

# cpp\_day1 과제 보고서

# 목차

1. 과제 1번 코드 설명

2. 과제 2번 코드 설명

3. 과제 3번 코드 설명

## 1. 과제 1번 코드 설명

```
1  #include <iostream>
2
3  namespace hw1{
4
5
6      class cal{
7      private :
8          int num;
9          int try_;
10         int sum=0;
11         double avg=0;
12         int max_num=0;
13         int min_num=10000000;
14         void max(int num);
15
16         void min(int num1);
17
18     public:
19         void request() ;
20
21
22
23
24 };
25
26 }
```

먼저 과제 1번의 hpp파일의 코드에 대해 설명하겠다. namespace의 이름을 hw1로 설정하고 프로그램에서 사용할 cal 클래스를 생성하였다. private 란에 사용자에게 몇 번 수를 입력받을 것인지 입력받는 num, try, 총 합 계산할 때 사용할 변수인 sum, 평균을 계산할 avg, 최댓값을 저장할 때 사용할 max\_num 변수를 0으로 초기화 하여 선언하였다. 최솟값을 찾을 때 사용할 min\_num 변수는 이 변수의 초기값이 0이라면 0보다 큰 수와 비교했을 때 오류가 발생할 수 있으므로 매우 큰 숫자를 입력하여 초기화하였다. 이렇게 멤버 변수를 선언해주었고 최댓값, 최솟값을 판정하는 함수인 max와 min을 멤버 함수로 선언하였다. main에서 주로 작동할 함수 request를 public 란에 입력하였다,

```

1  #include <iostream> //
2  #include "head.hpp"
3
4  void hw1::cal::max(int num) {
5
6      if(num>max_num) max_num = num ;
7  }
8
9  void hw1::cal::min(int num1) {
10
11      if(num1<min_num) min_num = num1 ;
12  }
13
14
15  void hw1::cal::request() {
16
17      std::cout<<"몇 개의 원소를 할당하겠습니까? : " ;
18      std::cin >> try_;
19
20      int*num = new int[try_];
21      if(!num){
22          std::cout<< "메모리를 사용할 수 없습니다.";
23          return;
24      }
25
26      for(int i=0; i<try_; i++){
27
28          std::cout<<"정수형 데이터 입력:" ;
29          std::cin >> num[i];
30
31          if (std::cin.fail()) {
32              std::cout << "정수가 아닙니다." << std::endl;
33              return;}
34          sum += num[i];
35          min(num[i]);
36          max(num[i]);
37      }
38
39      avg = sum / try_;
40
41      std::cout<<"최댓값: "<<max_num << std::endl;
42      std::cout<<"최솟값: "<<min_num << std::endl;
43      std::cout<<"전체합: "<<sum << std::endl;
44      std::cout<<"평균: "<<avg << std::endl;
45
46      delete[] num;
47  }

```

```

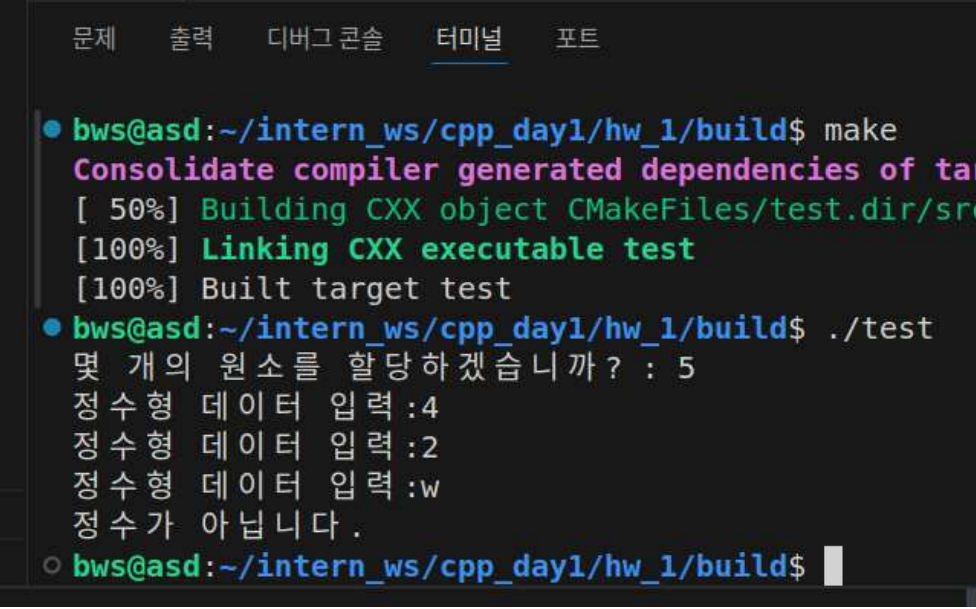
48
49  int main(int argc, char ** argv){
50
51      hw1::cal Cal;
52      Cal.request();
53
54
55  }

```

위 사진은 main cpp 파일의 코드를 캡처한 사진이다.

메인함수에서는 hpp에서 선언한 클래스 cal의 객체인 Cal을 선언하였다. request 함수를 살펴보면 설명을 진행하겠다. 먼저 사용자에게 몇 개의 원소를 할당할 것인지 묻는 문구가 출력되며 사용자가 입력한 수는

try\_에 저장된다. 이 저장된 값을 이용해 정수형 배열을 try\_ 값 만큼의 크기로 동적할당한다. 다만 이때 num이 NULL일 경우 예외처리를 해주어 오류문구를 출력하고 프로그램을 종료한다. 아까 저장한 try\_값 만큼 반복문을 작동시켜 동적할당한 배열에 계산할 값을 사용자로부터 입력받아 넣는다. 이 때 사용자가 정수형 데이터가 아닌 값을 입력할 경우 오류문구를 출력하고 프로그램을 종료시킨다.



```
문제   출력   디버그 콘솔   터미널   포트

● bws@asd:~/intern_ws/cpp_day1/hw_1/build$ make
Consolidate compiler generated dependencies of target test
[ 50%] Building CXX object CMakeFiles/test.dir/src/main.cpp.o
[100%] Linking CXX executable test
[100%] Built target test
● bws@asd:~/intern_ws/cpp_day1/hw_1/build$ ./test
몇 개의 원소를 할당하겠습니까? : 5
정수형 데이터 입력 :4
정수형 데이터 입력 :2
정수형 데이터 입력 :w
정수가 아닙니다.
○ bws@asd:~/intern_ws/cpp_day1/hw_1/build$
```

올바른 값이라면 그 값을 합계를 계산하기 위한 변수인 sum에 모두 합하여 저장한다. 그 후 최댓값, 최솟값을 저장하는 min과 max 함수를 이용한다. 이 두 함수를 살펴보면 현재 반복문 순서에서 사용자로부터 입력된 값을 매개변수로 받아 이전에 max\_num에 저장된 값보다 크면 max\_num을 해당 매개변수로 갱신하고 min\_num보다 작다면 min\_num을 해당 매개변수로 갱신한다. 이렇게 반복문이 모두 종료되면 숫자를 총합한 값이 저장된 sum을 try\_만큼 나누어 평균을 계산해 avg에 저장한다. 각 max\_num, min\_num, sum, avg에 계산에 맞는 값이 저장되어 있을 것이다. 이들을 cout을 이용해 모두 출력해준다. 마지막으로 동적할당한 num을 메모리해제 해주는 것도 잊지 않는다.

```
[100%] Built target test
bws@asd:~/intern_ws/cpp_day1/hw_1/build$ ./teset
bash: ./teset: No such file or directory
bws@asd:~/intern_ws/cpp_day1/hw_1/build$ ./test
몇 개의 원소를 할당하겠습니까? : 4
정수형 데이터 입력 :2
정수형 데이터 입력 :1
정수형 데이터 입력 :4
정수형 데이터 입력 :5
최댓값 : 5
최솟값 : 1
전체합 : 12
평균 : 3
bws@asd:~/intern_ws/cpp_day1/hw_1/build$
```

## 2. 과제 2번 코드 설명

```
pp_day1 > hw_2 > src > G-Head.hpp > {} hw2 > Cal
1 #include <iostream>
2
3 namespace hw2{
4
5
6 class Cal{
7
8
9 public:
10     void input();
11     struct map
12     {
13         int x;
14         int y;
15         int num;
16         int max;
17         int min;
18     };
19
20     void create_print();
21     void dist_cal();
22
23
24 private : // 메인에서 뒀음
25     int num;
26     int max;
27     int min;
28     double max_dist=0;
29     double min_dist=1000000;
30
31     int max_dist_x=0;
32     int min_dist_y=1000000;
33
34     int max_point1=0;
35     int max_point2=0 ;
36
37     int min_point1=0;
38     int min_point2=0 ;
39
40     map *Guzo;
41     double dist=0;
42
43     void max_dist(double num,int point3, int point4);
44     void min_dist(double num1,int point1, int point2);
45
46
47
48
49 };
50
51 }
```

해당 사진은 과제 2번의 hpp 파일을 캡처한 사진이다.

먼저 과제 2번의 hpp파일의 코드에 대해 설명하겠다. namespace의 이름을 hw2로 설정한 후 프로그램에서 사용할 Cal 클래스를 생성하였다.

public 란에 사용자에게 몇 번 좌표를 생성할 것인지 입력받는 멤버함수 input을 선언하였다. 생성한 2차원 좌표의 위치와 최솟값, 최댓값을 판정하기 위한 구조체 변수를 각각 선언해주었다.

생성한 좌표를 출력할 함수 create\_print와 각 좌표의 거리를 계산할 때 사용하는 dist\_cal 함수를 선언해주었다.

private 란에는 위에서 언급한 함수 내에서 사용할 멤버 변수를 선언하였다.(주로 최댓값, 최솟값 저장) 또한 프로그램에서 사용할 객체인 Guzo를 전역에서 사용하기 위해서 미리 선언하였다.

```
cpp_day1 > hw_2 > src > E- main.cpp > ...
1  #include <iostream>
2  #include "Head.hpp"
3  #include <cstdlib>
4  #include <ctime>
5  #include <cmath>
6
7
8  using namespace std;
9
10
11 void hw2::Cal::max_dist(double num,int point3, int point4) {
12
13     if(num>max_dist)
14     { max_dist = num ;
15       max_point1 = point3;
16       max_point2 = point4;
17     }
18 }
19
20 void hw2::Cal::min_dist(double num1,int point1, int point2) {
21
22     if(num1<min_dist) {
23         min_dist = num1 ;
24
25         min_point1 = point1;
26         min_point2 = point2;
27     }
28 }
29
30
31 void hw2::Cal::input() {
32
33
34     cout<< "좌표의 개수를 입력해주세요.: ";
35     cin>> num;
36     if(num<0) {cout<<"양수를 입력하세요"; exit(1); }
37     if (std::cin.fail()) {
38         std::cout << "양수를 입력하세요." << std::endl;
39         exit(1);
40     }
41     cout<< "좌표의 최소 범위를 입력해주세요.: ";
42     cin >> min;
43     cout<< "좌표의 최대 범위를 입력해주세요.: ";
44     cin >> max;
45     if(max<min) {cout<<"최대 범위가 최소 범위보다 작습니다."; exit(1); }
46     if (std::cin.fail()) {
47         std::cout << "정수가 아닙니다." << std::endl;
48         exit(1);
49     }
50     Guzo = new hw2::Cal::map[num];
51     for(int i=0; i<num; i++) {
```

```

31 void hw2::Cal::input() {
50
51     for(int i=0; i<num; i++) {
52
53         Guzo[i].x = rand() % (max - min + 1) + min;
54         Guzo[i].y = rand() % (max - min + 1) + min;
55
56
57
58     }
59
60 }
61
62
63 void hw2::Cal::create_print() {
64     cout<<"생성된 좌표 출력"<< endl;
65     cout<<"\n";
66
67     for(int i=0; i<num; i++){
68         cout << "point" << i+1<< ". X = " << Guzo[i].x << " , Y =" << Guzo[i].y << endl;
69     }
70 }
71
72 }
73
74 void hw2::Cal::dist_cal() {
75
76     for(int i=0; i<num; i++) {
77
78         for(int k=i+1; k<num; k++){
79
80             dist = std::sqrt(
81                 std::pow(Guzo[i].x - Guzo[k].x, 2) +
82                 std::pow(Guzo[i].y - Guzo[k].y, 2));
83
84             max_dist(dist,i,k);
85             min_dist(dist,i,k);
86         }
87     }
88
89     cout << "최소 거리: " << min_dist << endl;
90     cout << "해당 좌표: P1(" << Guzo[min_point1].x << ", "
91     << Guzo[min_point1].y<< ") & P2(" << Guzo[min_point2].x <<
92     ", "<< Guzo[min_point2].y << ")" << endl;
93
94
95     cout << "최대 거리: " << max_dist << endl;
96     cout << "해당 좌표: P1(" << Guzo[max_point1].x << ", "
97     << Guzo[max_point1].y<< ") & P2(" << Guzo[max_point2].x <<
98     ", "<< Guzo[max_point2].y << ")" << endl;
99

```

```

74 void hw2::Cal::dist_cal() {
76     for(int i=0; i<num; i++) {
78         for(int k=i+1; k<num; k++){
87
88     }
89
90     cout << "최소 거리: " << min_dist << endl;
91     cout << "해당 좌표: P1(" << Guzo[min_point1].x << ", "
92     << Guzo[min_point1].y<< ") & P2(" << Guzo[min_point2].x <<
93     ", "<< Guzo[min_point2].y << ")" << endl;
94
95
96     cout << "최대 거리: " << max_dist << endl;
97     cout << "해당 좌표: P1(" << Guzo[max_point1].x << ", "
98     << Guzo[max_point1].y<< ") & P2(" << Guzo[max_point2].x <<
99     ", "<< Guzo[max_point2].y << ")" << endl;
100
101     delete[] Guzo ;
102 }
103
104 int main() {
105
106     hw2::Cal Guzo;
107     Guzo.input();
108
109     cout<<"\n\n";
110
111     Guzo.create_print();
112     Guzo.dist_cal();
113
114
115     return 0;
116 }
117

```



과제 2번의 cpp 파일을 보며 설명을 계속하겠다. 랜덤으로 좌표를 생성하기 위해서 rand 함수를 불러오는데 사용하는 라이브러리들을 include 해주었다. 거리를 측정할 때 제곱 계산을 하기 위해서 cmath라이브러리를 추가하였다.

input 함수를 보며 설명을 이어가겠다. 먼저 생성할 좌표의 개수를 사용자로부터 입력받아 num 변수에 저장한다. 이때 사용자가 음수를 입력하여 오류가 발생할 수 있으므로 오류처리를 해주었다.

```
좌표의 개수를 입력해주세요.: -1
* 양수를 입력하세요 bws@asd:~/intern_ws/cpp_day1/hw_2/build$ ./test
좌표의 개수를 입력해주세요.: w
양수를 입력하세요.
o bws@asd:~/intern_ws/cpp_day1/hw_2/build$
```

올바르게 입력이 되었다면 최소 범위와 최대 범위를 사용자로부터 입력받는다. 이때도 정확하게 수를 입력하지 않는 상황에 대비하여 예외처리를 진행하였고 최대 범위를 입력할 때에 그 값이 최소 범위보다 작다면 오류가 발생하기에 이 또한 예외처리를 진행해 주었다.

```
o bws@asd:~/intern_ws/cpp_day1/hw_2/build$ ./test
좌표의 개수를 입력해주세요.: 20
좌표의 최소 범위를 입력해주세요.: 10
좌표의 최대 범위를 입력해주세요.: 4
o 최대 범위가 최소 범위보다 작습니다. bws@asd:~/intern_ws/cpp_day1/hw_2/build$
```

올바르게 입력 값을 모두 저장했다면 사용자가 생성하고자하는 좌표의 개수만큼 구조체 동적할당을 진행한다. 이렇게 만들어진 각 구조체의 x 및 y 좌표를 생성한 좌표의 개수만큼 반복문을 거치며 rand 함수 이용해 최소 범위와 최대 범위 사이의 값으로 입력한다.

올바르게 좌표가 생성되었는지 확인하기 위해 create\_print 멤버 함수를 통해 생성된 모든 좌표를 출력한다. 이제 각 좌표 간 거리를 계산해 최대 거리와 최소 거리를 가진 한 쌍의 좌표를 찾을 시간이다.

이 때 사용할 함수가 dist\_cal 함수이다. 이 함수를 살펴보면 이중반복문으로 두 구조체 배열의 x와 y값을 불러와서 pow 및 sqrt 함수로 거리 공식을 계산해 dist 변수에 저장한다. 이중반복문을 돌며 가장 dist가 큰 좌표와 가장 작은 좌표를 찾기 위해 계산한 dist, 조건에 부합하는 구조체의 정보를 저장하기 위한 인덱스 I,k값을 매개변수로 하는 max\_dist

와 min\_dist 멤버함수를 거친다. 이 두 함수를 살펴보겠다.  
max\_dist에 저장된 값보다 크면 max\_dist를 해당 매개변수로 갱신하고  
min\_dist보다 작다면 min\_dist를 해당 매개변수로 갱신한다.  
이 뿐만 아니라 최대, 최솟값을 가진 한 쌍의 좌표를 저장하기 위해  
max\_dist 또는 min\_dist를 저장할 때 l,k의 인덱스 값도 같이 저장한  
다. 최댓값 , 최솟값의 거리와 그 거리를 가지는 한 쌍의 좌표 또한 모  
두 저장하였다면 cout으로 모두 출력한다. 마지막으로 동적할당한 배열을  
메모리 해제하는 것 또한 잊지 않는다.

```
bws@asd:~/intern_ws/cpp_day1/hw_2/build$ ./test
좌표의 개수를 입력해주세요.: 10
좌표의 최소 범위를 입력해주세요.: 20
좌표의 최대 범위를 입력해주세요.: 40

생성된 좌표 출력

point1. X = 21 , Y =24
point2. X = 29 , Y =39
point3. X = 28 , Y =30
point4. X = 30 , Y =29
point5. X = 35 , Y =30
point6. X = 22 , Y =39
point7. X = 40 , Y =24
point8. X = 40 , Y =27
point9. X = 23 , Y =35
point10. X = 36 , Y =36
최소 거리: 2.23607
해당 좌표: P1(28,30) & P2(30,29)
최대 거리:23.4307
해당 좌표: P1(22,39) & P2(40,24)
bws@asd:~/intern_ws/cpp_day1/hw_2/build$
```

### 3. 과제 3번 코드 설명

```
cpp_day1 > hw_3 > src > HEAD.hpp > {} nws > Monster
1 namespace hw3
2 {class Monster;
3   char key;
4   class Player
5   {
6   public :
7     int HP, MP, x,y;
8
9     void Pplayer(int x,int y);
10    void Attack(Monster &m);
11    void Show_status();
12    void X_move(char key);
13    void Y_move(char key);
14
15  };
16
17
18  class Monster
19  {
20  public :
21    int HP, x,y;
22    Monster();
23    void Mmonster(int x, int y, int HP);
24    int Be_Attacked();
25
26  };
27 }
28
```

먼저 hpp 파일을 살펴보겠다. 우선 과제 명시되어 있는대로 코드를 작성하였고 main 코드에서 Player 멤버 변수가 계속 충돌을 일으켜 이름 Pplayer로 변경해주었다. Mmonster 멤버 함수 또한 마찬가지이다.

```

1 #include <iostream>
2 #include "HEAD.hpp"
3
4 using namespace hw3;
5 using namespace std;
6
7 Player p;
8 Monster m;
9
10 void Player::X_move(char key)
11 {
12     if(key=='L') {p.x -=1; cout << "X 축으로 -1 이동"<<endl;}
13     else if(key=='R') {p.x +=1; cout << "X 축으로 +1 이동"<<endl;}
14
15 }
16
17
18 void Player::Y_move(char key) {
19
20     if(key=='D') {p.y -=1; cout << "Y 축으로 -1 이동"<<endl;}
21     else if(key=='U') {p.y +=1; cout << "Y 축으로 +1 이동"<<endl;}
22 }
23
24
25 void Player::Show_status() {
26
27     cout << "HP: " << p.HP << endl;
28     cout << "MP: " << p.MP << endl;
29     cout << "Position: " << p.x << ", " << p.y << endl;
30 }
31
32 void Player::Attack(Monster &m) {
33
34
35     m.HP -=10;
36
37 }
38
39
40 void Monster::Mmonster(int x, int y, int hp) {
41
42     m.x = x;
43     m.y = y;
44     m.HP = 50;
45 }
46
47 void Player::Pplayer(int x, int y) {
48     p.x = x;
49     p.y = y;
50     p.HP = 50;
51     p.MP = 10;
52 }
53
54
55 int main () {
56
57     p.Pplayer(0,0);
58     m.Mmonster(5,4,50);
59     while (1)
60     {
61         cout << "Type Command (A/U/D/R/L/S)";
62         cin >> key;
63         if(key == 'M') exit(1);
64         if(key != 'A' && key != 'R' && key != 'D' && key != 'U' && key != 'S' && key != 'A') {cout << "올바른 키를 입력하세요."<<endl; continue;}
65         p.X_move(key);
66         p.Y_move(key);
67
68         if(key=='S') p.Show_status();
69         if(key=='A'){
70             if(p.MP<1) {cout << "MP 부족 ! "<<endl; exit(1);}
71             p.MP-=1;
72             if(p.x==5 && p.y==4) {
73
74                 cout << "공격 성공!"<<endl;
75                 p.Attack(m);
76                 cout << "남은 체력: " << m.HP << endl;
77             }
78             else {cout << "공격 실패!"<<endl;}
79         }
80     }
81
82     if(m.HP==0) {
83
84         cout << "Monster die!"<<endl;
85         exit(1);
86     }
87
88 }
89
90
91 return 0;
92 }
93

```

먼저 player와 Monster의 구조체 p,m을 전역으로 사용하기 위해서 각 선언해주었다. 메인함수를 살펴보면 설명을 이어가겠다. 프로그램이 시작되면 무한반복문에 들어가기 전 Pplayer, Mmonster 멤버 함수를 이용해 플레이어와 몬스터의 상태를 초기화해준다. 플레이어의 위치를

(0,0)으로 몬스터의 위치를 (5,4)로 설정하였고 몬스터의 Hp를 50으로 지정하였다. 플레이어의 mp와 hp를 각 10과 50으로 설정해주었다.

무한반복문을 돌며 cin을 이용해 명령키를 사용자로부터 입력받는다.

M키를 눌렀을 때 종료되는 것은 디버깅을 효율적으로 하기 위한 코드이다. 프로그램이 종료될 때까지 계속 진행해야하기 때문이다.

그 다음으로 넘어가서 사용자가 만약 지정된 명령키가 아닌 다른 키를 입력했다면 올바른 키를 입력해달라는 문구가 출력되고 무한반복문의 처음으로 돌아간다.

```
Type Command (A/U/D/R/L/S)Z
MP 부족 !
bws@asd:~/intern_ws/cpp_day1/hw_3/build$ ./test
Type Command (A/U/D/R/L/S)Z
올바른 키를 입력하세요 .
Type Command (A/U/D/R/L/S)
```

x\_move, y\_move 멤버 함수로 사용자가 이동 명령키를 눌렀을 때 각 키에 맞게 위치 값을 변경해주었고 S 명령키를 입력하면 현재 플레이어의 상태가 출력된다. show\_status 멤버 함수를 이용해 현재 플레이어의 위치 hp 및 mp상태가 출력된다. 사용자가 A명령키를 입력하면 mp가 남았는지 확인여부를 판단하는 조건문을 거친다. 만약 mp가 0 미만이라면 mp가 부족하다는 문구가 출력되고 프로그램은 종료된다. 0 미만이 아니라면 mp에서 -1을 빼주고 현재 플레이어의 위치가 몬스터의 위치와 동일하다면 조건문 아래가 실행된다. 공격을 성공했다는 문구가 출력되고 Attack 멤버 함수에서 몬스터의 체력을 10 감소시킨다. 플레이어의 위치가 몬스터의 위치와 동일하지 않다면 공격 실패 문구가 출력된다. 몬스터의 hp가 0이 되면 cout으로 몬스터가 물리쳤다는 문구가 출력되며 프로그램은 종료된다.

```
bws@asd:~/intern_ws/cpp_day1/hw_3/build$ ./test
```

```
Type Command (A/U/D/R/L/S)Z
```

```
올바른 키를 입력하세요 .
```

```
Type Command (A/U/D/R/L/S)U
```

```
Y 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)D
```

```
Y 축으로 -1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)L
```

```
X 축으로 -1 이동
```

```
Type Command (A/U/D/R/L/S)S
```

```
HP: 50
```

```
MP: 10
```

```
Position: 0,0
```

문제 출력 디버그 콘솔 터미널 포트

bash - build +

```
Y 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)U
```

```
Y 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)R
```

```
X 축으로 +1 이동
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 성공 !
```

```
남은 체력 : 40
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 성공 !
```

```
남은 체력 : 30
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 성공 !
```

```
남은 체력 : 20
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 성공 !
```

```
남은 체력 : 10
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 성공 !
```

```
남은 체력 : 0
```

```
Monster die!
```

```
Type Command (A/U/D/R/L/S)A 공격 실패 !
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 실패 !
```

```
Type Command (A/U/D/R/L/S)A
```

```
공격 실패 !
```

```
Type Command (A/U/D/R/L/S)S
```

```
HP: 50
```

```
MP: 0
```

```
Position: 0,5
```

```
Type Command (A/U/D/R/L/S)A
```

```
MP 부족 !
```

```
o bws@asd:~/intern_ws/cpp_day1/hw_3/build$
```