어썸-미로 (전공 보고서)

어썸-미로 : 2019.05.20

개발 환경 : Mac (GCC) / Windows (Visual Studio 2019)

전공 보고서

사용된 알고리즘

전역변수를 활용하여 main외에 기타 함수에서도 동일하게 변수를 사용을 하여 점수 추가와 캐릭터의 움직임에 대한 좌표를 설정 할수 있었습니다.

```
//Easy Play Map
int easyMap[10][10] = {
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 1, 1, 1, 0, 1, 1, 1, 0, 0},
    {0, 0, 0, 1, 0, 1, 0, 1, 0, 0},
    {0, 0, 0, 1, 0, 1, 0, 1, 0, 0},
    {0, 0, 1, 1, 0, 1, 0, 1, 0, 0},
    {0, 0, 1, 0, 0, 1, 0, 1, 0, 0},
    {0, 0, 1, 0, 0, 1, 0, 1, 1, 0},
    {0, 0, 1, 0, 0, 1, 0, 0, 1, 0},
    {0, 0, 1, 1, 1, 1, 0, 0, 2, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
};
```

맵들은 10x10, 20x20, 30x30 배열로 총 3개의 맵을 만들었습니다.

```
//Normal Play Map
int normalMap[20][20] = {
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 3, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0},
    {0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0},
    {0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0},
    {0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0},
    {0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0},
    {0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0},
    {0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0},
    {0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0},
    {0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0},
    {0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0},
    {0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0},
    {0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 2, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
};
```

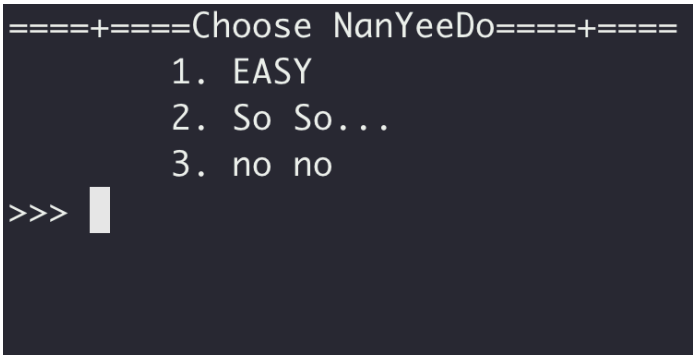
```
//Hard Play Map
int hardMap[30][30] = {
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 3, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0},
    {0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0},
    {0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0},
    {0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0},
    {0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0},
    {0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0},
    {0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0},
    {0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0},
    {0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0},
    {0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0},
    {0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0},
    {0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0},
    {0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0},
    {0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0},
    {0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0},
    {0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0},
    {0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 2, 0, 1, 0, 0, 0, 0},
    {0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0},
    {0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0},
    {0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0},
    {0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
};
```

0 일 경우에는 벽 * 을 출력하고 1일 경우에는 캐릭터가 지나갈 공백을 출력을 합니다. 2인 경우에는 도착 지점 $을 출력하고 3인 경우에는 캐릭터를 출력을 합니다. 맵은 정해져 있고 맵에서 나가기 위해서는 WASD 키 버튼을 눌러가면서 움직이여야 합니다.

```
109  //Intro Select Options
110  void intro() {
111      while (1) {
112          printf("====+====AWESOME_MIRO====+====\n");
113
114          printf("\t1.Play Game.\n");
115          printf("\t2.How to Play.\n");
116          printf("\t3. Exit :( \n");
117
118          printf(">>> ");
119          scanf("%d", &choose);
120
121          switch (choose) {
122              case 1:
123                  selectGrade();
124                  break;
125              case 2:
126                  printHowto();
127                  break;
128              case 3:
129                  exit(1);
130                  break;
131          }
132      }
133  }
```

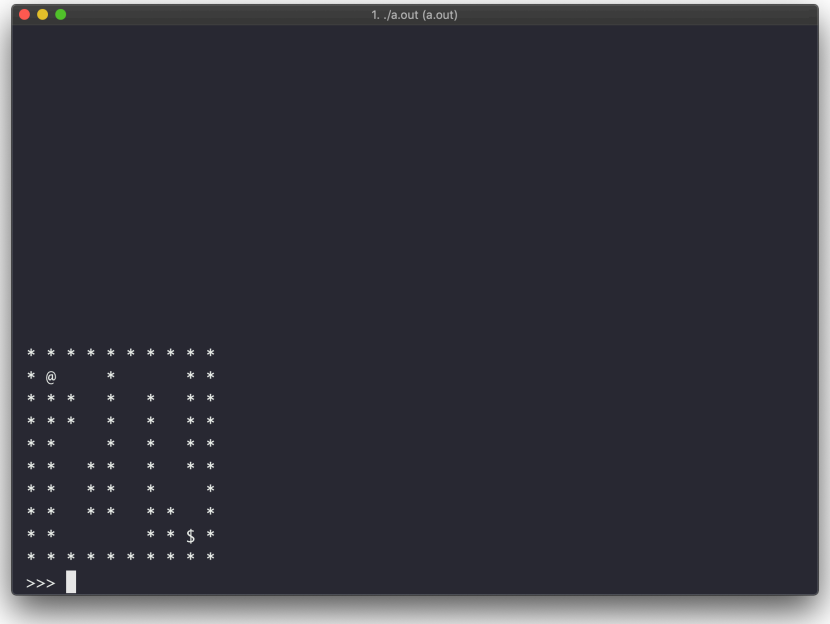
첫번째로 실행이 되는 인트로는 게임 플레이, 게임 플레이 방법, 종료를 선택 할수 있습니다. 1을 누르면 게임을 선택을 하고 게임 난이도 선택으로 넘어갑니다. 2을 누르면 게임을 플레이 하는 방법으로 넘어갑니다. 3을 누르면 exit(1); 함수를 이용하여 현재 프로세스를 종료합니다.

1를 누르면 아래와 같이 나옵니다.



여기서 게임의 난이도를 선택하게 됩니다. 1을 선택할 경우에는 가장 쉬운 Easy를 선택하게 됩니다. 2을 선택하게 될 경우에는 Normal을 선택하게 됩니다. 3을 누를 경우에는 가장 어려운 Hard을 선택하게 됩니다.

이후에 1, Easy를 누르면 mainEasyplay함수로 넘어가면서 게임의 주 플레이가 시작이 됩니다.



```
135  //Choose Easy, normal, hard...
136  //When it thourgh this code, then player have to go main play game Screen
137  void selectGrade() {
138      printf("====+====Choose NanYeeDo====+====\n");
139
140      printf("\t1. EASY\n");
141      printf("\t2. So So...\n");
142      printf("\t3. no no\n");
143
144      printf(">>> ");
145      scanf("%d", &choose);
146
147      switch(choose) {
148          case 1:
149              mainEasyplay();
150              break;
151          case 2:
152              mainNormalplay();
153              break;
154          case 3:
155              mainHardplay();
156              break;
157      }
158  }
```

가장 먼저 원활한 플레이를 위해서 printCleaning(); 함수를 활용하여 깔-끔하게 화면을 정리 해줍니다. RemoveCursor는 system("cls"); 을 활용하기 때문에 커서가 반짝 거립니다. 이를 해결 하기 위해서 RemoveCursor 함수를 만들어서 반짝 반짝을 해결 합니다. 그리고 가장 중요한 printEasyMap(); 함수는 Easy 맵을 출력하고 캐릭터의 위치를 받고 출력 하는 함수 입니다.

```
336  void printMove() {
337          int looping = 0;
338          char keyValue = _getch();
339
340          switch (keyValue) {
341          case 'w':
342                  --move_y;
343                  break;
344
345          case 'a':
346                  --move_x;
347                  break;
348
349          case 'd':
350                  ++move_x;
351                  break;
352
353          case 's':
354                  ++move_y;
355                  break;
356
357          case 'q':
358                  exit(1);
359                  break;
360
361          default:
362                  break;
363          }
364  }
```

EasyMap은 10 곱하기 10 배열 이기 때문에 루프를 10번, 10번 돌면서 출력을 해야합니다. 돌면서 배열에 숫자를 체킹을 하면서 출력을 합니다. 가장 첫번째로 캐릭터가 벽에 쿵 했는지 안했는지를 확인합니다. move_x, move_y는 캐릭터가 움직인 x와 y값을 담는 변수 입니다. 만일 캐릭터가 움직인 좌표의 값이 0 이라면 벽에 충돌한거 이기 때문에 crash(); 함수로 충돌로 끝을 냅니다. 그리고 캐릭터가 움직인 값이 2 일때 도착 지점에 도달하였기 때문에 ending(); 함수로 끝을 냅니다. 그리고 움직인 후에 값을 변경 해서 출력을 해야하기 때문에 움직인값과 루프의 값이 동일 할때 3으로 바꾸어 @ 캐릭터의 자리를 만들어줍니다. 그리고 최종적으로 Switch Case 문으로 벽, 공백, 도착지점, 캐릭터를 출력 해줍니다. 다만 캐릭터를 출력한 후에는 캐릭터가 지나간 자리에 공백으로 바꾸어 줍니다.
이 알고리즘은 Easy 말고도 Normal, Hard 모두 같은 알고리즘으로 출력 및 충돌 체크 등을 진행 합니다. 그러므로 printNormalMap(); 와 printHardMap(); 에 대한 추가 설명은 생략하겠습니다.

```
161         int random = 0;
162         int previousValue = 0;
163 
164         for (int x = 0; x < 10; x++) {
165                 for (int y = 0; y < 10; y++) {
166                         random = easyMap[x][y];
167                         previousValue = easyMap[x][y];
168 
169                         if (easyMap[move_y][move_x] == 0) {
170                                 crash();
171                                 exit(1);
172                         }
173 
174                         if (easyMap[move_x][move_y] == 2) {
175                                 ending();
176                         }
177 
178                         if (x == move_y && y == move_x) {
179                                 easyMap[x][y] = 3;
180                         }
181 
182                         switch (random) {
183                         case 0:
184                                 printf(" *");
185                                 break;
186                         case 1:
187                                 printf("  ");
188                                 break;
189                         case 2:
190                                 printf(" $");
191                                 break;
192                         case 3:
193                                 printf(" @");
194                                 easyMap[move_y][move_x] = 1;
195                                 break;
196                         }
197                 }   printf("\n");
```

캐릭터가 움직이기 위해서는 이동에 대한 값을 증가, 감소를 시키면서 배열에서 움직여야 합니다. 만일 캐릭터가 왼쪽으로 갔다면 x 값을 증가 하고 반대로 오른쪽으로 갔다고 하면 x 값을 감소 시켜야 합니다. y도 똑같습니다. 만일 캐릭터가 위로 갔다면 y 값은 감소, 아래로 갔다면 y 값은 증가 입니다. 이를 활용하여 wasd로 움직인 값을 받아서 move_x, move_y 값을 증가와 감소로 좌표를 만들어 줍니다.

```
291 void mainEasyplay() {
292         //ready('\n');
293         for (int i = 0; i >= score; i++) {
294                 printCleaning();
295                 RemoveCursor();
296                 printEasyMap();
297 
298                 printCleaning();
299                 RemoveCursor();
300                 printEasyMap();
301                 printMove();
302         }
303 }
```

그리고 위에 함수들을 한번에 모아서 돌려주는 함수 입니다.

.