

프로젝트 매뉴얼

<준식쓰와 형님들>

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

- 변 경 이 력 -

| 일자 | 버전 | 변경 내역 | 작 성 자 |
|---------------|----------|-------|-------|
| 2019. 12. 08. | Ver1.0.0 | 초안 작성 | 모두 |
| | | | |
| | | | |
| | | | |
| | | | |

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

- 목 차 -

| | | |
|-----|----------------------|--------|
| 1. | 소스코드 리뷰 | - 3 - |
| 1.1 | 클라이언트 | - 3 - |
| 1.2 | 서버 | - 11 - |
| 1.3 | 로그 뷰어 | - 27 - |
| 2. | PENTA-GO 사용지침서 | - 28 - |

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

1. 소스코드 리뷰

1.1 클라이언트

서버와 게임을 플레이하기 위한 클라이언트를 생성한다.

헤더 파일 추가 및 포트 번호 지정

```

1 #include <arpa/inet.h>
2 #include <sys/un.h>
3 #include <sys/socket.h>
4 #include <sys/types.h>
5 #include <netdb.h>
6 #include <stdio.h>
7 #include <unistd.h>
8 #include <stdlib.h>
9 #include <string.h>
10 #include <signal.h>
11 #include <time.h>
12 #include <stdio_ext.h>
13 #define PORTNUM 9000

```

1 ~ 12: 해당 파일에서 필요한 헤더 파일을 추가

13: 소켓 통신을 위한 포트 번호를 지정

함수 원형 선언

```

15 void get_board(int sd); // 현재 보드의 상태를 출력해주는 함수
16 int send_fix_board(int sd, char dol); // 현재 보드의 원하는 위치에 돌을 놓는 함수
17 void rotate_board(int sd); // 현재 보드에 원하는 사분면에 원하는 방향으로 회전시키는 함수
18 int check_pentago(int sd); // 게임이 끝났는지 확인하는 함수
19 int end_turn(int sd);

```

15 ~ 19: 해당 파일에서 구현할 함수의 원형을 선언

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

Main 함수

```

21 int main(void) {
22     signal(SIGINT, SIG_IGN); // 불계승/패 막기위한 시그널 이그노어
23     int is_end = 0;          // 게임이 끝난것을 확인하는 변수,
24     int sd; // 소켓파일기술자 위한 변수
25     struct sockaddr_in sin, cli; // 소켓통신 위한 변수
26     time_t start_time, end_time;
27     int play_time;
28
29     if((sd = socket(AF_INET, SOCK_STREAM, 0)) == -1) { // 소켓 생성하기
30         perror("socket");
31         exit(1);
32     }
33
34     memset((char*)&sin, '\0', sizeof(int));
35     sin.sin_family = AF_INET;
36     sin.sin_port = htons(PORTNUM);
37     sin.sin_addr.s_addr = inet_addr("222.236.11.238"); // 채선이 서버 연결시
38     //sin.sin_addr.s_addr = inet_addr("127.0.0.1"); // 로컬 서버 연결시
39
40     if(connect(sd, (struct sockaddr *)&sin, sizeof(sin))) { // 서버에 접속 요청
41         perror("connect");
42         exit(1);
43     }
44
45     start_time = time(NULL); // 게임 시작 시각
46
47     while (is_end == 0) { // 플레이 하는 일련의 과정
48         get_board(sd); // 보드를 받아온다
49
50         while (send_fix_board(sd, '0') != 0); // 돌 없는곳에 돌 두기
51         get_board(sd); // 보드를 받아온다
52         if(is_end = check_pentago(sd)) break; // 게임이 끝났다면, 게임을 끝낸다.
53         rotate_board(sd); // 보드를 돌린다
54         get_board(sd); // 보드를 받아온다
55         is_end = check_pentago(sd); // 게임 종료 확인한다.
56         if(is_end != 0) break; // 게임이 끝났다면, 게임을 끝낸다.
57         is_end = end_turn(sd); // 턴을 넘긴다.
58     }
59     end_time = time(NULL); // 게임 끝나는 시각
60     play_time = end_time - start_time; // 총 게임 시간
61     printf("플레이 시간 : %02d:%02d:%02d\n", (play_time) / 3600, (play_time / 60) % 60,
62     play_time % 60);
63 }

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

63  if (is_end == 2) printf("패배\n"); // 승리, 패배를 보여준다.
64  else printf("승리\n");
65  close(sd); // 소켓 닫기
66  return 0;
67 }

```

22: 불계승 / 패를 막기 위한 시그널 생성
23 ~ 27: 지역변수 선언
29 ~ 32: 소켓 생성
34 ~ 38: 소켓에 포트 지정 및 IP 지정
40 ~ 43: 서버와 연결
45: 게임 시작 시간을 start_time 변수에 저장
47 ~ 58: 플레이 과정
48: 보드 받아오기
50: 돌 없는 곳에 돌 두기
51: 보드 받아오기
52: 5개의 돌이 이어졌는지 확인 후 게임 진행 여부 결정
53: 보드 돌리기
54: 보드 받아오기
55: 게임 진행 여부 확인
56: is_end가 0이 아니며 게임 종료
57: 서버에게 턴 넘기기
59: end_time에 현재 시각 저장
60 ~ 61: play_time 계산 후 출력
63 ~ 64: 클라이언트의 승리 또는 패배 결정
65 ~ 66: 소켓 닫은 후 종료

보드의 상태를 받아오는 함수

```

69 // 보드 상태를 받아오는 함수, buf에 문자열 형태로 받아오고,
70 // 적절히 잘라서 출력해준다.
71
72 void get_board(int sd) {
73     char buf[365];
74     if(send(sd, "1", strlen("1")+1, 0) == -1) {
75         perror("send");
76         exit(1);
77     }
78
79     system("clear");
80

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문서명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

81  if(recv(sd, buf, sizeof(buf), 0) == -1) {
82      perror("recv");
83      exit(1);
84  }
85
86  for(int i = 0; i < 14; i++) {
87      for(int j = 0; j < 26; j++) {
88          fflush(stdout);
89          printf("%c", buf[26*i + j]);
90      }
91      printf("\n");
92  }
93 }

```

73: 판의 정보를 저장할 buf 배열 선언

74 ~ 77: 서버 측에 문자 "1" 전송

79: clear 시스템 명령 실행

81 ~ 84: 서버로부터 게임 판의 정보 받아와 buf 배열에 저장

86 ~ 92: 화면에 게임 판 출력

보드에 돌을 놓는 함수

```

96 // 보드에 돌을 놓는 함수
97 // row, col 에다가 dol을 놓는다.
98 int send_fix_board(int sd, char dol) {
99     char x, y;
100    char buf[128];
101    char str[4];
102    char rcv[4];
103    if(send(sd, "2", strlen("2")+1, 0) == -1) {
104        perror("send");
105        exit(1);
106    }
107
108    if(recv(sd, buf, sizeof(buf), 0) == -1) {
109        perror("recv");
110        exit(1);
111    }
112    fflush(stdout);
113    printf("좌표 (ex, A1) :\n");
114

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

115 while(1) {
116     // 플레이어가 원하는 좌표를 입력받는다.
117     __fpurge(stdin);
118     x = getc(stdin);
119     y = getc(stdin);
120     __fpurge(stdin);
121     if (( (x >= 'A' && x <= 'F') || (x >= 'a' && x <= 'f')) && y >= '1' && y <= '6')
122     {
123         if (x >= 'a' && x <= 'f') x -= 32; // 'a' - 'A' = 32 소문자를 대문자로
124         break;
125     } else {
126         printf("잘못 입력하셨습니다. 다시 입력하세요 :");
127     }
128 }
129 str[0] = x;
130 str[1] = y;
131 str[2] = dol;
132 str[3] = '\0';
133
134 if(send(sd, str, strlen(str)+1, 0) == -1) {
135     perror("send");
136     exit(1);
137 }
138 if(recv(sd, rcv, sizeof(rcv), 0) == -1) {
139     perror("recv");
140     exit(1);
141 }
142 if (strcmp(rcv, "0") == 0) return 0;
143 else return -1;
144 }

```

99 ~ 102: 해당 함수에서 필요한 지역 변수 선언

103 ~ 106: 서버 측에 "2" 보내기

108 ~ 111: 서버로부터 받은 정보를 buf 배열에 저장

112 ~ 128: 돌 놓을 좌표 입력 받기

118 ~ 119: 가로, 세로 위치 저장

121 ~ 128: 가로는 A ~ F(a ~ f), 세로는 1 ~ 6 정보만을 입력받을 수 있다.

만약 이외의 수가 들어오면 다시 입력받는다.

129 ~ 132: 서버측으로 위치 정보를 전달하기 위해 변수에 정보 저장

134 ~ 137: 서버측에 위치 정보 전달

138 ~ 141: 서버로부터 돌을 놓을 수 있는지 없는지에 대한 정보 받기

142 ~ 143: 돌을 놓을 수 있으면 0, 아니면 -1의 값에 따라 return 값 결정

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문서명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

게임 판 돌리기 함수

```

146 void rotate_board(int sd) {
147     char quadrant;    // 회전시킬 사분면을 저장하는 변수
148     char buf[16];
149     char c;           // 시계방향?
150     char str[3];
151
152     if(send(sd, "3", strlen("3")+1, 0) == -1) {
153         perror("send");
154         exit(1);
155     }
156     if(recv(sd, buf, sizeof(buf), 0) == -1) {
157         perror("recv");
158         exit(1);
159     }
160     fflush(stdout);
161     printf("┌───┐\n");
162     printf("| 1 | 2 |\n");
163     printf("├───┤\n");
164     printf("| 3 | 4 |\n");
165     printf("└───┘\n");
166
167     printf("회전할 사분면\n");
168     while (1) {
169         __fpurge(stdin);
170         quadrant = getc(stdin);
171         __fpurge(stdin);
172         if (quadrant - '0' >= 1 && quadrant - '0' <= 4) break;
173         printf("잘못 입력하셨습니다. 다시 입력하세요 :");
174     }
175     printf("시계방향?(y/n)\n");
176     while (1) {
177         __fpurge(stdin);
178         c = getc(stdin);
179         __fpurge(stdin);
180         if (c == 'y' || c == 'Y' || c == 'n' || c == 'N') {
181             if (c == 'y' || c == 'Y') c = '1';
182             else c = '3';
183             break;
184         } else printf("잘못 입력하셨습니다. 다시 입력하세요 :");
185     }
186     str[0] = quadrant;
187     str[1] = c;
188     str[2] = '\0';

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

189 if(send(sd, str, strlen(str)+1, 0) == -1) {
190     perror("send");
191     exit(1);
192 }
193
194
195
196 if(recv(sd, buf, sizeof(buf), 0) == -1) {
197     perror("recv");
198     exit(1);
199 }
200 fflush(stdout);
201
202
203 }

```

147 ~ 150: 해당 함수에서 사용할 지역 변수 선언
152 ~ 155: 서버에게 "3" 보내기
156 ~ 159: 서버에서 보낸 정보 저장(의미 없음)
161 ~ 167: 사분면 정보 화면에 출력
168 ~ 174: 회전할 사분면 정보 입력받기
175 ~ 185: 90도 회전을 시계방향 또는 반시계방향으로 할 것인지 입력
186 ~ 188: 사분면 및 회전 방향 정보 str 배열에 저장
189 ~ 192: str배열 서버에게 보내기
196 ~ 199: 서버에서 보낸 정보 저장(의미 없음)

게임이 끝났는지 확인하는 함수

```

205 // 게임이 끝났는지 확인하는 함수
206 int check_pentago(int sd) {
207     char buf[2];
208     if(send(sd, "4", strlen("4")+1, 0) == -1) {
209         perror("send");
210         exit(1);
211     }
212
213     if(recv(sd, buf, sizeof(buf), 0) == -1) {
214         perror("recv");
215         exit(1);
216     }
217
218     if(strcmp(buf, "0") == 0) return 0;
219     else return 1;
220 }

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

207: 게임 진행 여부 정보 저장할 변수 선언
 208 ~ 211: 서버에게 "4" 전달
 213 ~ 216: 서버로부터 게임 진행 여부 정보 받아 buf 배열에 저장
 218 ~ 220: 0이면 게임 진행, 1이면 게임 종료

서버에게 턴 넘기는 함수

```

221 int end_turn(int sd) {
222     char buf[2];
223     if(send(sd, "5", strlen("5")+1, 0) == -1) {
224         perror("send");
225         exit(1);
226     }
227
228     if(recv(sd, buf, sizeof(buf), 0) == -1) {
229         perror("recv");
230         exit(1);
231     }
232     // 0이 반환되면 게임이 안끝남
233     if(strcmp(buf, "0") == 0) return 0;
234     else return 2;
235 }
```

222: 변수 선언
 223 ~ 226: 서버에게 "5" 전달
 228 ~ 231: 서버로부터 정보 전달 받아 buf 배열에 저장
 233 ~ 234: 서버로부터 받은 정보가 0이면 게임 종료, 2면 게임 진행

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문서명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

1.2 서버

헤더 파일 및 포트 번호 지정

```

1 #include <arpa/inet.h>
2 #include <sys/un.h>
3 #include <sys/socket.h>
4 #include <sys/types.h>
5 #include <sys/stat.h>
6 #include <signal.h>
7 #include <netdb.h>
8 #include <stdio.h>
9 #include <unistd.h>
10 #include <stdlib.h>
11 #include <string.h>
12 #include <signal.h>
13 #include <fcntl.h>
14 #include <time.h>
15 #include <stdio_ext.h>
16 #define PORTNUM 9000

```

1 ~ 15: 해당 파일에서 필요한 헤더 파일을 추가

16: 소켓 통신을 위한 포트 번호를 지정

전역 변수 및 함수 원형 선언

```

18 char arr[6][6]; // 펜타고 보드 배열
19 int fd; // 파일 디스크립터 , 기보 저장을 위함
20
21 /* 클라이언트 플레이 함수 */
22 void send_board(int ns); // 현재 보드를 문자열로 보내는 함수
23 void fix_board(int ns); // 현재 보드의 원하는 위치에 돌을 놓는 함수
24 void rotate_board(int ns); // 현재 보드에 원하는 사분면에 원하는 방향으로 회전시키는
함수
25 int is_finish(int ns);
26
27 /* 서버 플레이 함수 */
28 void print_board(); // 현재 보드의 상태를 출력해주는 함수
29 int my_turn(int ns, char dol);
30 void init_board(); // 보드를 깨끗한 상태로 초기화 하는 함수
31 int my_fix_board(int col, int row, char dol); // 현재 보드의 원하는 위치에 돌을 놓는
함수
32 void my_rotate_board(int quad, int c); // 현재 보드에 원하는 사분면에 원하는 방향으로
회전시키는 함수
33 int check_pentago(); // 게임이 끝났는지 확인하는 함수

```

18 ~ 19: 전역 변수 지정

22 ~ 33: 해당 파일에서 구현할 함수의 원형을 선언

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

Main 함수

```

35 int main(void) {
36     signal(SIGINT, SIG_IGN);
37     char type[2]; // 수행할 서비스의 종류를 저장하기 위한 변수
38     char file_name[128];
39     struct sockaddr_in sin, cli;
40     int sd, ns, clientlen = sizeof(cli);
41     struct tm *tm;
42     time_t start_time = time(NULL), end_time;
43     int play_time;
44     int is_end = 0; // 게임이 끝난것을 확인하는 변수,
45                     // 0이면 게임이 끝나지 않은 상태,
46                     // 1이면 흑돌 win, 2이면, 백돌 win
47
48     // 기보를 저장하기 위해 파일 디스크립터 지정
49     tm = localtime(&start_time);
50     sprintf(file_name, "./Pentagologs/%d%02d%02d_%02d_%02d_%02d.txt", (int)tm->tm_year
+ 1900,
51             (int)tm->tm_mon+1, (int)tm->tm_mday, (int)tm->tm_hour, (int)tm->tm_min, (int)tm-
>tm_sec);
52     mkdir("./Pentagologs", 0777);
53     fd = open(file_name, O_CREAT | O_WRONLY | O_APPEND ,0664);
54     if (fd == -1) {
55         perror("Creat");
56         exit(1);
57     }
58
59     if((sd = socket(AF_INET, SOCK_STREAM, 0)) == -1) { // 소켓 생성하기
60         perror("socket");
61         exit(1);
62     }
63
64     memset((char*)&sin, '\0', sizeof(int));
65     sin.sin_family = AF_INET;
66     sin.sin_port = htons(PORTNUM);
67     //sin.sin_addr.s_addr = inet_addr("127.0.0.1");
68     sin.sin_addr.s_addr = INADDR_ANY;
69
70     if(bind(sd, (struct sockaddr *)&sin, sizeof(sin))) {
71         perror("bind");
72         exit(1);
73     }
74
75     system("clear");
76     printf("도전자를 기다리는중...\n"); // 소켓 바인드 되기 전까지 대기
77
78     if(listen(sd, 1)) {
79         perror("listen");
80         exit(1);
81     }

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

82
83 if ((ns = accept(sd, (struct sockaddr *)&cli, &clientlen)) == -1) {
84     perror("accept");
85     exit(1);
86 }
87
88 system("clear");
89 printf("게임이 시작됩니다.\n");
90 sleep(1);
91
92 init_board(); // 게임을 시작하기 전 보드의 상태를 초기화 한다.
93
94 start_time = time(NULL); // 게임 시작 시각
95 while(is_end == 0) { // 플레이 하는 일련의 과정
96     system("clear");
97     print_board(); // 보드를 출력함
98
99     if (recv(ns, type, sizeof(type), 0) == -1) { // 수행할 서비스 종류 받는다
100         perror("recv");
101         exit(1);
102     }
103
104     if( strcmp(type, "1") == 0) { // send_board를 실행
105         send_board(ns);
106     } else if(strcmp(type, "2") == 0) { // fix_board를 실행
107         fix_board(ns);
108     } else if (strcmp(type, "3") == 0) { // rotate_board를 실행
109         rotate_board(ns);
110     } else if (strcmp(type, "4") == 0) { // is_finish 확인
111         is_end = is_finish(ns);
112         if(is_end != 0) printf("패배\n");
113     } else if (strcmp(type, "5") == 0) { // 클라이언트측 턴이 끝나고 서버 측 턴
114         is_end = my_turn(ns, 'X');
115         if(is_end != 0) printf("승리\n");
116     }
117 }
118
119 end_time = time(NULL); // 게임 끝나는 시각
120 play_time = end_time - start_time; // 총 게임 시간
121 printf("플레이 시간 : %02d:%02d:%02d\n", (play_time) / 3600, (play_time / 60) % 60,
play_time % 60);
122
123 close(ns); // 소켓을 닫음
124 close(sd);
125 close(fd); // 파일 디스크립터 닫음
126 return 0;
127 }

```

36: SIGNIT(Ctrl + C) 시그널 무시

37 ~ 44: 지역 변수 선언

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

49 ~ 57: 기보를 저장하기 위해 현재 시간을 파일명으로 하여Pentagologs 폴더에 생성
59 ~ 62: 소켓 생성
64 ~ 73: 소켓 포트 및 연결 IP 지정
75 ~ 81: 클라이언트 연결 기다리기
83 ~ 86: 클라이언트 연결 요청 수락하기
88 ~ 92: 게임을 시작, 보드 상태 초기화하기
94: 게임의 시작 시간을 start_time에 저장
95 ~ 117: 게임 플레이 과정
96 ~ 97: 화면 clear 후 보드 출력
99 ~ 102: 클라이언트로부터 수행할 type 받아오기
104 ~ 116: type에 따라 알맞은 함수 실행
type 1 -> 게임 판 보내기
type 2 -> 판 수정하기
type 3 -> 판 돌리기
type 4 -> 게임이 끝났는지 확인
type 5 -> 클라이언트가 턴 넘기면 실행
119: 게임 종료 시간을 end_time에 저장
120 ~ 121: 게임 시간 계산 후 출력
123 ~ 126: 종료

보드 초기화 함수

```
129 // 보드(판)을 ' '로 초기화 해주는 함수
130 void init_board() {
131     for (int i = 0; i < 6; i++ )
132         for(int j = 0; j < 6; j++ )
133             arr[i][j] = ' ';
134 }
```

130 ~ 133: 보드를 표시할 배열을 ' '으로 초기화

보드 현재 상태 출력 함수

```
136 // 보드의 현재 상태를 출력해주는 함수
137 // 가로, 세로축에 A~F, 1~6 을 추가로 출력해준다.
138
139 void print_board() {
140     printf(" | A | B | C | D | E | F |\n");
141     printf("-----|\n");
142     int i=0, j=0;
143     for(int l = 0; l < 6; l++) {
144         printf("%d|", l+1);
145         for (int m = 0; m < 6; m++) {
146             printf(" %c |", arr[l][m]);
```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

147     }
148     printf("\n-+-+-+-+\n");
149 }
150 }

```

140 ~ 150: 보드의 현재 상태를 출력한다

가로: A ~ F / 세로: 1 ~ 6

클라이언트에게 보드 보내는 함수

```

152 void send_board(int ns) {
153     char buf[365];
154     memset(buf, 0, sizeof(buf));
155     int i, j;
156     print_board();
157     for(i = 0; i < 14; i++) {
158         for(j = 0; j < 26; j++) {
159             if ( i == 0 ) {
160                 if ( j % 2 == 0) buf[26*i+j] = ' ';
161                 else if ( j % 4 == 1) buf[26*i+j] = '|';
162                 else if ( j % 4 == 3) buf[26*i+j] = j/4 + 'A';
163             }
164             else if (i % 2 == 1) {
165                 if ( j % 4 == 1) buf[26*i+j] = '+';
166                 else buf[26*i+j] = '-';
167             }
168             else {
169                 if (j==0) buf[26*i+j] = i/2+'0';
170                 else if ( j % 4 == 1) buf[26*i+j] = '|';
171                 else if ( j % 2 == 0) buf[26*i+j] = ' ';
172                 else buf[26*i+j] = arr[i/2-1][j/4];
173             }
174         }
175     }
176     buf[364] = '\0';
177     if(send(ns, buf, strlen(buf) + 1, 0) == -1) {
178         perror("send");
179         exit(1);
180     }
181     buf[364] = '\n';
182     buf[365] = '\0';
183     write(fd, buf, 365); // 로그로 쓴다.
184 }

```

153 ~ 154: 클라이언트에게 보낼 보드 정보 저장할 buf 배열 선언 후 0으로 초기화

156: 현재 보드 출력

157 ~ 175: buf 배열에 보드의 전체적인 모양을 저장

177 ~ 179: 클라이언트에게 보드 배열 보내기

181 ~ 183: 보드 배열을 로그로 씀

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

보드에 돌을 놓는 함수

```

186 // 보드에 돌을 놓는 함수
187 // row, col 에다가 dol을 놓는다.
188 void fix_board(int ns) {
189     char rowcol[4];
190     int row, col;
191     if(send(ns, "OK", strlen("OK") + 1, 0) == -1) { // 의미 없음
192         perror("send");
193         exit(1);
194     }
195
196     if (recv(ns, rowcol, sizeof(rowcol), 0) == -1) {
197         perror("recv");
198         exit(1);
199     }
200
201     row = rowcol[1] - '0' - 1;
202     col = rowcol[0] - 'A';
203
204     printf("%d %d\n", row, col);
205     if (arr[row][col] == ' ') {
206         arr[row][col] = rowcol[2];
207         if(send(ns, "0", strlen("0") + 1, 0) == -1) {
208             perror("send");
209             exit(1);
210         }
211     } else {
212         if(send(ns, "-1", strlen("-1") + 1, 0) == -1) {
213             perror("send");
214             exit(1);
215         }
216     }
217 }

```

189 ~ 190: 변수 선언

191 ~ 194: 클라이언트에게 OK 보내기(의미 없음)

196 ~ 199: 클라이언트로부터 받은 위치 정보 rowcol 배열에 저장

201 ~ 202: 위치 정보를 int형으로 변환하여 저장

204 ~ 217: 위치 정보를 게임의 위치 배열에 저장 후 클라이언트에게 성공 시 1, 실패 시 -1 전송

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

보드의 사분면 회전 함수

```

219 // 보드의 한 사분면을 회전하는 함수, is_clock_wise 가 y이거나 Y이면 시계방향
회전이다.
220 // 화면 출력 기준으로
221
222 //   1 2
223 //   3 4
224
225 // 사분면이다.
226 void rotate_board(int ns) {
227     int row, col;
228     char qc[3];
229     if(send(ns, "1", strlen("1") + 1, 0) == -1) { // 의미 없음
230         perror("send");
231         exit(1);
232     }
233
234     if (recv(ns, qc, sizeof(qc), 0) == -1) {
235         perror("recv");
236         exit(1);
237     }
238     if (qc[0] == '1') {
239         row = 0;
240         col = 0;
241     } else if (qc[0] == '2') {
242         row = 0;
243         col = 3;
244     } else if (qc[0] == '3') {
245         row = 3;
246         col = 0;
247     } else if (qc[0] == '4') {
248         row = 3;
249         col = 3;
250     }
251     for (int i = 0; i < qc[1] - '0'; i++) {
252         char tmp = arr[0+row][0+col];
253         arr[0+row][0+col] = arr[2+row][0+col];
254         arr[2+row][0+col] = arr[2+row][2+col];
255         arr[2+row][2+col] = arr[0+row][2+col];
256         arr[0+row][2+col] = tmp;
257         tmp = arr[0+row][1+col];
258         arr[0+row][1+col] = arr[1+row][0+col];
259         arr[1+row][0+col] = arr[2+row][1+col];
260         arr[2+row][1+col] = arr[1+row][2+col];
261         arr[1+row][2+col] = tmp;
262     }
263
264     if(send(ns, "1", strlen("1") + 1, 0) == -1) { // 의미 없음
265         perror("send");
266         exit(1);

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

267 }
268
269 }

227 ~ 228: 변수 선언

229 ~ 232: 클라이언트에게 "1" 보내기(의미 없음)

234 ~ 237: 클라이언트로부터 회전할 사분면 및 방향 정보 받아오기

238 ~ 262: 받아온 정보에 따라 게임 판 배열을 회전

264 ~ 267: 클라이언트에게 "1" 보내기(의미 없음)

클라이언트가 턴을 넘기면 서버가 게임을 진행하는 함수

```

271 // 클라이언트가 턴을 넘기면, 서버가 게임을 진행하는 함수
272 // 순서는 클라이언트 코드랑 같다.
273 // 돌 놓기 -> 돌리기 -> 검증 -> 결과 반환
274
275 int my_turn(int ns, char dol) {
276     char str[2];
277     int ret = check_pentago();
278     char x, y, quad, c;
279
280     printf("좌표 (ex, A1) : ");
281     while(1) {
282         __fpurge(stdin);
283         x = getc(stdin);
284         y = getc(stdin);
285         __fpurge(stdin);
286         if (( (x >= 'A' && x <= 'F') || (x >= 'a' && x <= 'f'))
287             && y >= '1' && y <= '6') break;
288         printf("잘못 입력하셨습니다. 다시 입력하세요 :");
289     }
290     if (x >= 'a' && x <= 'f') x -= 32; // 'a' - 'A' = 32 소문자를 대문자로
291     while (my_fix_board(x-'A', y-'0' - 1, dol)!=0) {
292         printf("이미 두신곳에 두셨습니다. 다시 두세요 :");
293         while(1) {
294             __fpurge(stdin);
295             x = getc(stdin);
296             y = getc(stdin);
297             __fpurge(stdin);
298             if (( (x >= 'A' && x <= 'F') || (x >= 'a' && x <= 'f')) && y >= '1' && y <=
299                 '6') break;
300             printf("잘못 입력하셨습니다. 다시 입력하세요 :");
301         }
302     }
303     system("clear");
304     print_board();
305     if (ret = check_pentago()) {
306         str[0] = ret + '0';

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

307     str[1] = '\0';
308     if(send(ns, str, strlen(str) + 1, 0) == -1) {
309         perror("send");
310     }
311     if( ret == 1 ) {
312         return 1; // 게임 끝
313     }
314     else return 0; // 게임 계속 진행
315 };
316
317 printf("
318 | 1 | 2 |
319 |---|
320 | 3 | 4 |
321 |---|
322 회전할 사분면\n");
323
324 while (1) {
325     __fpurge(stdin);
326     quad = getc(stdin);
327     __fpurge(stdin);
328     if (quad >= '1' && quad <= '4') break;
329     printf("잘못 입력하셨습니다. 다시 입력하세요 :");
330 }
331
332 printf("시계방향?(y/n)\n");
333 while (1) {
334     __fpurge(stdin);
335     c = getc(stdin);
336     __fpurge(stdin);
337     if (c == 'y' || c == 'Y' || c == 'n' || c == 'N') break;
338     printf("잘못 입력하셨습니다. 다시 입력하세요 :");
339 }
340 if (c == 'y' || c == 'Y') my_rotate_board(quad - '0', 1);
341 else my_rotate_board(quad - '0', 3);
342
343 system("clear");
344 print_board();
345 ret = check_pentago();
346
347 str[0] = ret + '0';
348 str[1] = '\0';
349 if(send(ns, str, strlen(str) + 1, 0) == -1) {
350     perror("send");
351 }
352 if( ret == 1 ) {
353     return 1; // 게임 끝
354 }
355 else return 0; // 게임 계속 진행
356 }

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

276 ~ 278: 지역 변수 선언

280 ~ 301: 사용자로부터 돌을 놓을 좌표 입력받음

가로 A ~ F(a ~ f), 세로 1 ~ 6 의 정보만을 입력받음

303 ~ 315: 보드를 출력한 후, check_pentago 함수를 통해 돌이 5개 이어졌는지 확인

만약 check_pentago 함수의 반환값이 ret이 1이면 클라이언트에게 1 보내

ret이 1이면 게임 끝, 아니면 게임을 계속 진행한다.

317 ~ 322: 사분면 정보 출력

324 ~ 330: 입력받은 사분면 정보(1 ~ 4)를 quad에 저장

332 ~ 341: 판을 시계/반시계 중 어느 방향으로 돌릴지 입력 후 my_rotate_board 함수 호출

343 ~ 344: 화면 초기화 후 게임 판 출력

345 ~ 356: check_pentago()를 호출하여 게임의 진행 여부 파악 후 클라이언트에게 결과값 보내기

ret값이 1이면 게임 끝, 아니면 계속 진행

게임 판에 돌 놓는 함수

```

358 // col, row 위치에 dol을 놓는다.
359 int my_fix_board(int col, int row, char dol) {
360
361     if (arr[row][col] == ' ') {
362         arr[row][col] = dol;
363         return 0;
364     } else return -1;
365 }

```

361 ~ 364: 돌을 두려는 위치에 돌이 없으면 돌을 두고, 있으면 -1 리턴

3x3 판 돌리기 함수

```

367 // quad 사분면에 c 번 90도 회전한다.,
368 void my_rotate_board(int quad, int c) {
369     int row, col;
370     if (quad == 1) {
371         row = 0;
372         col = 0;
373     } else if (quad == 2) {
374         row = 0;
375         col = 3;
376     } else if (quad == 3) {
377         row = 3;
378         col = 0;
379     } else if (quad == 4) {
380         row = 3;

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

381     col = 3;
382 }
383 for (int i = 0; i < c ; i++) {
384     char tmp = arr[0+row][0+col];
385     arr[0+row][0+col] = arr[2+row][0+col];
386     arr[2+row][0+col] = arr[2+row][2+col];
387     arr[2+row][2+col] = arr[0+row][2+col];
388     arr[0+row][2+col] = tmp;
389     tmp = arr[0+row][1+col];
390     arr[0+row][1+col] = arr[1+row][0+col];
391     arr[1+row][0+col] = arr[2+row][1+col];
392     arr[2+row][1+col] = arr[1+row][2+col];
393     arr[1+row][2+col] = tmp;
394 }
395 }

```

370 ~ 394: 사분면을 선택 후 시계/반시계 방향으로 돌리기

게임 종료 체크 함수

```

397 // 게임이 끝나면 1을 반환한다.
398 int is_finish(int ns) {
399     char str[2];
400     int ret = check_pentago();
401     str[0] = ret + '0';
402     str[1] = '\0';
403     if(send(ns, str, strlen(str) + 1, 0) == -1) {
404         perror("send");
405     }
406     if( ret == 1 ) {
407         return 1; // 게임 끝
408     }
409     else return 0; // 게임 계속 진행
410 }

```

399: 지역 배열 선언

400: check_pentago()를 호출하여 돌이 5개 이어졌는지 체크

401 ~ 405: 클라이언트에게 check_pentago() 반환값 전달

406 ~ 409: ret가 1이면 게임 종료, 아니면 게임 계속 진행

5개의 돌이 이어졌는지 체크하는 함수

```

413 //5개의 돌이 이어졌는지 체크한다.
414 int check_pentago() {
415     int count = 0; //5개 이어졌는지 체크
416     int isNull = 0; // ' '가 있는지 체크

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

417 //가로
418 for(int i = 0 ; i < 6 ; i++) {
419     for(int j = 1 ; j < 5 ; j++) {
420         if(arr[i][j] == ' ') {
421             isNull = 1;
422             break;
423         }
424     }
425     if((isNull != 1 && arr[i][0] != ' ') || (isNull != 1 && arr[i][5] != ' ')) {
426         for(int j = 0 ; j < 6 ; j += 5) {
427             for(int k = 1 ; k < 5 ; k++) {
428                 if(arr[i][j] == arr[i][k]) {
429                     count++;
430                 }
431             }
432             if(count == 4) {
433                 return 1;
434             }
435             else {
436                 count = 0 ;
437             }
438         }
439     }
440     isNull = 0;
441 }
442
443 //세로
444 count = 0;
445 isNull = 0;
446 for(int i = 0 ; i < 6 ; i++) {
447     for(int j = 1 ; j < 5 ; j++) {
448         if(arr[j][i] == ' ') {
449             isNull = 1;
450             break;
451         }
452     }
453     if((isNull != 1 && arr[0][i] != ' ') || (isNull != 1 && arr[5][i] != ' ')) {
454         for(int j = 0 ; j < 6 ; j += 5 ) {
455             for(int k = 1 ; k < 5 ; k++) {
456                 if(arr[j][i] == arr[k][i]) {
457                     count++;
458                 }
459             }
460             if(count == 4) {
461                 return 1;
462             }
463             else {
464                 count = 0;
465             }
466         }

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

467     }
468     isNull = 0;
469 }
470
471 //왼쪽 위 오른쪽 아래 1
472 count = 0;
473 isNull = 0;
474 {
475     for(int i = 1 ; i < 5 ; i++) {
476         if(arr[i][i] == ' ') {
477             isNull = 1;
478             break;
479         }
480     }
481     if((isNull != 1 && arr[0][0] != ' ') || (isNull != 1 && arr[5][5] != ' ')) {
482         for(int i = 0 ; i < 6 ; i += 5) {
483             for(int j = 1 ; j < 5 ; j++) {
484                 if(arr[i][i] == arr[j][j]) {
485                     count++;
486                 }
487             }
488             if(count == 4) {
489                 return 1;
490             }
491             else {
492                 count = 0 ;
493             }
494         }
495     }
496     isNull = 0;
497 }
498 //왼쪽 아래 오른쪽 위 2
499 count = 0;
500 isNull = 0;
501 {
502     for(int i = 1, j = 4 ; i < 5 ; i++, j--) {
503         if(arr[i][j] == ' ') {
504             isNull = 1;
505             break;
506         }
507     }
508     if((isNull != 1 && arr[0][5] != ' ') || (isNull != 1 && arr[5][0] != ' ')) {
509         for(int i = 0 , j = 5 ; i < 6 ; i += 5, j -= 5) {
510             for(int k = 1, l = 4 ; k < 5 ; k++, l--) {
511                 if(arr[i][j] == arr[k][l]) {
512                     count++;
513                 }
514             }
515             if(count == 4) {
516                 return 1;

```


| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

517     }
518     else {
519         count = 0;
520     }
521 }
522 }
523 isNull = 0;
524 }
525 //왼쪽 위 오른쪽 아래 4개 위 3
526 count = 0;
527 if(arr[0][1] != ' ') {
528     for(int i = 1 ; i < 5 ; i++) {
529         if(arr[0][1] == arr[i][i+1]) {
530             count++;
531         }
532     }
533     if(count == 4) {
534         return 1;
535     }
536     else {
537         count = 0 ;
538     }
539 }
540 //왼쪽 위 오른쪽 아래 4개 아래 4
541 count = 0;
542 if(arr[1][0] != ' ') {
543     for(int i = 1 ; i < 5 ; i++) {
544         if(arr[1][0] == arr[i+1][i]) {
545             count++;
546         }
547     }
548     if(count == 4) {
549         return 1;
550     }
551     else {
552         count = 0;
553     }
554 }
555 //왼쪽 아래 오른쪽 위 4개 위 5
556 count = 0;
557 if(arr[0][4] != ' ') {
558     for(int i = 1, j = 3; i < 5 ; i++, j--) {
559         if(arr[0][4] == arr[i][j]) {
560             count++;
561         }
562     }
563     if(count == 4) {
564         return 1;
565     }
566     else {

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

567     count = 0;
568 }
569 }
570 //왼쪽 아래 오른쪽 위 4개 아래 6
571 count = 0;
572 if(arr[1][5] != ' ') {
573     for(int i = 2, j = 4 ; i < 6 ; i++, j--) {
574         if(arr[1][5] == arr[i][j]) {
575             count++;
576         }
577     }
578     if(count == 4) {
579         return 1;
580     }
581     else {
582         count = 0;
583     }
584 }
585 return 0;
586 }

```

415 ~ 416: 변수 선언

417 ~ 441: 가로 방향으로 5개의 돌이 이어졌는지 체크

444 ~ 469: 세로 방향으로 5개의 돌이 이어졌는지 체크

471 ~ 497: 왼쪽 위 오른쪽 아래의 대각선으로 5개의 돌이 이어졌는지 체크

498 ~ 524: 왼쪽 아래 오른쪽 위의 대각선으로 5개의 돌이 이어졌는지 체크

525 ~ 539: 왼쪽 위 오른쪽 아래의 대각선에서 아랫줄로 5개의 돌이 이어졌는지 체크

540 ~ 554: 왼쪽 위 오른쪽 아래의 대각선에서 위의 줄로 5개의 돌이 이어졌는지 체크

555 ~ 569: 왼쪽 아래 오른쪽 위의 대각선에서 위의 줄로 5개의 돌이 이어졌는지 체크

570 ~ 584: 왼쪽 아래 오른쪽 위의 대각선에서 아랫줄로 5개의 돌이 이어졌는지 체크

5개의 돌이 이어졌으면 return 1, 아니면 return 0;

실시간 플레이 시간 기능을 추가할 시,

```

1  #include <pthread.h>
2  pthread_mutex_t mutex; // 쓰레드 뮤텝스
3  void* print_playtime(); // 쓰레드 실행할 함수, 시간을 출력하는 역할을 한다.
4
5  int main(void) {
6      int play_time;
7      pthread_t t1; // 쓰레드
8      rcc = pthread_create(&t1, NULL, print_playtime, NULL);
9      int rcc, t = 1; // 쓰레드
10
11 }
12
13 void *print_playtime() {
14     time_t start_time = time(NULL);

```

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

```

15  time_t now;
16  while (1) {
17      now = time(NULL);
18      pthread_mutex_lock(&mutex);
19      printf("\33[%dd\033[%dG", 0, 30);
20      printf("%02d : %02d", (now - start_time)/60, (now - start_time)%60);
21      sleep(0.9);
22      pthread_mutex_unlock(&mutex);
23  }
24 }

```

1: 헤더 파일 추가

2: 전역 변수 선언

3: 함수 원형 선언

6 ~ 9: 지역 변수 선언, pthread 생성

14 ~ 15: 현재 플레이 시간을 출력하는 함수 내의 지역 변수 선언

16 ~ 23: 현재 시간을 mutex를 걸어주며 출력

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

1.3 로그 뷰어

프로그램 인자로 주어진 로그 파일명을 출력해주는 프로그램

| 헤더 파일 |
|---|
| <pre> 1 #include <sys/types.h> 2 #include <sys/stat.h> 3 #include <fcntl.h> 4 #include <unistd.h> 5 #include <stdio.h> 6 #include <stdlib.h> </pre> |
| 1 ~ 6: 해당 파일에서 필요한 헤더 파일을 추가한다. |

| 변수 선언 및 File Open |
|---|
| <pre> 9 int main(int argc, char* argv[]) { 10 char buf[366]; 11 int fd, n; 12 int cnt = 1; 13 fd = open(argv[2], O_RDONLY); 14 if (fd == 0) { 15 perror(argv[2]); 16 exit(2); 17 } </pre> |
| 10 ~ 16: 변수 선언 및 main에서 인자로 받아들인 파일 열기 만약 파일을 열지못하면 에러를 출력하고 종료 |

| 기보 출력 |
|--|
| <pre> 19 while ((n = read(fd, buf, sizeof(buf))) > 1) { 20 printf("%d\n", ++cnt); 21 int i, j; 22 for(i = 0; i < 14; i++) { 23 for(j = 0; j < 26; j++) { 24 printf("%c", buf[26*i+j]); 25 } 26 printf("\n"); 27 } 28 printf("\n\n"); 29 } 30 31 close(fd); 32 return 0; 33 } </pre> |
| 19 ~ 29: open한 파일로부터 buf 크기만큼 읽어오며 화면에 출력 |

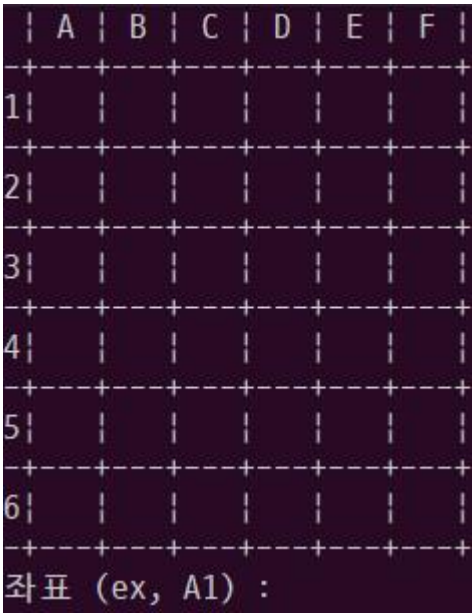
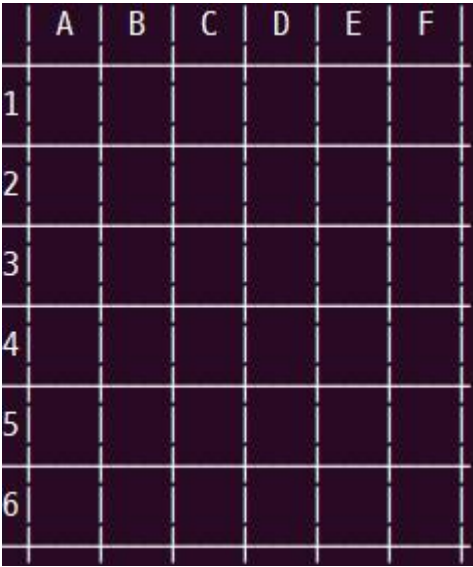
| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

2. PENTA-GO 사용지침서

게임 방법

1. 준비

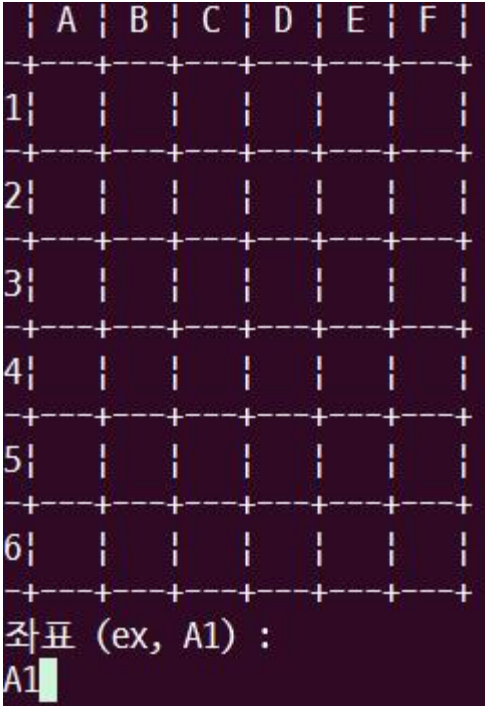
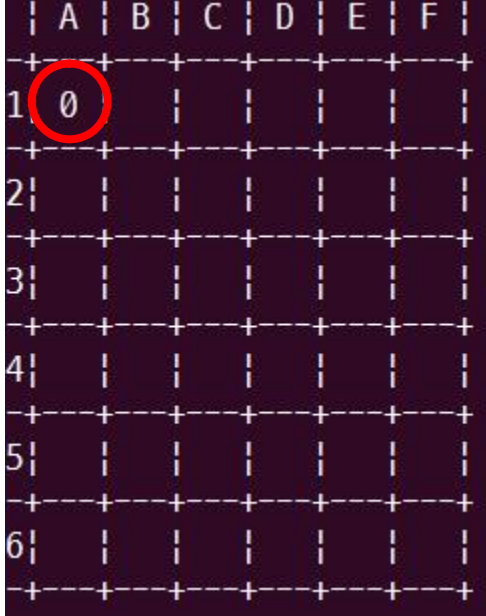
- 서버측 이 PentaGo-Server 프로그램을 실행시키고 대기한다.
- 클라이언트가 PentaGo-Client 프로그램을 실행하여 접속한다.

| | |
|--|---|
| <Client> | <server> |
| | 도전자를 기다리는중... |
| | 1. 서버가 클라이언트의 접속을 대기하는 중이다. |
| ./PentaGo-Client | 게임이 시작됩니다. |
| 2. 클라이언트 프로그램을 구동시킨다. | 3. 서버에서는 클라이언트의 접속이 확인된다. |
|  |  |
| 4. 클라이언트측 화면이다. 클라이언트가 선이어서 입력화면이 같이 출력되었다. | 5. 서버측 대기 화면이다. |

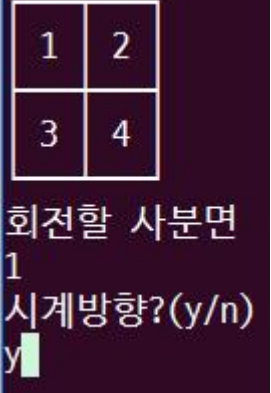
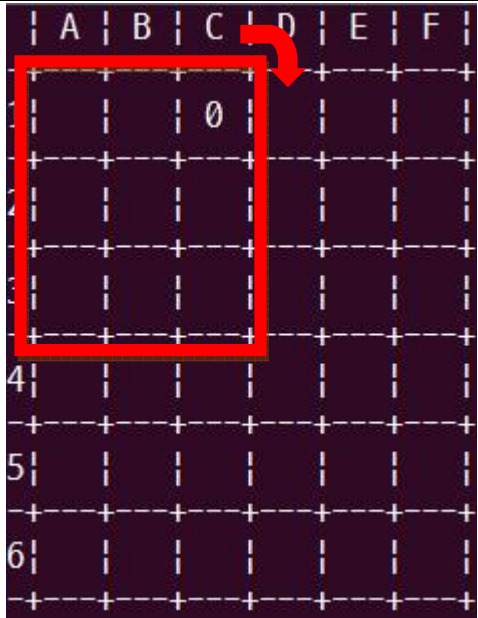
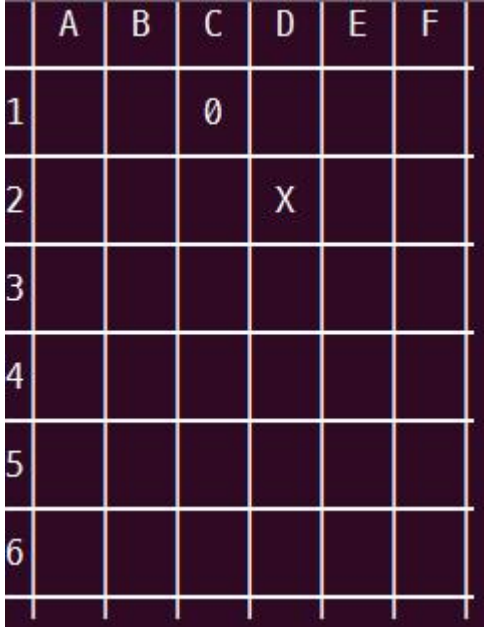
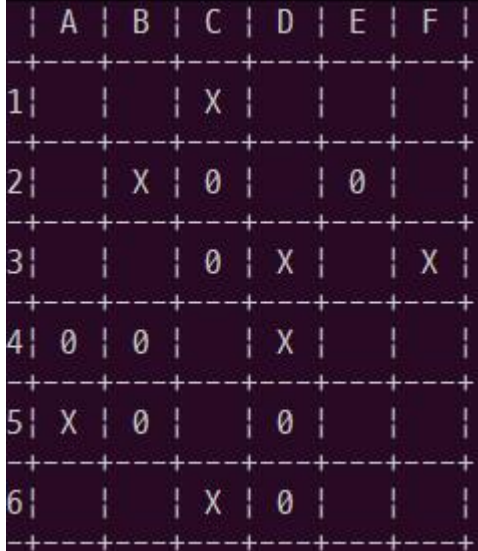
| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

2. 진행

- 클라이언트부터 돌을 둔다.
(클라이언트는 '0'를 판에 놓게 된다. 서버는 'X' 모양)
- 돌을 두는 규칙은 비어 있는 보드 위에 돌을 둔 후에 3x3 보드 중 하나를 선택하여 90도를 시계방향 또는 반시계방향으로 회전 시킨다.

| | |
|--|---|
|  <p>좌표 (ex, A1) : A1</p> |  |
| <p>1. 현재 판의 모습이 출력되면, 좌표를 입력한다.</p> | <p>2. 좌표를 올바르게 입력하면 다음과 같이 보드에 반영이 된다.</p> |

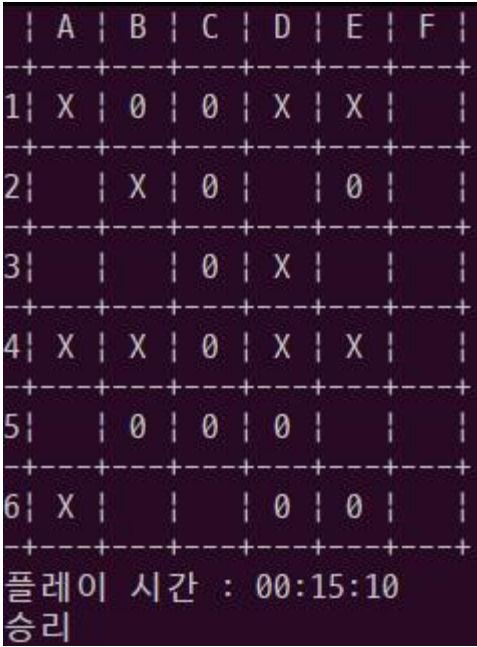
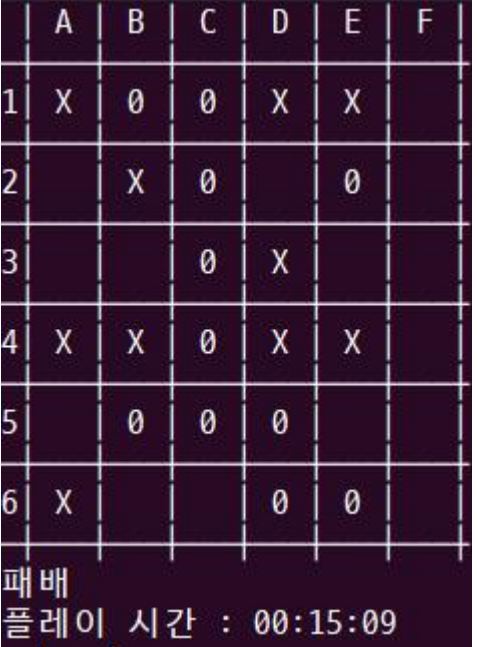
| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

| | |
|--|--|
|  |  |
| <p>3. 9X9 보드를 위와 같이 3X3의 서브 보드 4개로 보고, 1개의 서브보드를 택하여 시계방향(y) 또는 반시계 방향(n)으로 90도 돌린다.</p> | <p>4. 다음과 같이 보드에 반영이 된다. 사진의 예시처럼 1번 서브보드가 시계 방향으로 90도 돌아간 것을 확인할 수 있다.</p> |
|  |  |
| <p>5. 서버측도 마찬가지로 진행하면 된다.</p> | <p>6. 다음과 같이 계속 진행한다.</p> |

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

3. 승리 조건

- 돌이 이어져서 5개가 되면 게임은 끝나게 된다.

| | |
|--|---|
|  <p>플레이 시간 : 00:15:10 승리</p> |  <p>패배 플레이 시간 : 00:15:09</p> |
| <p>1. 플레이 중 [C1 ~ C5] 처럼 5개의 연속된 모양이 나오면 게임이 중단되고, 승패가 결정되게 된다.</p> | |

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

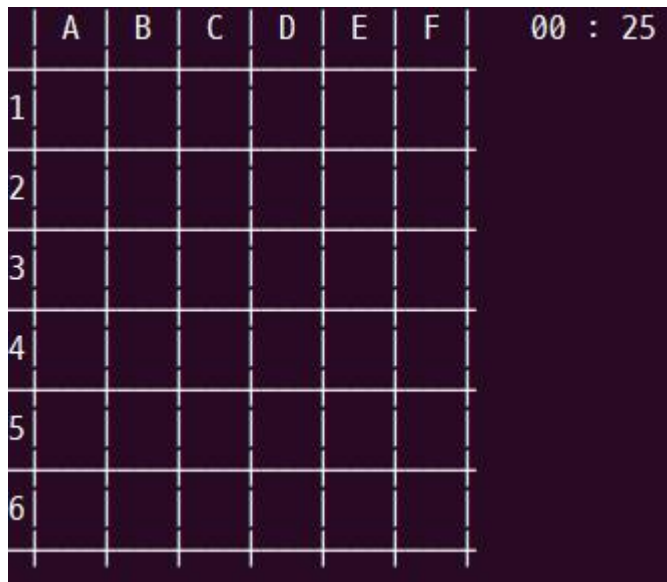
4. 기보 보기

- 플레이 한 기록은 서버의 ./Pentagologs 디렉토리에 플레이 시작한 시각의 이름으로 저장되며, log_viewer 프로그램을 이용하여 복원 할 수 있다.
- 명령 인자로 파일 이름을 입력하면 된다.

| | |
|---|--|
| <pre>ls ./Pentagologs 20191205_01_05_39.txt 20191205_01_06_54.txt 20191205_01_09_01.txt</pre> | <pre>../log_viewer 20191206_00_23_47.txt more</pre> |
| 로그가 저장되어있다. | 이와 같이 기보 보기 프로그램을 실행시킨다. |
| | <pre>32 A B C D E F +---+---+---+---+---+---+ 1 X 0 0 X X +---+---+---+---+---+---+ 2 X 0 0 +---+---+---+---+---+---+ 3 0 X +---+---+---+---+---+---+ 4 X X 0 X X +---+---+---+---+---+---+ 5 0 0 0 </pre> |
| | 이런 식으로 기보를 볼 수 있다. |

| | | | |
|--------|----------|---------|-------------------------|
| 프로젝트 명 | Penta-Go | 프로젝트 기간 | 2019.11.26.~2019.12.09. |
| 문 서 명 | 프로젝트 매뉴얼 | 버전 | Ver 1.0.0 |

추가적으로 PentaGo-Server-Thread 프로그램을 실행시키면 게임 진행 시간을 알 수 있다.



PentaGo-Server-thread를 실행시키면 다음과 같이 진행 시각도 표시된다.