# CPP\_week8 프로젝트 보고서

213832 성세현

### 1. 서론

- 1.1. 프로젝트 목적 : 프로젝트 실습
- 1.2. 목표 : mud.game 완성 및 기능 추가

#### 2. 요구사항

- 2.1. 사용자 요구사항 : 사용자는 상하좌우로 움직여 제한된 체력 내에 목적지에 도달하는 게임
- 2.2. 기능 계획
  - 2.2.1. 사용자에게 상하좌우 중 하나를 입력 받아 user 이동
  - 2.2.2. 지도와 사용자 위치 출력
  - 2.2.3. 지도 밖으로 나가면 에러 메시지 출력
  - 2.2.4. 목적지에 도달하면 성공 메시지 출력 후 종료
  - 2.2.5. 체력 20으로 시작, 이동할 때 마다 체력 1씩 감소
  - 2.2.6. 체력이 0이되면 사망 메시지 출력 후 종료
  - 2.2.7. 포션, 적 만났을 때 체력 증가, 감소

### 2.3. 함수 계획

- 2.3.1. 지도와 사용자 위치 출력하는 함수
- 2.3.2. 이동하려는 곳이 유효한지 검사하는 함수
- 2.3.3. 유저의 위치가 목적지인지 체크하는 함수
- 2.3.4. 유저의 위치와 적, 표션위치를 체크하여 체력 증감 함

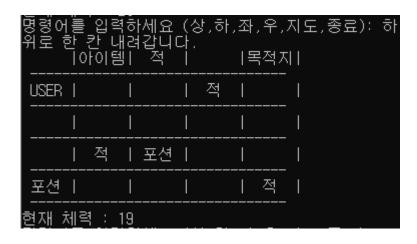
# 3. 설계 및 구현

## 3.1. 스크린 샷

```
if (user_input — "&") {
    // 위로 한 한 급리가기
    user_y -- 1;
    bool inMap - checkY(user_x, mapX, user_y, mapY);
    if (inMap = faisə) {
        cout << "점을 벙어났습니다. 다시 돌아갑니다." << end1;
        user_y += 1;
    }
    else {
        cout << "위로 한 한 급리가기
        user_y += 1;
    }
    else if (user_input — "하") {
        // TODD: chale 한 한 내려간기
        user_y += 1;
        bool inMap - checkY(user_x, mapX, user_y, mapY);
        if (inMap = faisə) {
            cout << "함을 벙어났습니다. 다시 돌아갑니다." << end1;
            displeyMap(map, user_x, user_y);
            pp-;
        }
    }
    slas {
        cout << "위로 한 칸 내려갑니다." << end1;
        displeyMap(map, user_x, user_y);
        if (inMap = faisə) {
            cout << "함을 벙어났습니다. 다시 돌아갑니다." << end1;
            displeyMap(map, user_x, mapX, user_y, mapY);
        if (inMap = faisə) {
            cout << "함을 벙어났습니다. 다시 돌아갑니다." << end1;
            user_x += 1;
        }
        else {
            cout << "원론으로 이용하기
            user_x, user_y);
            pp-:
        }
    }
    plas if (user_input — "주") {
            / TODD: 모든목으로 이용하기
            user_x += 1;
            bool inMap - checkY(vser_x, mapX, user_y, mapY);
        if (inMap = faisə) {
            cout << "원론으로 이용하기
            user_x += 1;
            bool inMap - checkY(vser_x, mapX, user_y, mapY);
        if (inMap = faisə) {
            cout << "함을 벗어났습니다." << end1;
            user_x += 1;
            bool inMap - checkY(user_x, mapX, user_y, mapY);
        if (inMap = faisə) {
            cout << "원론으로 이용하기
            user_x += 1;
            bool inMap - checkY(user_x, mapX, user_y);
        if (inder_m = faisə) {
            cout << "원론으로 이용하기
            user_x += 1;
            bool inMap - checkY(user_x, mapX, user_y);
        if (inder_m = faisə) {
            cout << "원론으로 이용하기
            user_x -= 1;
        }
```

## 3.1.1. 입력

- 3.1.1.1. "상", "하". "좌", "우"
- 3.1.2. 반환값
  - 3.1.2.1. user\_x, user\_y
- 3.1.3. 결과



- 3.1.4. 설명
  - 3.1.4.1. "상", "하", "좌", "우" 중 하나의 값을 받아 방향에 맞는 값을 user\_x, user\_y에 전달한다.
- 3.2. 지도와 사용자 위치 출력
  - 3.2.1. 스크린 샷

```
// 지도와 사용자 위치 출력하는 함수
|void displayMap(int map[][mapX], int user_x, int user_y) {
       for (int j = 0; j < mapX; j++) {
           if (i == user_y && j == user_x) {
: cout << " USER I"; // 양 옆 1칸 공백
               int posState = map[i][j];
               switch (posState) {
               case 0:
                   cout << " I"; // 6칸 공백
                   break;
               case 1:
                   cout << "아이템!";
                   break;
                  cout << " 적 I"; // 양 옆 2칸 공백
                 break;
               case 3:
                   cout << " 포션 I"; // 양 옆 1칸 공백
                   break;
               case 4:
                   cout << "목적지1";
       cout << endl;
       cout << "
```

#### 3.2.2. 입력

3.2.2.1. map, user\_y, user\_x

#### 3.2.3. 출력

3.2.3.1. 현재 게임 맵 지도와 user, 포션, 적, 아이템 목 적지의 위치를 표시한다.

#### 3.2.4. 결과

	0	FOLE	ijĪ	'적'	-i		=	록적 지	П
USER	1		1		I	적	I		Ī
					١		I		I
		적		포션			I		I
포션 			I					적 	I

- 3.2.5. 설명
  - 3.2.5.1. For문을 통해 기본적인 map을 구성한다.
  - 3.2.5.2. Map의 각 공간 안에는 배열에 포함된 값에 대응하는 객체를 삽입한다.
- 3.3. 지도 밖으로 나가면 에러메시지 출력
  - 3.3.1. 이동하려는 곳의 좌표를 검사하고 불가능할 경우 에러 메시지 출력

```
// 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;
|}
```

- 3.3.2. 입력
  - 3.3.2.1. map, user\_y, user\_x
- 3.3.3. 반환값
  - 3.3.3.1. chekFlag : 해당 좌표의 유효성을 전달
- 3.3.4. 결과

3.3.5. 설명

- 3.3.5.1. user가 이동하려는 좌표의 인덱스와 map에 지정 된 5X5의 인덱스를 비교하여 맵을 벗어나는지 확인
- 3.3.5.2. 맵을 벗어난다면 에러 메시지 출력
- 3.4. 목적지 도달 시 성공 메시지 출력
  - 3.4.1. 목적지의 배열 인덱스와 현재 user의 인덱스가 같다면 게임을 종료한다.

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

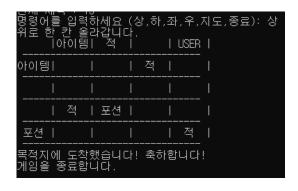
3.4.2. 입력

3.4.2.1. map, user\_x, user\_y

3.4.3. 반환값

3.4.3.1. 없음

3.4.4. 결과



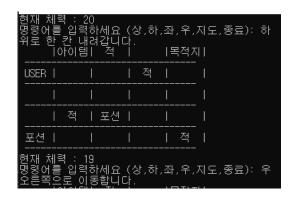
- 3.4.5. 설명
  - 3.4.5.1. 목적지의 배열 인덱스 값과 현재 user\_x, user\_y 로 이루어진 배열 인덱스 값이 맞다면 게임 종료

- 3.5. 체력 20으로 시작, 이동 시 체력 감소
  - 3.5.1. 최초 체력은 20으로 시작하며 이동 시 체력이 1씩 감소한다.

```
//유저의 체력을 저장할 변수
int hp = 20;
```

```
else {
cout << "위로 한 칸 올라갑니다." << endl;
displayMap(map, user_x, user_y);
hp--;
}
```

- 3.5.2. 입력
  - 3.5.2.1. 없음
- 3.5.3. 반환값
  - 3.5.3.1. hp, 현재 체력을 나타내는 출력
- 3.5.4. 결과

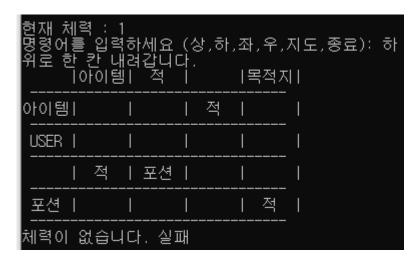


- 3.5.5. 설명
  - 3.5.5.1. "상", "하", "좌", "우" 중 하나의 입력을 받아 정상 적으로 user가 이동하였다면 hp--
  - 3.5.5.2. while문의 조건을 hp > 0으로 설정하여 hp가 0 보다 작아진다면 자동으로 반복문 탈출

- 3.6. 체력 소진 시 사망
  - 3.6.1. 체력이 0이 되면 사망 메시지와 함께 게임 종료

```
// 게임이 끝났는지 체크
if (hp <= 0) {
; cout << "체력이 없습니다. 실패" << endl;
}
```

- 3.6.2. 입력
  - 3.6.2.1. 없음
- 3.6.3. 반환값
  - 3.6.3.1. 체력 소진 시 사망 메시지
- 3.6.4. 결과



- 3.6.5. 이동할 때마다 hp가 1씩 작아짐
- 3.6.6. hp가 0보다 작아지면 체력고갈 메시지 출력
- 3.7. 포션에 도달시 체력 증가, 적에 도달 시 체력 감소
  - 3.7.1. 포션에 도달 시 체력이 2 증가하며 적과 만날 시에는 체력이 2 감소한다.

```
gstring checkState(int map[mapV][mapX], int user_x, int user_y) {
    string value;

if (map[user_y][user_x] == 2) {
    cout << "적을 만났습니다. 체력이 감소합니다." << endl;
    value = "enemy";
  }

else if (map[user_y][user_x] == 3) {
    cout << "포션을 먹었습니다.. 체력이 증가합니다." << endl;
    value = "poshon";
  }

return value;
```

```
// 포션, 적을 만났는지 체크, hp감소, 증가
string value = checkState(map, user_x, user_y);
if (value == "enemy")
hp = hp - 2;
else if (value == "poshon")
hp = hp + 2;
```

- 3.7.2. 입력
  - 3.7.2.1. cap, user\_y, user\_x
- 3.7.3. 반환값
  - 3.7.3.1. hp, 적 또는 포션과 조우했다는 메시지
- 3.7.4. 결과



- 3.7.5. 설명
  - 3.7.5.1. User 위치의 배열 인덱스값과 적, 포션의 인덱스 값을 비교한다
  - 3.7.5.2. 적과 위치가 같으면 hp = hp-2
  - 3.7.5.3. 포션과 위치가 같으면 hp = hp + 2

# 4. 테스트

- 4.1. 기능별 결과는 요구사항에 첨부
- 4.2. 최종 테스트

```
11박 : 20
를 입력하세요 (상,하,좌,우,지도,종료): 하
반 칸 내려갑니다.
|아이템| 적 | |목적지|
    체력 : 18
거를 입력하세요 (상,하,좌,우,지도,종료): 우
목으로 이동합니다.
| 아이템| 적 | 목적지|
    체력 : 17
어를 입력하세요 (상,하,좌,우,지도,종료): 우
쪽으로 이동합니다.
|아이템| 적 | 목적지|
    만났습니다. 체력이 감소합니다.
체력 : 14
      세력 : 14
를 입력하세요 (상,하,좌,우,지도,종료): 우
으로 이동합니다.
|아이템| 적 | |목적지|
     | 적 | 포션 |
포션 | | | | 적 |
-----
목적지에 도착했습니다! 축하합니다!
게임을 중료합니다.
 #lkers#2#source#repos#week8#x64#Debug#week8.exe(프로세스 10828개)이(가) 종료되었습니다(코드: 0개).
| 창울 달으려면 아무 키나 누르세요....
```

## 5. 결과 및 결론

- 5.1. 프로젝트 결과 : 여러 기능이 추가된 mud.game을 최종 완성 하였다.
- 5.2. 느낀 점 : 프로그램 구현은 생각보다 어렵지 않았지만 체계적으로 보고서를 작성하는것에 어려움을 느꼈고 보다 연습하여보기좋고 깔끔한 보고서를 작성하는 연습을 해야할 것 같다.