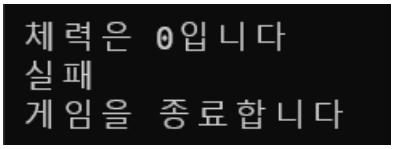
(3) 목적지에 도착하면 메시지 출력하고 프로그램 종료



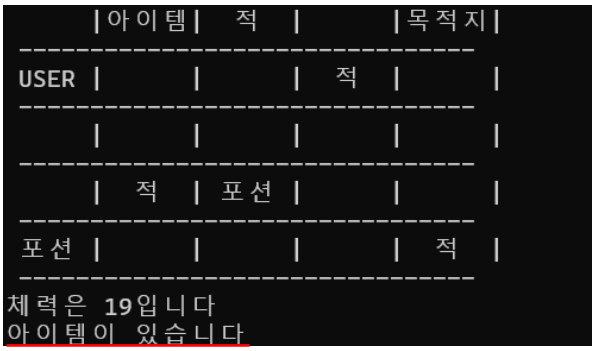
(4) 사용자가 이동할 때마다 체력 1씩 감소

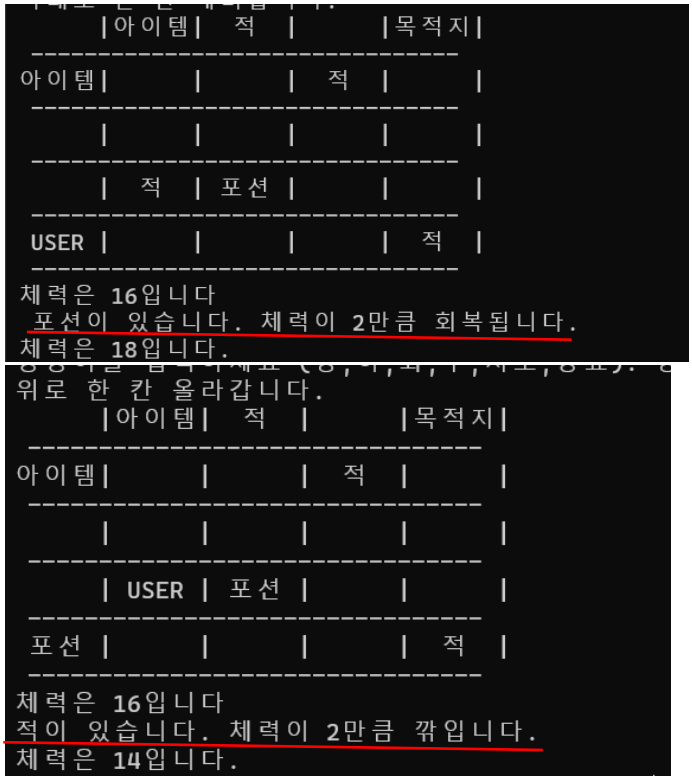


(5) 체력이 0이 되면 '실패' 출력하고 종료

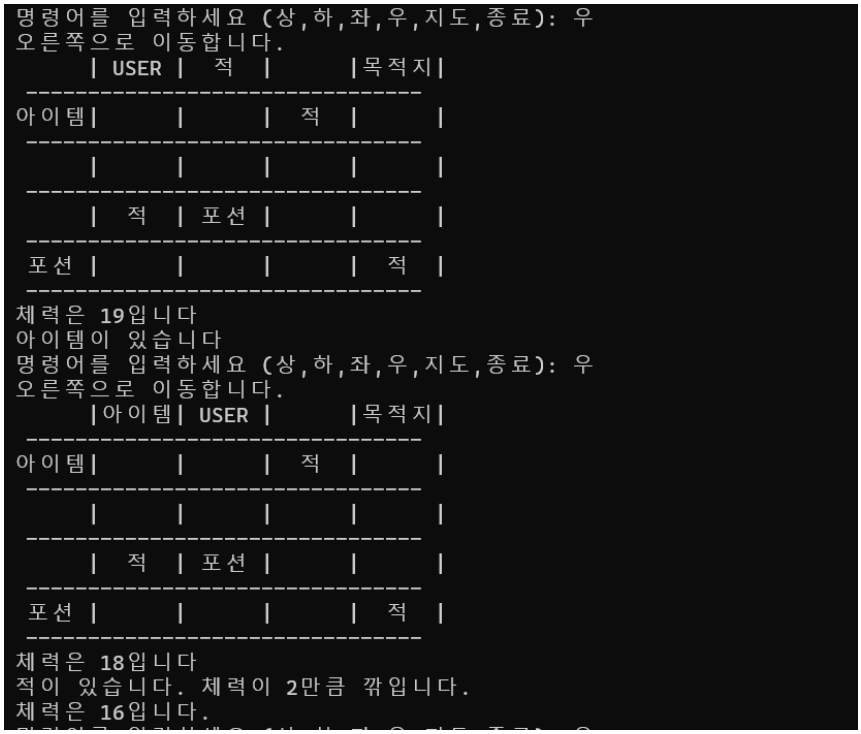


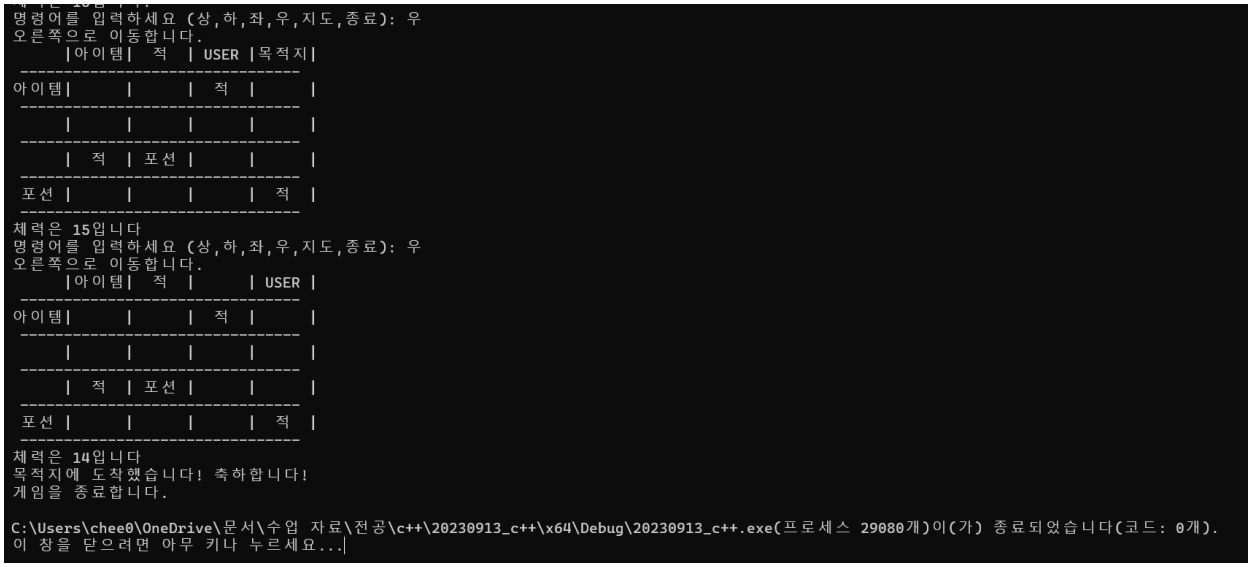
(6) 아이템/포션/적을 만났을 때 해당 메시지 출력





4-2.최종 테스트 스크린샷





5. 결과 및 결론

5-1. 프로젝트 결과

- Mud 게임을 C++로 구현하였음.

5-2. 소감

- 저번 프로젝트 실습 보다는 코드를 이해하고 적는 데 조금 더 수월했던 것 같습니다. 코드를 함수로 표현하고 호출하는 부분이 조금 어려웠습니다.