

Final project

Project description

In the beginning, the player needs to set the size of the map, then set village, monsters, player, gun and trap location. The player need to combat monster in this game. Each monster has a different value of strength and the player can only beat the monster which has less value of strength than the player. If the player defeats the monster, the player's strength can be incremented by 1. However, if the player can't beat the monster, the player's blood will be decremented by 1 to 4 according to the number of monsters. The player will have 10 blood and 2 strength at the beginning. The player can choose to go to the village before selecting which monster to combat. In the village, the player can do three actions. The first action is to buy a magical sword to increase the number of strengths by 1. The player can only buy one sword during the same visit to the village. If the player buys n swords after n iterations without fighting any monster, the player's strength will be increased by n. The second action is to go to the bar and have a drink, which will increase the value of the blood by 1. The player can go to the bar multiple times during the same visit to the village until the value of blood reaches 10. The third action is to exit the village. After exiting the village the player will be asked to choose which monster to fight. If the player move to the location of gun, player can increase player_attack. If the player move to trap, the player will minus his/her player_health.

Playing method

Using direction key to control the player's location. Follow the instruction on the screen. You can leave the game if you choose 5 to exit, or you will stop the game until your player_health less than 0.

Function description

- start_game() : Initial map, village, monster, player, gun, trap
- create_map() : Create a map for the game by using 2D array and asking user to input the size of row and column of the array

- `print_map()` : The program use **v** to label the location of the village, **m** to label the location of monsters, **p** to label the location of player, **g** to label the location of gun, **t** to label the location of trap
- `setup_village()` : set village location
- `setup_monster()` : set monster location
- `setup_player()` : set player location
- `setup_gun()` : set gun location
- `setup_trap()` : set trap location
- `encounter_village()` : player can do three things here, 1. buy a magical sword 2. go to the bar 3. leave the village
- `encounter_monster()` : player have to choose a monster(1~4)
- `battle_result()` : set random seed for checking if the monster dodge successfully and judge whether player or monster win
- `go_up()/go_down()/go_right()/go_left()` : check every moves legal, judge if the player encounter v, m, g, t
- `check_boundary()` : Show a warning message **“the location is outside the map”** when the location of the village or monsters is outside the range of the array.
- `check_availability()` : Check whether the location is already occupied by the village or other monsters. If the location is already taken by the village or other monsters, the program needs to show a warning message **“the location is occupied”**
- `travel_map()` : move the player up/down/right/left

Variable description

- `char** map` : declare a double pointer of a map
- `int map_row, map_col` : `map_row*map_col` size of map
- `int player_row, player_col /village_row, village_col/monster_row, monster_col/gun_row, gun_col/trap_row, trap_col` : the location of player/village/monster/gun/trap
- `int choose` : create a new map or start travel the map
- `int monster` : decide which monster to fight
- `int player_health` : initial as 10
- `int player_attack` : initial as 2
- `int seed` : dodge probability
- `int go` : decide whether go to the village or not
- `int sword` : judge whether the play already buy a sword

Version history

Global variables version

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4  #include<time.h>
5
6  int map_row, map_col, village_row, village_col, monster_row, monster_col, player_row, player_col, gun_row, gun_col, trap_row, trap_col;
7  int monster; // choose a monster to fight
8  int player_health =10;
9  int player_attack = 2;
10 char map[1000][1000];
11
12 int main(){
13     printf("Welcome, Adventurer!\n");
14     init_game();
15     return 0;
16 }
17 void create_map(){
18     printf("Input the number of row and column for the map: \n");
19     scanf("%d%d", &map_row, &map_col);
20     //set all the elements to be '.'
21     for(int i=0; i<map_row; i++){
22         for(int j=0; j<map_col; j++){
23             map[i][j] = '.';
24         }
25     }
26     print_map();
27 }
28 void print_map(){
29     printf("=== MAP ===\n");
30     for(int j=0; j<map_row; j++){
31         for(int k=0; k<map_col; k++){
32             printf("%c ", map[j][k]);
33         }
34         printf("\n");
35     }
36     printf("=== MAP ===\n");
37 }
```

```
37 void setup_village(){
38     while(1){
39         printf("Input the row and column for the village location: \n");
40         scanf("%d%d", &village_row, &village_col);
41         if(check_boundary(village_row, village_col) != 0 && check_availability(village_row, village_col) != 0){
42             map[village_row][village_col] = 'v';
43             break;
44         }
45     }
46     print_map(map_row, map_col);
47 }
48 void encounter_village(){
49     int go; // decide whether go to the village or not
50     int choose; // what to do in the village
51     int sword=1;
52     while(1){
53         printf("Do you want to go to our village first (1:Yes; 0: No)?\n"); //Ask if the player wants to go to the village.
54         scanf("%d", &go);
55         if(go == 1){
56             printf("What do you want to do in our village (1:buy a magical sword; 2: go to the bar; 3:leave the village)?\n");
57             scanf("%d", &choose);
58             if(choose == 1){
59                 if(sword == 1){
60                     player_attack++;
61                     printf("Nice sword! Now you have %d strength!\n", player_attack);
62                     sword = 0;
63                 }
64                 else if(sword == 0)
65                     printf("You already brought a sword.\n");
66             }
67             else if(choose == 2){
68                 if(player_health<10){
69                     player_health++;
70                     printf("Nice beer! Now you have %d player_health!\n", player_health);
71                 }
72                 else if(player_health>=10)
73                     printf("Your player_health is full. You don't need a beer.\n");
74             }
75         }
76     }
77 }
```

```

75         else if(choose == 3){
76             printf("You are welcome to come back anytime!\n");
77             break;
78         }
79     }
80     else
81         break;
82 }
83 }
84 void setup_monster(){
85     int monster = (map_row/10) + (map_row%10);
86     printf("Input the row and column for the monster location: \n");
87     for(int i=0; i<monster; i++){
88         printf("Input the row and column for monster %d: \n", i);
89         scanf("%d%d", &monster_row, &monster_col);
90         if(check_boundary(monster_row, monster_col) != 0 && check_availability(monster_row, monster_col) != 0)
91             map[monster_row][monster_col] = 'm';
92         else
93             i--;
94     }
95     print_map();
96 }
97 void encounter_monster(){
98     printf("Please choose which monster (1 to 4) you want to fight: \n");
99     scanf("%d", &monster);
100     if(monster<1 || monster>4)
101         printf("No such monster, please enter 1 to 4 to choose the monster.\n");
102     else{
103         printf("You are fighting monster %d!\n", monster);
104         battle_result();
105     }
106     if(player_health<=0){
107         printf("You Dead. Bye!\n");
108         exit(0);
109     }
110 }
111 void battle_result(){
112     srand(time(0));

```

```

113     int seed;
114     while(1){
115         seed = rand()%101;
116         if(seed>50){
117             printf("dodge probability is : %d\n", seed);
118             printf("monster dodge successfully\n");
119         }
120         else{
121             printf("dodge probability is : %d\n", seed);
122             if(player_attack>monster){
123                 player_attack++;
124                 printf("You Win! \n");
125                 break;
126             }
127             else{
128                 player_health -= monster;
129                 printf("You Lose!\n");
130                 break;
131             }
132         }
133         printf("Please choose which monster (1 to 4) you want to fight: \n");
134         scanf("%d", &monster);
135     }
136 }
137 void setup_player(){
138     while(1){
139         printf("Input the row and column for player: \n");
140         scanf("%d%d", &player_row, &player_col);
141         if(check_boundary(player_row, player_col) != 0 && check_availability(player_row, player_col) != 0){
142             map[player_row][player_col] = 'p';
143             break;
144         }
145     }
146     print_map();
147 }
148 void go_up(){
149     player_row--;
150     if(check_boundary(player_row, player_col) != 0){

```

```

151         if(map[player_row][player_col] == '.'){
152             map[player_row][player_col] = 'p';
153             map[player_row+1][player_col] = '.';
154         }
155         else if(map[player_row][player_col] == 'v'){
156             encounter_villiage();
157             player_row++;
158         }
159         else if(map[player_row][player_col] == 'm'){
160             encounter_monster();
161             player_row++;
162         }
163         else if(map[player_row][player_col] == 'g'){
164             player_attack++;
165             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
166             player_row++;
167         }
168         else if(map[player_row][player_col] == 't'){
169             player_health--;
170             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
171             player_row++;
172         }
173         print_map();
174     }
175     else{
176         printf("You can't go there!\n");
177         player_row++;
178     }
179 }
180 void go_down(){
181     player_row++;
182     if(check_boundary(player_row, player_col) != 0){
183         if(map[player_row][player_col] == '.'){
184             map[player_row][player_col] = 'p';
185             map[player_row-1][player_col] = '.';
186         }
187         else if(map[player_row][player_col] == 'v'){
188             encounter_villiage();

```

```

189         player_row--;
190     }
191     else if(map[player_row][player_col] == 'm'){
192         encounter_monster();
193         player_row--;
194     }
195     else if(map[player_row][player_col] == 'g'){
196         player_attack++;
197         printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
198         player_row--;
199     }
200     else if(map[player_row][player_col] == 't'){
201         player_health--;
202         printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
203         player_row--;
204     }
205     print_map();
206 }
207 else{
208     printf("You can't go there!\n");
209     player_row--;
210 }
211 }
212 void go_right(){
213     player_col++;
214     if(check_boundary(player_row, player_col) != 0){
215         if(map[player_row][player_col] == '.'){
216             map[player_row][player_col] = 'p';
217             map[player_row][player_col-1] = '.';
218         }
219         else if(map[player_row][player_col] == 'v'){
220             encounter_villiage();
221             player_col--;
222         }
223         else if(map[player_row][player_col] == 'm'){
224             encounter_monster();
225             player_col--;
226     }

```

```

227         else if(map[player_row][player_col] == 'g'){
228             player_attack++;
229             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
230             player_col--;
231         }
232         else if(map[player_row][player_col] == 't'){
233             player_health--;
234             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
235             player_col--;
236         }
237         print_map();
238     }
239     else{
240         printf("You can't go there!\n");
241         player_col--;
242     }
243 }
244 void go_left(){
245     player_col--;
246     if(check_boundary(player_row, player_col) != 0){
247         if(map[player_row][player_col] == '.'){
248             map[player_row][player_col] = 'p';
249             map[player_row][player_col+1] = '.';
250         }
251         else if(map[player_row][player_col] == 'v'){
252             encounter_villiage();
253             player_col++;
254         }
255         else if(map[player_row][player_col] == 'm'){
256             encounter_monster();
257             player_col++;
258         }
259         else if(map[player_row][player_col] == 'g'){
260             player_attack++;
261             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
262             player_col++;
263         }

```

```

264         else if(map[player_row][player_col] == 't'){
265             player_health--;
266             printf("You have %d player_health and %d player_attack.\n", player_health, player_attack);
267             player_col++;
268         }
269         print_map();
270     }
271     else{
272         printf("You can't go there!\n");
273         player_col++;
274     }
275 }
276 void setup_gun(){
277     while(1){
278         printf("Input the row and column for gun: \n");
279         scanf("%d%d", &gun_row, &gun_col);
280         if(check_boundary(gun_row, gun_col) != 0 && check_availability(gun_row, gun_col) != 0){
281             map[gun_row][gun_col] = 'g';
282             break;
283         }
284     }
285     print_map();
286 }
287 void setup_trap(){
288     while(1){
289         printf("Input the row and column for trap: \n");
290         scanf("%d%d", &trap_row, &trap_col);
291         if(check_boundary(trap_row, trap_col) != 0 && check_availability(trap_row, trap_col) != 0){
292             map[trap_row][trap_col] = 't';
293             break;
294         }
295     }
296 }
297 int check_boundary(int row, int col){
298     if(row<0 || row>map_row-1 || col<0 || col>map_col-1){
299         printf("the location is outside the map\n");
300         return 0;
301     }

```

```

302     else if(row<map_row && col<map_col)
303         return 1;
304 }
305 int check_availability(int row, int col){
306     if(map[row][col] == '.')
307         return 1;
308     else{
309         printf("the location is occupied\n");
310         return 0;
311     }
312 }
313 void travel_map(){
314     print_map();
315     while(1){
316         int choose;
317         printf("[1] go up [2] go down [3] go right [4] go left [5] exit: ");
318         scanf("%d", &choose);
319         if(choose == 1)
320             go_up();
321         else if(choose == 2)
322             go_down();
323         else if(choose == 3)
324             go_right();
325         else if(choose == 4)
326             go_left();
327         else if(choose == 5)
328             exit(0);
329     }
330 }
331 void init_game(){
332     while(1){
333         int choose;
334         printf("[1] Create a new map [2] Start travel the map: ");
335         scanf("%d", &choose);
336         if(choose == 1){
337             create_map();
338             setup_village();
339             setup_monster();
340         }
341         else if(choose == 2){
342             setup_player();
343             setup_gun();
344             setup_trap();
345             travel_map();
346         }
347     }
348 }

```