## C++프로그래밍및실습

# Mud Game

제출일자:

제출자명: 범지성

제출자학번: 214930

# 1. 프로젝트 목표

## 1) 배경 및 필요성

7주차까지 배운 내용에 대한 실습을 위해 진행

## 2) 프로젝트 목표

간단한 Mud 게임 구현

# 2. 요구사항

## 1) 사용자 요구사항

유저가 상하좌우로만 이동하며 목적지에 도착한다

## 2-1) 기능 요구사항

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

- 상/하/좌/우 입력 시 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) 유저는 체력 20을 가지고 게임 시작

3) 처음 명령문을 입력 받을 때 마다 HP 함께 출력

4) 지도 밖으로 나가게 되면 에러 메시지 출력

5) 목적지에 도착하면 "성공"을 출력하고 종료

6) 사용자가 이동할 때 마다 사용자 체력 1씩 감소

7) HP가 0이 되면 "실패"를 출력하고 종료

8) 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력

### 2-2) 함수 계획

1) 메인 함수: 사용자에게 값을 계속 입력 받고, 그에 대한 함수 호출

2) 지도와 현재 위치 출력 함수: displayMap()

3) 사용자 위치 체크 함수: checkXY()

4) 목적지에 도착 체크 함수: checkGoal()

5) 잔여 체력 0이하 체크 함수: checkHP()

## 3-1. 기능 구현

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

- 변수

① user_input : 유저의 입력값

② hp : 유저의 체력

- 설명

명령을 입력하라는 지시문을 출력하며 사용자 입력 대기

- 코드 스크린샷

```
38      // 사용자의 입력을 저장할 변수
39      string user_input = "";
40
41      cout << "현재 HP: " << hp << " 명령어를 입력하세요 (상,하,좌,우,지도,종료): ";
42      cin >> user_input;
```

#### (2) 유저는 체력 20을 가지고 게임 시작

- 설명

유저의 체력을 초기값 20으로 변수에 저장해 추후 로직에 사용한다.

- 코드 스크린샷

```
int hp = 20; // 유저 체력
```

### (3) 처음 명령문을 입력 받을 때 마다 HP 함께 출력

- 변수

① hp : 유저의 체력

- 설명

명령문 입력하라는 안내 메시지의 맨 앞에 현재 체력 변수값을 출력한다.

- 코드 스크린샷

```
41          cout << "현재 HP: " << hp << " 명령어를 입력하세요 (상,하,좌,우,지도,종료): ";
```

### (4) 지도 밖으로 나가게 되면 에러 메시지 출력

- 변수

① checkFlag : 사용자의 이동 시도가 정상적인지 확인하는 변수

- 설명

checkFlag가 false로 맵을 벗어났음을 가리키면 이 사실을 출력해 알림.

- 코드 스크린샷

```
170     if(checkFlag == false)
171         cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
172     // 이동 가능한 경우
```

### (5) 목적지에 도착하면 “성공”을 출력하고 종료

- 입력

① map : 맵 정보

② user_x : 유저의 x좌표

③ user_y : 유저의 y좌표

④ finish : 목적지 도달 여부(참, 거짓)

- 설명

유저가 목적지에 도달했는지 checkGoal 함수(함수 구현 참조)로 판별한 후 finish가 true가 되면 목적지에 도착했음을 알리고 게임이 종료된다.

- 코드 스크린샷

```
114         bool finish = checkGoal(map, user_x, user_y);
115         if (finish == true) {
116             cout << "목적지에 도착했습니다! 축하합니다!" << endl;
117             cout << "게임을 종료합니다." << endl;
118             break;
119         }
```

### (6) 사용자가 이동할 때 마다 사용자 체력 1씩 감소

- 변수

① checkFlag : 사용자의 이동 시도가 정상적인지 확인하는 변수

② hp : 유저의 체력

- 설명

checkFlag가 true면 사용자가 이동을 성공했음을 의미한다. 따라서 이 경우에 체력이 1 감소하도록 한다.

- 코드 스크린샷

```
170     if(checkFlag == false)
171         cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
172     // 이동 가능한 경우
173     else
174         hp--;
```

### (7) HP가 0이 되면 “실패”를 출력하고 종료

- 변수

① hp : 유저의 체력

- 설명

checkHP(함수 구현 참조) 결과가 false면 게임 루프를 종료한다.

- 코드 스크린샷

```
105         // 체력이 고갈됐는지 체크
106         if(!(checkHP(hp))){
107             break;
108         }
```

### (8) 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력

- 입력

① hp : 유저의 체력

② map : 맵 정보

③ user_x : 유저의 x좌표

④ user_y : 유저의 y좌표

- 설명(함수 로직 전체가 기능이기에 함수 구현 항목에서 제외)

유저의 현 위치에 아이템/적/포션이 있으면 해당 사실 안내문구 출력. 적/포션을 만난 경우 hp 2 감소/증가 후 해당 사실 출력. switch 구문으로 아이템/적/포션을 만난 경우를 나누고 아무것도 만나지 않은 경우 아무것도 출력되지 않는다.

- 코드 스크린샷

```
188 void checkState(int map[][mapX], int user_x, int user_y, int &hp){
189     int encounter = map[user_y][user_x];
190     switch(encounter){
191         case 1:
192             cout << "아이템이 있습니다" << endl;
193             break;
194         case 2:
195             cout << "적이 있습니다. HP가 2 줄어듭니다." << endl;
196             hp -= 2;
197             break;
198         case 3:
199             cout << "포션이 있습니다. HP가 2 증가합니다." << endl;
200             hp += 2;
201             break;
202     }
203 }
```

## 3-2. 함수 구현

### (1) main()

- 입력 및 전역변수

① mapX: 맵의 가로 크기

② mapY: 맵의 세로 크기

- 지역변수

① hp : 유저의 체력

② map : 맵 정보

③ user_x : 유저의 x좌표

④ user_y : 유저의 y좌표

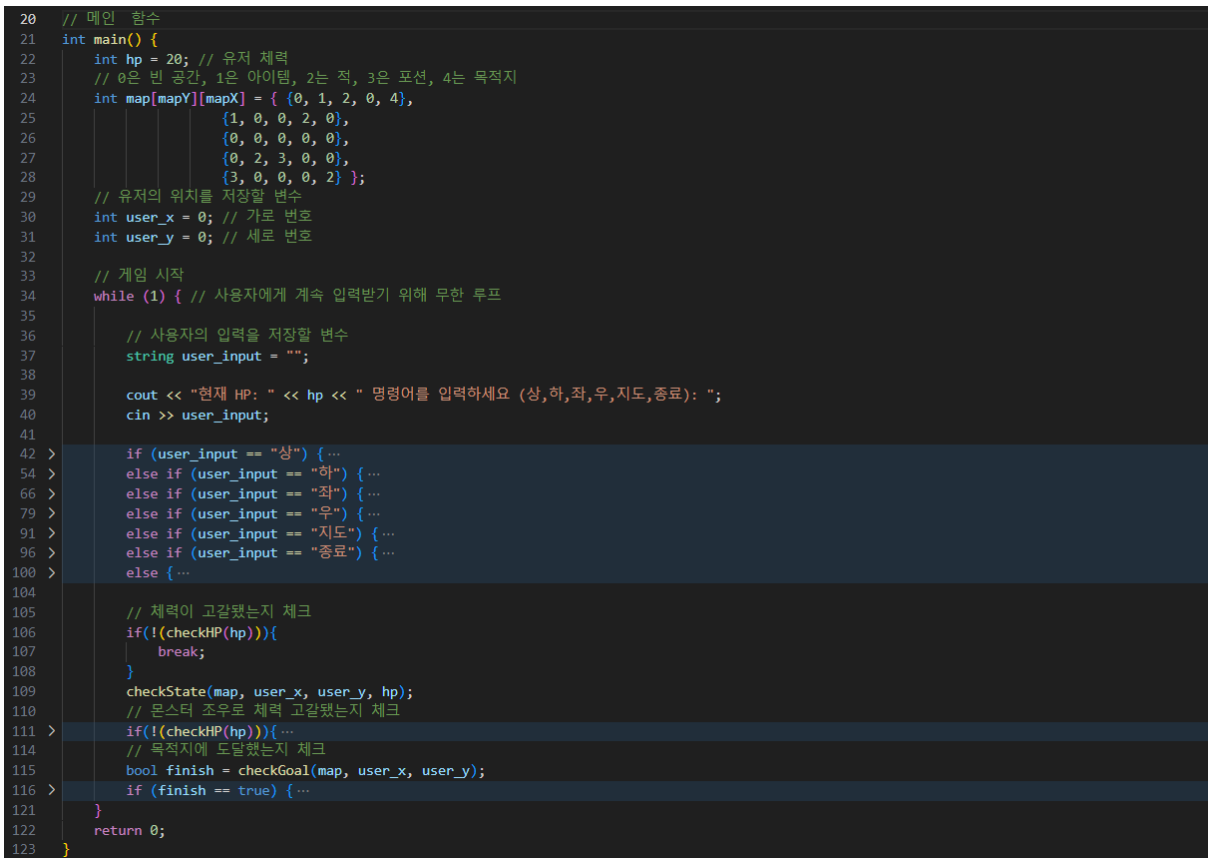
⑤ finish : 목적지 도달 여부(참, 거짓)

⑥ user_input : 유저의 입력값

- 설명

함수가 시작되면 맵 생성, 체력 20 초기화, 유저 초기좌표 (0, 0)으로 설정된다. 이후 while(1)로 무한 루프 구문을 생성하고 해당 블록 속에서 다음 내용이 반복된다. 유저의 잔여 체력과 명령 입력 필요를 알리며 유저 명령을 받는다. 유저 명령에 따라 조건문으로 나뉜 유저 좌표 이동 및 출력, 지도 출력, 루프문 탈출 중 하나가 실행된다. 종료가 아닌 유저 명령이 시행되면 잔여 체력이 0이하인지 체크해 0이하면 루프문을 탈출한다. 체력 체크 이후 유저가 만난 요소에 따라 안내 메시지와 체력 변화가 발생한다. 적 조우로 체력이 0 이하가 되는 경우도 있어 해당 로직 이후 체력 검사를 다시 시행한다. 이후 목적지 도착 여부를 체크한다. 여기까지 break가 실행이 안 됐으면 다시 루프문 처음으로 돌아간다.

- 코드 스크린샷 (기능 구현, 타함수 구현에 구체적으로 명시된 코드는 축약 표시)



```
// 메인 함수
int main() {
    int hp = 20; // 유저 체력
    // 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

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

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

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

        if (user_input == "상") {…
        else if (user_input == "하") {…
        else if (user_input == "좌") {…
        else if (user_input == "우") {…
        else if (user_input == "지도") {…
        else if (user_input == "종료") {…
        else {…

        // 체력이 고갈됐는지 체크
        if(!(checkHP(hp))){
            break;
        }
        checkState(map, user_x, user_y, hp);
        // 몬스터 조우로 체력 고갈됐는지 체크
        if(!(checkHP(hp))){…
        // 목적지에 도달했는지 체크
        bool finish = checkGoal(map, user_x, user_y);
        if (finish == true) {…
    }
    return 0;
}
```

## (2) displayMap()

- 입력 및 전역변수

① mapX: 맵의 가로 크기

② mapY: 맵의 세로 크기

③ map : 맵 정보

④ user_x : 유저의 x좌표

⑤ user_y : 유저의 y좌표

- 설명

이중 반복문으로 map의 요소 하나 하나를 출력한다. 이 때 유저 위치에 해당하는 맵 요소는 읽지 않고 "USER"가 출력된다. 맵 요소를 읽어 switch문으로 공백/"아이템"/"적"/"포션"/"목적지"를 구분해서 출력한다.

- 코드 스크린샷

```
126  // 지도와 사용자 위치 출력하는 함수
127  void displayMap(int map[][mapX], int user_x, int user_y) {
128      for (int i = 0; i < mapY; i++) {
129          for (int j = 0; j < mapX; j++) {
130              if (i == user_y && j == user_x) {
131                  cout << " USER |"; // 양 옆 1칸 공백
132              }
133              else {
134                  int posState = map[i][j];
135                  switch (posState) {
136                  case 0:
137                      cout << "      |"; // 6칸 공백
138                      break;
139                  case 1:
140                      cout << "아이템|";
141                      break;
142                  case 2:
143                      cout << "  적  |"; // 양 옆 2칸 공백
144                      break;
145                  case 3:
146                      cout << " 포션 |"; // 양 옆 1칸 공백
147                      break;
148                  case 4:
149                      cout << "목적지|";
150                      break;
151                  }
152              }
153          }
154          cout << endl;
155          cout << " ------------------------------ " << endl;
156      }
157  }
```

### (3) checkXY()

- 입력 및 전역변수

① mapX: 맵의 가로 크기

② mapY: 맵의 세로 크기

③ hp : 유저의 체력

④ map : 맵 정보

⑤ user_x : 유저의 x좌표

⑥ user_y : 유저의 y좌표

- 지역변수

① checkFlag : 사용자의 이동 시도가 정상적인지 확인하는 변수

- 설명

입력받은 유저의 x, y 좌표가 맵의 범위 내에 있으면 return할 checkFlag를 true로 아니면 false로 한다. 이동 실패 시의 경고 메시지 출력과 이동 성공 시 체력 감소 로직도 해당 메소드 안에 존재한다.

- 코드 스크린샷

```
159 // 이동하려는 곳이 유효한 좌표인지 체크하는 함수
160 // 경고 메시지 출력 내부화
161 // 이 함수가 true 반환 시 이동 성공 따라서 해당 로직에서 체력 감소 가능
162 bool checkXY(int user_x, int mapX, int user_y, int mapY, int &hp) {
163     bool checkFlag = false;
164     if (user_x >= 0 && user_x < mapX && user_y >= 0 && user_y < mapY) {
165         checkFlag = true;
166     }
167     // 맵 벗어나는 시도인 경우
168     if(checkFlag == false)
169         cout << "맵을 벗어났습니다. 다시 돌아갑니다." << endl;
170     // 이동 가능한 경우
171     else
172         hp--;
173     return checkFlag;
174 }
```

**(4) checkGoal()**

- 입력 및 전역변수

① mapX: 맵의 가로 크기

② map : 맵 정보

③ user_x : 유저의 x좌표

④ user_y : 유저의 y좌표

- 설명

유저가 이동해서 조우한 요소가 목적지이면 true를 아니면 false를 반환한다.

- 코드 스크린샷

```
176  // 유저의 위치가 목적지인지 체크하는 함수
177  bool checkGoal(int map[][mapX], int user_x, int user_y) {
178      // 목적지 도착하면
179      if (map[user_y][user_x] == 4) {
180          return true;
181      }
182      return false;
183  }
```

**(5) checkHP()**

- 입력 및 전역변수

① hp : 유저의 체력

- 지역변수

② isAlive: 유저 생존 여부 변수

- 설명

유저의 잔여 체력을 입력 받아 0이하면 실패 메시지와 종료 안내문을 출력하고 false 반환. 아니면 true 반환

- 코드 스크린샷

```
202  // 유저 잔여 체력 0이하인지 체크하는 함수
203  bool checkHP(int &hp){
204      bool isAlive = true;
205      if(hp <= 0){
206              cout << "HP가 0 이하가 되었습니다. 실패했습니다." << endl;
207              cout << "게임을 종료합니다." << endl;
208              isAlive = false;
209      }
210      return isAlive;
211  }
```

# 4-1. 기능별 테스트 결과

**(1) 사용자에게 “상”, “하”, “좌”, “우”, “지도”, “종료” 중 하나를 입력 받기, 유저 초기 체력 20, 명령문 입력 받을 때 잔여체력 출력**

- 테스트 결과 스크린샷

```
현재 HP: 20 명령어를 입력하세요 (상,하,좌,우,지도,종료):
```

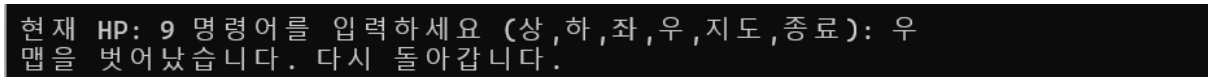
- 설명

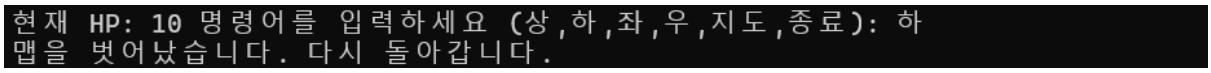
유저 체력과 명령 입력 지시문이 출력되고 유저 입력대기 버퍼 생성됨.

프로그램 실행 후 최초 출력된 유저 체력으로 20임을 확인할 수 있다.

### (2) 지도 밖으로 나가게 되면 에러 메시지 출력

- 테스트 결과 스크린샷



- 설명

상하좌우 모든 이동에서 맵 벗어나는 시도 시 에러 메시지 출력됨.

### (3) 목적지에 도착하면 “성공”을 출력하고 종료

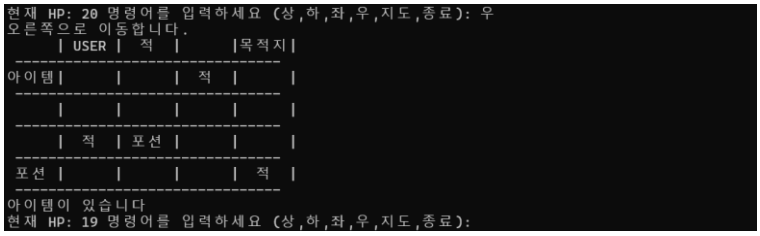
- 테스트 결과 스크린샷

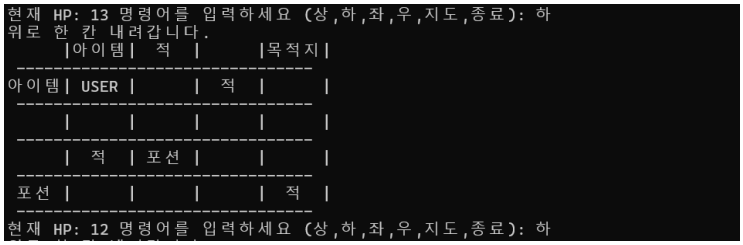

- 설명

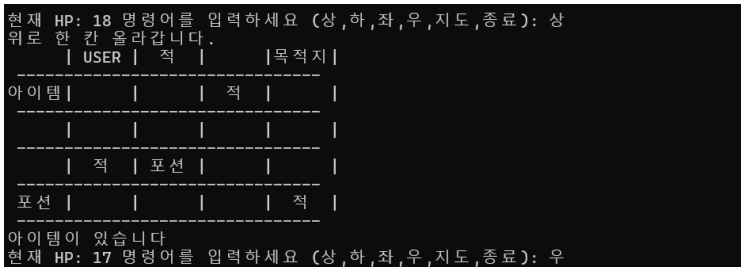
유저가 목적지에 도착했을 때 안내 메시지 출력 후 게임 종료됨.

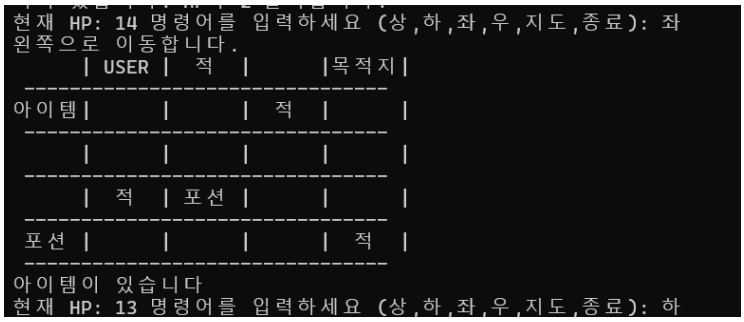
### (4) 사용자가 이동할 때 마다 사용자 체력 1씩 감소

- 테스트 결과 스크린샷



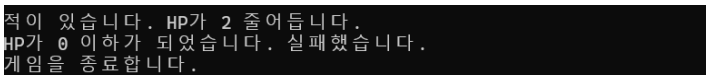




- 설명

  상하좌우 이동에서 체력이 1씩 감소됨을 확인

## (5) HP가 0이 되면 “실패”를 출력하고 종료

- 테스트 결과 스크린샷





- 설명

이동으로 체력이 고갈되거나 적 조우로 체력이 고갈되면 다음 입력은 받지 않고 체력이

0이하로 실패임을 고지하고 게임을 종료한다.

### (6) 무기/갑옷, 포션, 적을 만났을 때 그에 대한 메시지를 출력

- 테스트 결과 스크린샷

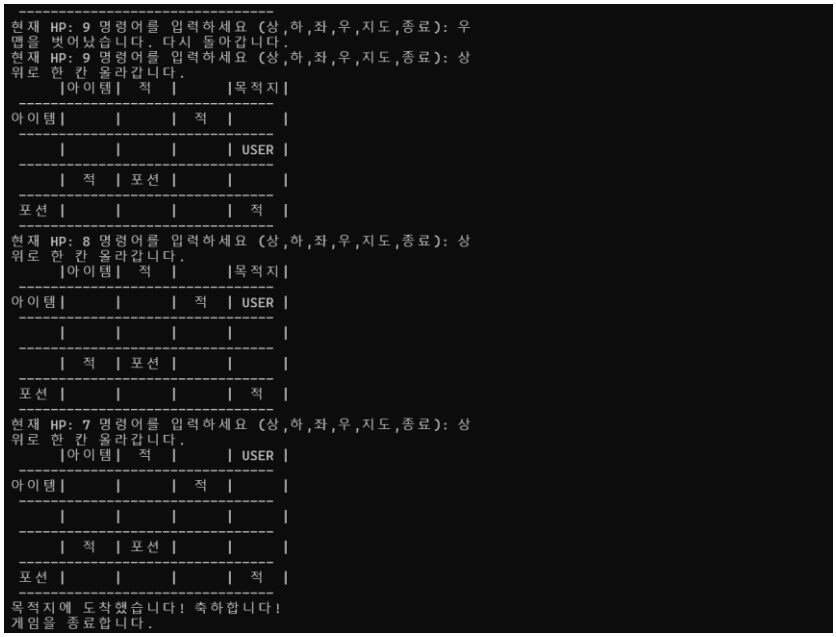



- 설명

아이템을 만나면 조우 사실 출력. 포션 만나면 조우 사실 출력 및 체력 2증가 (증가 2 이동 -1로 1증가). 적 만나면 조우 사실 출력 및 체력 2감소 (이동 + 적 조우 3 감소)

## 4-2. 최종 테스트 결과

- 테스트 결과 스크린샷




- 설명

유저가 상/하/좌/우 명령을 입력해 아이템, 적, 포션을 조우하며 이동하다가 목적지에 도달한다.

# 5. 결과 및 결론

## 1) 프로젝트 결과

- 유저 이동, 목적지 존재, 소수 적과 아이템이 존재하는 간단한 형태의 mud game을 만들었다.

## 2) 느낀 점

- week 5의 프로젝트 Tic Tac Toe 때와는 달리 함수, string 클래스를 배우고 코드를 작성했지만 크게 다른 느낌이 없었다. 그래서 어떤 라이브러리를 사용하고 어떤 프로그램을 만들든 프로그래밍을 할 때 중요한 것은 알고리즘 최적화를 위해 조건문과 반복문을 잘 이용하는 것이라는 걸 체감할 수 있었다.