Tetris

0.0.1

다음에 의해 생성됨 : Doxygen 1.9.1

1 HM2J_c 1
1.1 Table of Contents (목차)
1.2 Requirements
1.3 Usage
1.3.1 Git
1.3.1.1 Commands (명령어)
1.3.1.2 Git으로 프로젝트 관리하기
1.4 TODO
2 클래스 색인
2.1 클래스 목록
3 파일 색인 7
3.1 파일 목록
4 클래스 문서화
· · · · · · · · · · · · · · · · · · ·
4.1.2 멤버 데이터 문서화
4.1.2.1 day
4.1.2.2 hour
4.1.2.3 min
4.1.2.4 month
4.1.2.5 name
4.1.2.6 point
4.1.2.7 year
5 파일 문서화 11
5.1 include/db.h 파일 참조
5.1.1 함수 문서화
5.1.1.1 close_db()
5.1.1.2 drop_table()
5.1.1.3 init_db()
5.1.1.4 read_db()
5.1.1.5 write_db()
5.2 include/display_tetris.h 파일 참조
5.2.1 매크로 문서화
5.2.1.1 BLUE_COLOR
5.2.1.2 CYAN_COLOR
5.2.1.3 GREEN_COLOR
5.2.1.4 MAGENTA_COLOR
5.2.1.5 RED_COLOR
5.2.1.6 RESET_COLOR
5.2.1.7 WHITE_COLOR

5.2.1.8 YELLOW_COLOR	18
5.2.2 변수 문서화	19
5.2.2.1 block	19
5.2.2.2 block_state	19
5.2.2.3 tetris_table	19
5.3 README.md 파일 참조	20
5.4 src/db.c 파일 참조	20
5.4.1 상세한 설명	21
5.4.2 매크로 문서화	21
5.4.2.1 ERR	21
5.4.2.2 OK	21
5.4.3 함수 문서화	22
5.4.3.1 close_db()	22
5.4.3.2 create_table()	22
5.4.3.3 drop_table()	23
5.4.3.4 get_one_row()	24
5.4.3.5 init_db()	25
5.4.3.6 print_result()	26
5.4.3.7 read_db()	26
5.4.3.8 test_db()	27
5.4.3.9 use_db()	28
5.4.3.10 write_db()	29
5.4.4 변수 문서화	30
5.4.4.1 conn	30
5.4.4.2 db	30
5.4.4.3 host	30
5.4.4.4 passwd	30
5.4.4.5 port	30
5.4.4.6 tmp	31
5.4.4.7 user	31
5.5 src/display_menu.c 파일 참조	31
5.5.1 함수 문서화	31
5.5.1.1 display_menu()	32
5.6 src/display_tetris.c 파일 참조	32
5.6.1 함수 문서화	33
5.6.1.1 display_tetris()	33
5.6.2 변수 문서화	35
5.6.2.1 i_block	35
5.6.2.2 j_block	35
5.6.2.3 l_block	35
5.6.2.4 next_block_number	35
5.6.2.5 o_block	36

5.6.2.6 s_block	36
5.6.2.7 t_block	36
5.6.2.8 z_block	36
5.7 src/getch.c 파일 참조	36
5.7.1 매크로 문서화	37
5.7.1.1 CCHAR	37
5.7.1.2 CTIME	37
5.7.1.3 FLAG	37
5.7.2 함수 문서화	37
5.7.2.1 getch()	38
5.8 src/main.c 파일 참조	38
5.8.1 상세한 설명	40
5.8.2 매크로 문서화	40
5.8.2.1 GAME_END	40
5.8.2.2 GAME_START	40
5.8.3 함수 문서화	40
5.8.3.1 display_menu()	41
5.8.3.2 display_tetris()	41
5.8.3.3 game_start()	43
5.8.3.4 init_tetris_table()	44
5.8.3.5 main()	45
5.8.3.6 print_color()	45
5.8.3.7 print_result()	46
5.8.3.8 save_result()	46
5.8.3.9 update()	46
5.8.4 변수 문서화	48
5.8.4.1 game	48
5.8.4.2 point	48
5.8.4.3 q	48
5.8.4.4 tetris_table	48
5.8.4.5 x	49
5.8.4.6 y	49
5.9 src/new.c 파일 참조	49
5.9.1 함수 문서화	49
5.9.1.1 chooseDoll()	50
5.9.1.2 getRandomScore()	51
5.9.1.3 new()	51
5.10 src/result.c 파일 참조	52
5.11 src/update.c 파일 참조	52
5.11.1 매크로 문서화	53
5.11.1.1 GAME_END	53
5.11.1.2 GAME_START	53

Index

5.1	1.2 열거형 타입 문서화	53
	5.11.2.1 COMMAND	53
5.1	1.3 함수 문서화	54
	5.11.3.1 check_one_line()	54
	5.11.3.2 collision_test()	55
	5.11.3.3 display_tetris()	56
	5.11.3.4 drop()	57
	5.11.3.5 getch()	58
	5.11.3.6 move_block()	59
	5.11.3.7 update()	61
5.1	1.4 변수 문서화	62
	5.11.4.1 block	63
	5.11.4.2 block_number	63
	5.11.4.3 block_state	63
	5.11.4.4 game	63
	5.11.4.5 i_block	63
	5.11.4.6 j_block	64
	5.11.4.7 I_block	64
	5.11.4.8 next_block_number	64
	5.11.4.9 o_block	65
	5.11.4.10 point	65
	5.11.4.11 s_block	65
	5.11.4.12 t_block	66
	5.11.4.13 tetris_table	66
	5.11.4.14 x	66
	5.11.4.15 y	66
	5.11.4.16 z_block	67

69

Chapter 1

HM2J_c

C + MySQL

1.1 Table of Contents (목차)

- Requirements (필요사항)
- Usage (사용법)
- TODO (할일)

1.2 Requirements

To clone this repo git clone https://github.com/skoler/HM2J_c.git

sudo snap install git

• Doxygen >= 1.9.1

For documentation

sudo apt install doxygen doxygen-gui graphviz brew install --cask doxygen $\,$

• CMake >= 3.10
To build this project

sudo apt install cmake
brew install cmake

• MySQL >= 8.3.0 To manage data by Database sudo apt install mysql-server sudo mysql

2 HM2J_c

1.3 Usage

1.3.1 Git

1.3.1.1 Commands (명령어)

• 깃허브 원격저장소(remote repository)를 현재컴퓨터(local)에 복사(clone) 하기

git clone <url>

- 저장소 줄기(branch)들 확인하기
 - 현재 접속중인 줄기(branch)들 확인
 - 모든 줄기(branch)들 확인

git branch

• 줄기(branch) 만들기

git branch <branch>

• 줄기(branch) 삭제하기

git branch -d <branch>

• 줄기(branch) 접속(checkout)하기

git checkout <branch>

• 원격저장소 origin의 파일들을 현재컴퓨터에 가져와,덮어쓰기(pull) - 주의 기존파일과 충돌(conflict)할 수 있음

git pull origin <branch>

• git이 관리할 파일 추가(add)하기

```
# 하나씩 추가하기
git add <filename>
# 현재폴더(.)에 있는 파일 모두 추가하기
# 이때 .gitignore 에 있는 파일들은 제외된다
git add .
```

• git이 관리중이던(--cached 옵션) 파일을, 더 이상 관리하지 않도록 삭제(remove: rm)하기

git rm --cached <filename>

• git이 관리중인 파일 상태(status) 확인하기

git status

• git이 관리중인 파일들의 상태를, 설명(-m 옵션)과 함께 세이브, 저장(commit)하기

```
git commit -m "message"
```

• 줄기(branch)안에 모든 세이브 파일들(commit)을, Github 원격저장소(remote repository, name: origin)에 업로드(push)하기

git push origin <branch>

• feature/gotcha 줄기(branch)에, main 줄기를 합치기(merge). 방향 주의

```
# feature/gotcha 줄기 접속
git checkout feature/gotcha
# 현재줄기(feature/gotcha)에 main 줄기를 합치기
git merge feature/gotcha
```

1.4 TODO 3

1.3.1.2 Git으로 프로젝트 관리하기

1.3.1.2.1 Github Flow Github가 Git으로 프로젝트를 관리하는 방법

- 1. main 줄기(branch)를 만들기 실제로 사용자에게 배포(release)하는 줄기
 - 이 줄기에 다른 줄기를 합치려면, feature 줄기에서 합친후(merge), 테스트가 되어야한다.
- feature/기능 줄기(branch)를 만들기 새로운 기능을 추가하는 줄기
 - 새로운 기능이 필요할 때마다, 새로운 프랜치를 만든다. ex: git branch feature/gotcha

1.4 TODO

- [] Add Usage to documentation
- [] Add MySQL support

4 HM2J_c

Chapter 2

클래스 색인

2.1	클래스	모로

다음은 클래스, 구조체, 공용체 그리고 인터페이스들입니다. (간략한 설명만을 보여줍니다) :	
RESULT	ç

클래스 색인

Chapter 3

파일 색인

3.1 파일 목록

다음은 모든 파일에 대한 목록입니다. (간략한 설명만을 보여줍니다):

nclude/db.h	1
nclude/display_tetris.h	6
src/db.c	
MySQL DB Functions	
src/display_menu.c	1
src/display_tetris.c	2
src/getch.c	6
src/main.c	
테트리스 게임을 실행하는 메인 함수	8
src/new.c	.9
src/result.c	2
src/update.c	

8 파일 색인

Chapter 4

클래스 문서화

4.1 RESULT 구조체 참조

#include <db.h>

RESULT에 대한 협력 다이어그램:

RESULT

- + name
- + point
- + year
- + month
- + day
- + hour
- + min

Public 속성

- char name [30]
- int point
- int year
- int month
- int day
- int hour
- int min

4.1.1 상세한 설명

db.h 파일의 5 번째 라인에서 정의되었습니다.

10 클래스 문서화

4.1.2 멤버 데이터 문서화

4.1.2.1 day

int RESULT::day

db.h 파일의 10 번째 라인에서 정의되었습니다.

4.1.2.2 hour

int RESULT::hour

db.h 파일의 11 번째 라인에서 정의되었습니다.

4.1.2.3 min

int RESULT::min

db.h 파일의 12 번째 라인에서 정의되었습니다.

4.1.2.4 month

int RESULT::month

db.h 파일의 9 번째 라인에서 정의되었습니다.

4.1.2.5 name

char RESULT::name[30]

db.h 파일의 6 번째 라인에서 정의되었습니다.

4.1.2.6 point

int RESULT::point

db.h 파일의 7 번째 라인에서 정의되었습니다.

4.1.2.7 year

int RESULT::year

db.h 파일의 8 번째 라인에서 정의되었습니다.

이 구조체에 대한 문서화 페이지는 다음의 파일로부터 생성되었습니다.:

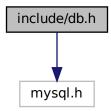
• include/db.h

Chapter 5

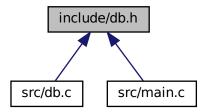
파일 문서화

5.1 include/db.h 파일 참조

#include <mysql.h> db.h에 대한 include 의존 그래프



이 그래프는 이 파일을 직/간접적으로 include 하는 파일들을 보여줍니다.:



클래스

struct RESULT

함수

```
void init_db ()
    MySQL DB를 초기화합니다
• void close_db ()
    MySQL DB를 종료합니다
void read_db ()
    MySQL DB에서 데이터를 가져올 수 있습니다
void write_db (int point)
    MySQL DB에 데이터를 저장할 수 있습니다

    int drop_table (MYSQL *conn)
```

MySQL Table을 삭제합니다

5.1.1 함수 문서화

5.1.1.1 close_db()

182 }

```
void close_db ( )
MySQL DB를 종료합니다
db.c 파일의 178 번째 라인에서 정의되었습니다.
178 {
179 if (conn != NULL) {
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.1.1.2 drop_table()

```
int drop_table (
            MYSQL * conn)
```

MySQL Table을 삭제합니다

매개변수

```
conn MySQL DB
```

db.c 파일의 124 번째 라인에서 정의되었습니다.

```
124
125
      printf("###########\n");
      printf("[drop_table] DROP TABLE log\n");
126
      // drop table
if (mysql_query(conn, "DROP TABLE IF EXISTS log;")) {
128
129
       printf("DROP TABLE Query failed\n");
        return ERR;
130
131
132
133
     return print_result(conn);
134 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



5.1.1.3 init_db()

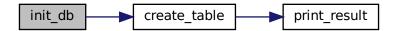
```
void init_db ( )
```

MySQL DB를 초기화합니다

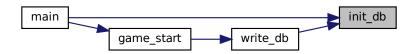
db.c 파일의 161 번째 라인에서 정의되었습니다.

```
161
          // establish connection with mysql
         // establish connection mysql
conn = mysql_init(NULL);
// if connection failed
if (!mysql_real_connect(conn, host, user, passwd, db, port, NULL, 0)) {
    fprintf(stderr, "%s\n", mysql_error(conn));
163
164
165
166
           mysql_close(conn);
167
168
            exit(1);
169
170
         // test_db(conn);
171
172
         create_table(conn);
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.1.1.4 read db()

```
void read_db ( )
```

MySQL DB에서 데이터를 가져올 수 있습니다

매개변수

name 이름을 통해 사용자 검색

반환값

RESULT 구조체 주소 or NULL을 돌려줍니다

db.c 파일의 189 번째 라인에서 정의되었습니다.

```
189
     char query[256];
printf("이름을 입력하세요: ");
scanf("%s", tmp.name);
snprintf(query, sizeof(query), "SELECT * FROM log WHERE name = '%s'", tmp.name);
if (mysql_query(conn, query)) {
190
191
192
193
194
195
      printf("SELECT Query failed\n");
196
197
198
     RESULT *res = get_one_row(conn);
     if (res == NULL) {
199
       system("clear");
200
201
       202
       printf("No History");
       printf("********************************
203
204
     } else {
205
        system("clear");
206
         printf("*******************************
         207
208
209
210
211
     }
212 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.1.1.5 write_db()

```
void write_db (
         int point )
```

MySQL DB에 데이터를 저장할 수 있습니다

매개변수

point 점수, 아이디

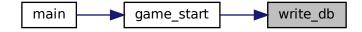
db.c 파일의 218 번째 라인에서 정의되었습니다.

```
218
        printf("이름을 입력하세요: <sup>'</sup>");
219
220
221
        scanf("%s", tmp.name);
       new(point);
tmp.point = point;
222
223
       time_t now_sec = time(NULL);
struct tm *now = localtime(&now_sec);
225
226
227
       tmp.year = now->tm_year + 1900;
tmp.month = now->tm_mon + 1;
228
229
       tmp.day = now->tm_mday;
tmp.hour = now->tm_hour;
230
232
       tmp.min = now->tm_min;
233
234
       .conn == 1
init_db();
}
       if (conn == NULL) {
235
236
238
       char query[512];
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:

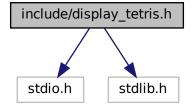


이 함수를 호출하는 함수들에 대한 그래프입니다.:

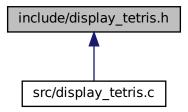


5.2 include/display_tetris.h 파일 참조

```
#include <stdio.h>
#include <stdlib.h>
display_tetris.h에 대한 include 의존 그래프
```



이 그래프는 이 파일을 직/간접적으로 include 하는 파일들을 보여줍니다.:



매크로

- #define RESET_COLOR "\x1b[0m"
- #define RED_COLOR "\x1b[31m"
- #define GREEN_COLOR "\x1b[32m"
- #define YELLOW_COLOR "\x1b[33m"
- #define BLUE_COLOR "\x1b[34m"
- #define MAGENTA_COLOR "\x1b[35m"
- #define CYAN_COLOR "\x1b[36m"
- #define WHITE_COLOR "\x1b[37m"

변수

- int block [4][4][4]
- int tetris_table [21][10]
- · int block_state

5.2.1 매크로 문서화

5.2.1.1 BLUE_COLOR

#define BLUE_COLOR "\x1b[34m"

display_tetris.h 파일의 11 번째 라인에서 정의되었습니다.

5.2.1.2 CYAN_COLOR

#define CYAN_COLOR "\x1b[36m"

display_tetris.h 파일의 13 번째 라인에서 정의되었습니다.

5.2.1.3 GREEN_COLOR

```
#define GREEN_COLOR "\x1b[32m"
```

display_tetris.h 파일의 9 번째 라인에서 정의되었습니다.

5.2.1.4 MAGENTA_COLOR

```
#define MAGENTA_COLOR "\x1b[35m"
```

display_tetris.h 파일의 12 번째 라인에서 정의되었습니다.

5.2.1.5 **RED_COLOR**

```
#define RED_COLOR "\x1b[31m"
```

display_tetris.h 파일의 8 번째 라인에서 정의되었습니다.

5.2.1.6 RESET_COLOR

```
#define RESET_COLOR "\x1b[0m"
```

display_tetris.h 파일의 7 번째 라인에서 정의되었습니다.

5.2.1.7 WHITE COLOR

```
#define WHITE_COLOR "\x1b[37m"
```

display_tetris.h 파일의 14 번째 라인에서 정의되었습니다.

5.2.1.8 YELLOW_COLOR

#define YELLOW_COLOR "\x1b[33m"

display_tetris.h 파일의 10 번째 라인에서 정의되었습니다.

5.2.2 변수 문서화

5.2.2.1 block

display_tetris.h 파일의 16 번째 라인에서 정의되었습니다.

5.2.2.2 block_state

```
int block_state [extern]
```

update.c 파일의 7 번째 라인에서 정의되었습니다.

5.2.2.3 tetris_table

```
int tetris_table[21][10]
```

display_tetris.h 파일의 34 번째 라인에서 정의되었습니다.

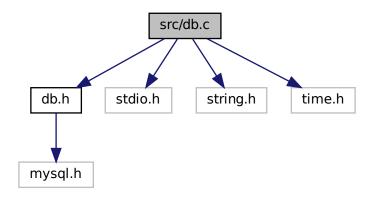
과일 문서화

5.3 README.md 파일 참조

5.4 src/db.c 파일 참조

MySQL DB Functions

```
#include "db.h"
#include <stdio.h>
#include <string.h>
#include <time.h>
db.c에 대한 include 의존 그래프
```



매크로

- #define ERR 1
- #define OK 0

함수

• int print_result (MYSQL *conn)

MySQL Result를 출력합니다

• RESULT * get_one_row (MYSQL *conn)

MySQL DB에서 log 데이터를 가져올 수 있습니다

int test_db (MYSQL *conn)

MySQL DB를 테스트합니다

• int use_db (MYSQL *conn)

MySQL DB를 사용합니다

int drop_table (MYSQL *conn)

MySQL Table을 삭제합니다

• int create_table (MYSQL *conn)

MySQL DB를 생성합니다

void init_db ()

5.4 src/db.c 파일 참조 21

```
MySQL DB를 초기화합니다

• void close_db ()

MySQL DB를 종료합니다

• void read_db ()
```

MySQL DB에서 데이터를 가져올 수 있습니다

void write_db (int point)

MySQL DB에 데이터를 저장할 수 있습니다

변수

```
• static MYSQL * conn = NULL
```

• static const char * host = "localhost"

• static const char * user = "root"

• static const char * passwd = "0000"

• static const char * db = "hm2j"

• static const int port = 3306

static RESULT tmp

5.4.1 상세한 설명

MySQL DB Functions

작성자

XXXX

날짜

2024-03-21 @detail MySQL DB와 연결하고, 데이터를 저장, 가져올 수 있습니다.

5.4.2 매크로 문서화

5.4.2.1 ERR

#define ERR 1

db.c 파일의 12 번째 라인에서 정의되었습니다.

5.4.2.2 OK

#define OK 0

db.c 파일의 13 번째 라인에서 정의되었습니다.

5.4.3 함수 문서화

5.4.3.1 close_db()

```
void close_db ( )
```

MySQL DB를 종료합니다

db.c 파일의 178 번째 라인에서 정의되었습니다.

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.2 create_table()

```
int create_table ( {\tt MYSQL} \ * \ {\tt conn} \ )
```

MySQL DB를 생성합니다

매개변수

conn | MySQL DB

db.c 파일의 140 번째 라인에서 정의되었습니다.

```
141
     142
143
144
                          "point INT,"
"year INT,"
"month INT,"
145
146
147
148
                          "day INT,"
                          "hour INT,"
"min INT);")) {
149
150
151
       printf("CREATE TABLE Query failed\n");
152
       return ERR;
153
```

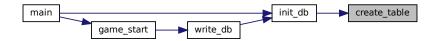
5.4 src/db.c 파일 참조 23

```
154
155    return print_result(conn);
156 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.3 drop_table()

```
int drop_table ( {\tt MYSQL} \ * \ {\tt conn} \ )
```

MySQL Table을 삭제합니다

매개변수

conn MySQL DB

db.c 파일의 124 번째 라인에서 정의되었습니다.

```
124

125  printf("#############|n");

126  printf("[drop_table] DROP TABLE log\n");

127  // drop table

128  if (mysql_query(conn, "DROP TABLE IF EXISTS log;")) {

129  printf("DROP TABLE Query failed\n");

130  return ERR;

131  }

132

133  return print_result(conn);

134 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



5.4.3.4 get_one_row()

MySQL DB에서 log 데이터를 가져올 수 있습니다

매개변수

conn MySQL DB

반환값

RESULT

db.c 파일의 48 번째 라인에서 정의되었습니다.

```
48
      MYSQL_RES *res;
50
      MYSQL_ROW row;
51
      res = mysql_store_result(conn);
      if (!res) {
  fprintf(stderr, "%s\n", mysql_error(conn));
52
53
54
         return NULL;
55
      while ((row = mysql_fetch_row(res)) != NULL) {
   // printf("%s, %s, %s, %s, %s, %s, %s, %s, n", row[0], row[1], row[2],
   // row[3],
57
58
59
        // row[4
// string copy
60
                      row[4], row[5], row[6], row[7]);
62
         strcpy(tmp.name, row[1]);
63
         // convert string to int
         tmp.point = atoi(row[2]);
tmp.year = atoi(row[3]);
tmp.month = atoi(row[4]);
64
6.5
66
         tmp.day = atoi(row[5]);
tmp.hour = atoi(row[6]);
         tmp.min = atoi(row[7]);
69
70
71
72
      return &tmp;
```

5.4 src/db.c 파일 참조 25

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.5 init_db()

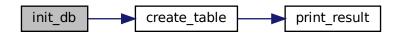
void init_db ()

MySQL DB를 초기화합니다

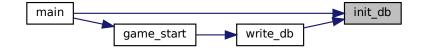
db.c 파일의 161 번째 라인에서 정의되었습니다.

```
161
162
      // establish connection with mysql
      conn = mysql_init(NULL);
// if connection failed
163
164
165
      if (!mysql_real_connect(conn, host, user, passwd, db, port, NULL, 0)) {
166
        fprintf(stderr, "%s\n", mysql_error(conn));
167
        mysql_close(conn);
168
       exit(1);
169
170
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.6 print_result()

```
int print_result ( {\tt MYSQL * conn} \ )
```

MySQL Result를 출력합니다

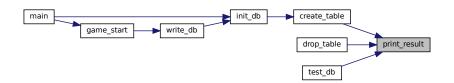
매개변수

```
conn MySQL DB
```

db.c 파일의 28 번째 라인에서 정의되었습니다.

```
MYSQL_RES *res;
MYSQL_ROW row;
29
30
     res = mysql_store_result(conn);
if (!res) {
31
32
        fprintf(stderr, "%s\n", mysql_error(conn));
35
     while ((row = mysql_fetch_row(res)) != NULL) {
  printf("%s\n", row[0]);
36
37
38
39
40
     return OK;
41 }
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.7 read_db()

void read_db ()

MySQL DB에서 데이터를 가져올 수 있습니다

매개변수

name 이름을 통해 사용자 검색

반환값

RESULT 구조체 주소 or NULL을 돌려줍니다

db.c 파일의 189 번째 라인에서 정의되었습니다.

5.4 src/db.c 파일 참조 27

```
189
190
      char query[256];
      printf("이름을 입력하세요: ");
scanf("%s", tmp.name);
191
192
      snprintf(query, sizeof(query), "SELECT * FROM log WHERE name = '%s'", tmp.name);
if (mysql_query(conn, query)) {
  printf("SELECT Query failed\n");
193
194
195
196
197
      RESULT *res = get_one_row(conn);
if (res == NULL) {
198
199
       system("clear");
200
       printf("*******************************
201
202
       printf("No History");
203
204
          system("clear");
printf("*******************************\n");
205
206
          207
209
210
211
     }
212 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.4.3.8 test_db()

```
int test_db ( \label{eq:mysql} \texttt{MYSQL} \, * \, \textit{conn} \, )
```

MySQL DB를 테스트합니다

매개변수

conn | MySQL DB

과일 문서화

db.c 파일의 79 번째 라인에서 정의되었습니다.

```
80
       printf("###########\n");
       printf("[test_db] SHOW databases\n");
81
82
      // show all databases
if (mysql_query(conn, "SHOW DATABASES;")) {
  printf("SHOW Query failed\n");
83
85
86
          return ERR;
87
88
      if (print_result(conn)) {
  printf("Print Result failed\n");
89
90
91
92
93
     printf("###############\n");
printf("[show_table] SHOW TABLES\n");
if (mysql_query(conn, "SHOW TABLES;")) {
94
95
       printf("SHOW TABLES Query failed\n");
98
         return ERR;
99
100
101
        return print_result(conn);
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



5.4.3.9 use_db()

MySQL DB를 사용합니다

매개변수

conn | MySQL DB

db.c 파일의 108 번째 라인에서 정의되었습니다.

```
108
109
       printf("###########\n");
110
      printf("[use_db] USE hm2j\n");
111
       // use database
       if (mysql_query(conn, "USE hm2j")) {
  printf("USE Query failed\n");
  return TDD:
112
113
      return ERR;
114
115
116
      return OK;
118 }
```

5.4 src/db.c 파일 참조 29

5.4.3.10 write_db()

```
void write_db (
         int point )
```

MySQL DB에 데이터를 저장할 수 있습니다

매개변수

point 점수, 아이디

db.c 파일의 218 번째 라인에서 정의되었습니다.

```
218
       printf("이름을 입력하세요: ");
219
220
       scanf("%s", tmp.name);
221
222
       new(point);
223
       tmp.point = point;
224
225
       time_t now_sec = time(NULL);
226
      struct tm *now = localtime(&now_sec);
228
       tmp.year = now->tm_year + 1900;
229
       tmp.month = now->tm_mon + 1;
       tmp.day = now->tm_mday;
tmp.hour = now->tm_hour;
230
2.31
       tmp.min = now->tm_min;
232
233
234
       if (conn == NULL) {
235
         init_db();
236
237
       char query[512];
238
239
       // safe version of printf
240
       // store query
241
       snprintf(query, sizeof(query),
                  "INSERT INTO log (name, point, year, month, day, hour, min) VALUES " "('%s', %d, %d, %d, %d, %d, %d)",
242
243
      tmp.name, tmp.point, tmp.year, tmp.month, tmp.day, tmp.hour, tmp.min);
if (mysql_query(conn, query)) {
  printf("INSERT Query failed: %s\n", mysql_error(conn));
244
245
246
247
248 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:

```
main game_start write_db
```

5.4.4 변수 문서화

5.4.4.1 conn

```
MYSQL* conn = NULL [static]
```

db.c 파일의 15 번째 라인에서 정의되었습니다.

5.4.4.2 db

```
const char* db = "hm2j" [static]
```

db.c 파일의 19 번째 라인에서 정의되었습니다.

5.4.4.3 host

```
const char* host = "localhost" [static]
```

db.c 파일의 16 번째 라인에서 정의되었습니다.

5.4.4.4 passwd

```
const char* passwd = "0000" [static]
```

db.c 파일의 18 번째 라인에서 정의되었습니다.

5.4.4.5 port

```
const int port = 3306 [static]
```

db.c 파일의 20 번째 라인에서 정의되었습니다.

5.4.4.6 tmp

```
RESULT tmp [static]
```

db.c 파일의 22 번째 라인에서 정의되었습니다.

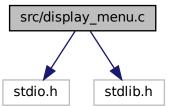
5.4.4.7 user

```
const char* user = "root" [static]
```

db.c 파일의 17 번째 라인에서 정의되었습니다.

5.5 src/display_menu.c 파일 참조

#include <stdio.h> #include <stdlib.h> display_menu.c에 대한 include 의존 그래프



함수

• int display_menu ()

5.5.1 함수 문서화

5.5.1.1 display_menu()

```
int display_menu ( )
```

display_menu.c 파일의 4 번째 라인에서 정의되었습니다.

```
8
     system("clear");
     10
                                                                      \\ /\n");
     printf("\t\t--
printf("\t\t|
                                    -----\n");
-|\n \t\t\t GAME MENU\n\t\t|____|\n");
-----\n");
13
     printf("\t\t----
14
15
                                            1. Game Start\n\t\t|_____|\n");
2. Search history\n\t\t|_____|\n");
3. Record Output\n\t\t|____|\n");
4. Quit\n\t\t|____|\n");
      printf("\t\t|
16
                                    |\n\t\t
      printf("\t\t|
printf("\t\t|
                                    |\n\t\t
18
                                    | n t 
19
     printf("\t\t|
                                    | n t 
20
      printf("\n");
21
22
      24
2.5
      printf("Select The Number: ");
      scanf("%d", &menu);
26
      return menu;
```

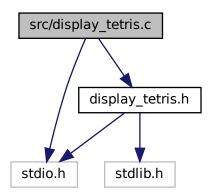
이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.6 src/display tetris.c 파일 참조

```
#include <stdio.h>
#include "display_tetris.h"
```

display_tetris.c에 대한 include 의존 그래프



함수

void display_tetris ()

변수

- int i_block [4][4][4]
- int t_block [4][4][4]
- int s_block [4][4][4]
- int z_block [4][4][4]
- int l_block [4][4][4]
- int j_block [4][4][4]
- int o_block [4][4][4]
- int next_block_number

5.6.1 함수 문서화

5.6.1.1 display_tetris()

```
void display_tetris ( )
```

display_tetris.c 파일의 13 번째 라인에서 정의되었습니다.

```
14 {
15    int i, j;
16    int(*block_pointer)[4][4][4] = NULL;
17    int block_color; // 인트 블럭 색상 선언
18
19    switch (next_block_number)
20    {
21    case 0:
22    block_pointer = &i_block;
```

```
23
              break;
24
25
              block_pointer = &t_block;
             break;
2.6
2.7
         case 2:
28
              block pointer = &s block;
29
              break;
30
31
            block_pointer = &z_block;
              break;
32
         case 4:
33
34
            block_pointer = &l_block;
35
              break;
36
         case 5:
37
             block_pointer = &j_block;
38
39
         case 6:
40
             block_pointer = &o_block;
41
              break;
         }
43
         // 터미널 화면을 지우고 다음 블록 출력
44
4.5
         system("clear");
         printf("Next Block\n");
46
         //블록 출력 시작
49
         for (i = 0; i < 4; i++)
50
51
               for (j = 0; j < 4; j++)
52
                    // 블록이 있는 경우 해당 색상으로 출력합니다.
53
                    if ((*block_pointer)[0][i][j] == 1)
55
                         printf(RED_COLOR " RESET_COLOR);
56
                    }else if ((*block_pointer)[0][i][j] == 2){
   printf(GREEN_COLOR "■ " RESET_COLOR);
57
58
                    }else if ((*block_pointer)[0][i][j] == 3){
   printf(YELLOW_COLOR " RESET_COLOR);
59
60
                    }else if ((*block_pointer)[0][i][j] == 4){
                         printf(BLUE_COLOR " RESET_COLOR);
                    }else if ((*block_pointer)[0][i][j] == 5){
   printf(MAGENTA_COLOR "■ " RESET_COLOR);
63
64
                    }else if ((*block_pointer)[0][i][j] == 6){
   printf(CYAN_COLOR "  RESET_COLOR);
6.5
66
                    }else if ((*block_pointer)[0][i][j] == 7);
printf(WHITE_COLOR " RESET_COLOR);
68
69
                    lelse {
                         printf("\Box ");
70
                    }
71
72
              printf("\n");
73
74
         // 블록이 없는 경우 공백을 출력합니다.
7.5
         printf("\n\n"); //두줄 공백 출력하는 것
printf("Tetris Table\n");
76
77
                                                              printf("\t+++
78
                              ( ≧ Д ≦ )
                                                     \n\t |
        줘!!\n\t |
         79
         for (i = 2; i < 21; i++)
80
               printf("\t");
81
               for (j = 0; j < 10; j++)
82
83
                    if (j == 0 || j == 9)
85
                         printf("= ");
86
87
                    else if (i == 20 && (j > 0 && j < 9))
88
89
90
                         printf("=");
91
                    else if (tetris_table[i][j] == 1){ //
    printf(RED_COLOR "■ " RESET_COLOR);
}else if (tetris_table[i][j] == 2){
    printf(GREEN_COLOR "■ " RESET_COLOR);
                                                                          //블럭포인터로 표현했었었는데, 테트리스테이블로 다시 함.
92
93
94
95
                    printf(GREEN_COLOR " " RESET_COLOR);
}else if (tetris_table[i][j] == 3) {
    printf(YELLOW_COLOR " RESET_COLOR);
}else if (tetris_table[i][j] == 4) {
    printf(BLUE_COLOR " RESET_COLOR);
}else if (tetris_table[i][j] == 5) {
    printf(MAGENTA_COLOR " RESET_COLOR);
}else if (tetris_table[i][i] == 6);
97
98
99
100
101
                     }else if (tetris_table[i][j] == 6) {
   printf(CYAN_COLOR " " RESET_COLOR);
102
103
                     104
105
106
107
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.6.2 변수 문서화

5.6.2.1 i_block

```
int i_block[4][4][4] [extern]
```

update.c 파일의 17 번째 라인에서 정의되었습니다.

5.6.2.2 j_block

```
int j_block[4][4][4] [extern]
```

update.c 파일의 122 번째 라인에서 정의되었습니다.

5.6.2.3 I_block

```
int l_block[4][4][4] [extern]
```

update.c 파일의 101 번째 라인에서 정의되었습니다.

5.6.2.4 next_block_number

```
int next_block_number [extern]
```

update.c 파일의 12 번째 라인에서 정의되었습니다.

5.6.2.5 o_block

```
int o_block[4][4][4] [extern]
```

update.c 파일의 143 번째 라인에서 정의되었습니다.

5.6.2.6 s_block

```
int s_block[4][4][4] [extern]
```

update.c 파일의 59 번째 라인에서 정의되었습니다.

5.6.2.7 t_block

```
int t_block[4][4][4] [extern]
```

update.c 파일의 38 번째 라인에서 정의되었습니다.

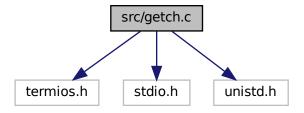
5.6.2.8 z_block

```
int z_block[4][4][4] [extern]
```

update.c 파일의 80 번째 라인에서 정의되었습니다.

5.7 src/getch.c 파일 참조

```
#include <termios.h>
#include <stdio.h>
#include <unistd.h>
getch.c에 대한 include 의존 그래프
```



매크로

- #define CCHAR 0
- #define CTIME 1
- #define FLAG TCSAFLUSH

함수

• int getch (void)

5.7.1 매크로 문서화

5.7.1.1 CCHAR

#define CCHAR 0

getch.c 파일의 6 번째 라인에서 정의되었습니다.

5.7.1.2 CTIME

#define CTIME 1

getch.c 파일의 10 번째 라인에서 정의되었습니다.

5.7.1.3 FLAG

#define FLAG TCSAFLUSH

5.7.2 함수 문서화

5.7.2.1 getch()

45 46 }

```
int getch (
               void )
getch.c 파일의 13 번째 라인에서 정의되었습니다.
14 {
15
       char ch;
       int error;
16
17
       static struct termios Otty, Ntty;
18
       // fflush(stdout);
19
20
       tcgetattr(0, &Otty);
21
       Ntty = Otty;
       Ntty.c_iflag = 0;
       Ntty.c_oflag = 0;
23
       Ntty.c_lflag &= ~ICANON;
25 #if 1
2.6
       Ntty.c_lflag &= ~ECHO;
27 #else
      Ntty.c_lflag |= ECHO;
29 #endif
       Ntty.c_cc[VMIN] = CCHAR;
Ntty.c_cc[VTIME] = CTIME;
30
31
32
33 #if 1
34 #define FLAG TCSAFLUSH
35 #else
36 #define FLAG TCSANOW
37 #endif
38
39
       if (0 == (error = tcsetattr(0, FLAG, &Ntty)))
40
           error = read(0, &ch, 1);
error += tcsetattr(0, FLAG, &Otty);
42
43
44
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:

return (error == 1 ? (int)ch : -1);



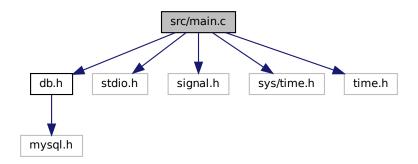
5.8 src/main.c 파일 참조

테트리스 게임을 실행하는 메인 함수

```
#include "db.h"
#include <stdio.h>
#include <signal.h>
#include <sys/time.h>
```

5.8 src/main.c 파일 참조 39

#include <time.h>
main.c에 대한 include 의존 그래프



매크로

- #define GAME START 0
- #define GAME_END 1

함수

- int display_menu ()
- void display_tetris ()
- int update (int signum)
- void save_result (int)
- int print_result (void)
- int print_color ()
- int game_start ()

이 함수는 signal을 설정하는 함수입니다.

- int main ()
- int init_tetris_table ()

변수

- int game
- int x = 3
- int y = 0
- int point = 0
- int tetris_table [21][10]
- char q

5.8.1 상세한 설명

테트리스 게임을 실행하는 메인 함수

작성자

최수길

날짜

2024-3-20

이 게임은 signal을 사용해서 1/60 초 마다 화면을 갱신 하는 것을 이용한 프로그램입니다.

5.8.2 매크로 문서화

5.8.2.1 **GAME_END**

#define GAME_END 1

main.c 파일의 17 번째 라인에서 정의되었습니다.

5.8.2.2 GAME_START

#define GAME_START 0

main.c 파일의 16 번째 라인에서 정의되었습니다.

5.8.3 함수 문서화

5.8 src/main.c 파일 참조 41

5.8.3.1 display_menu()

```
int display_menu ( )
```

display_menu.c 파일의 4 번째 라인에서 정의되었습니다.

```
int menu;
8
     system("clear");
                    printf("\t\t\t
printf("\t\t|
9
10
                                                                         \\ /\n");
      GAME\n\t\t|___
11
12
                                     | \n \t\t\ GAME MENU\n\t\t|_____|\n");
14
1.5
                                             1. Game Start\n\t\t|______|\n");
2. Search history\n\t\t|_____|\n");
3. Record Output\n\t\t|_____|\n");
4. Quit\n\t\t|_____|\n");
                                     |\n\t\t
|\n\t\t
      printf("\t\t|
16
      printf("\t\t|
17
      printf("\t\t|
                                     |\n\t\t
18
19
     printf("\t\t|
                                     | n t 
20
21
      printf("\n");
22
                    2.3
25
      printf("Select The Number: ");
26
      scanf("%d", &menu);
27
      return menu;
28 }
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.8.3.2 display_tetris()

```
void display_tetris ( )
```

display_tetris.c 파일의 13 번째 라인에서 정의되었습니다.

```
14
        int i, j;
int(*block_pointer)[4][4][4] = NULL;
int block_color; // 인트 블럭 색상 선언
1.5
16
17
18
        switch (next_block_number)
20
        case 0:
21
        block_pointer = &i_block;
22
23
            break;
       case 1:
        block_pointer = &t_block;
25
            break;
27
       case 2:
       block_pointer = &s_block;
break;
2.8
29
30
          block_pointer = &z_block;
```

```
32
            break;
        case 4:
33
34
            block_pointer = &1_block;
            break;
35
36
        case 5:
37
            block_pointer = &j_block;
38
            break;
39
        case 6:
           block_pointer = &o_block;
40
41
42
43
        // 터미널 화면을 지우고 다음 블록 출력
44
45
        system("clear");
46
        printf("Next Block\n");
47
        //블록 출력 시작
48
        for (i = 0; i < 4; i++)
49
50
             for (j = 0; j < 4; j++)
52
                  // 블록이 있는 경우 해당 색상으로 출력합니다.
53
54
                  if ((*block_pointer)[0][i][j] == 1)
5.5
                      printf(RED_COLOR " RESET_COLOR);
56
                 }else if ((*block_pointer)[0][i][j] == 2){
    printf(GREEN_COLOR " RESET_COLOR);
58
                  }else if ((*block_pointer)[0][i][j] == 3){
   printf(YELLOW_COLOR "■ " RESET_COLOR);
59
60
                 }else if ((*block_pointer)[0][i][j] == 4){
    printf(BLUE_COLOR "■ " RESET_COLOR);
61
62
                 }else if ((*block_pointer)[0][i][j] == 5) {
    printf(MAGENTA_COLOR " RESET_COLOR);
63
65
                  }else if ((*block_pointer)[0][i][j] == 6){
                  printf(CYAN_COLOR " " RESET_COLOR);
}else if ((*block_pointer)[0][i][j] == 7){
    printf(WHITE_COLOR " " RESET_COLOR);
66
67
68
69
70
                      printf("□ ");
71
72
             printf("\n");
7.3
74
        // 블록이 없는 경우 공백을 출력합니다.
75
        printf("\n\n"); //두줄 공백 출력하는 것
76
77
        printf("Tetris Table\n");
                                                        78
        printf("\t++
                                                                                                                              살려
                            ( ≧ Д ≦ )
                                               \n\t |
\n");
       줘!!\n\t|
기\n\t
79
        for (i = 2; i < 21; i++)
80
81
             printf("\t");
82
             for (j = 0; j < 10; j++)
83
                  if (j == 0 || j == 9)
84
85
                      printf("= ");
88
                  else if (i == 20 && (j > 0 && j < 9))
89
                      printf("=");
90
91
                  else if (tetris_table[i][j] == 1){
   printf(RED_COLOR "■ " RESET_COLOR);
                                                                 //블럭포인터로 표현했었었는데, 테트리스테이블로 다시 함.
                  }else if (tetris_table[i][j] == 2){
94
                      printf(GREEN_COLOR "■ " RESET_COLOR);
95
                  }else if (tetris_table[i][j] == 3){
   printf(YELLOW_COLOR "■ " RESET_COLOR);
96
                  }else if (tetris_table[i][j] == 4) {
   printf(BLUE_COLOR " RESET_COLOR);
98
                   }else if (tetris_table[i][j] == 5) {
   printf(MAGENTA_COLOR " " RESET_COLOR);
100
101
                   }else if (tetris_table[i][j] == 6) {
   printf(CYAN_COLOR " " RESET_COLOR);
102
103
                   104
105
106
                      printf(" ");
107
108
109
              printf("\n");
110
111
         printf("\t 신해철 - 그대에게\n\t0:35 —
112
                                                                                                  \n");
113 }
```

5.8 src/main.c 파일 참조 43

5.8.3.3 game_start()

```
int game_start ( )
```

이 함수는 signal을 설정하는 함수입니다.

매개변수

signum	signal 에 연결된 함수에 전달되는 인자
timer	signal을 설정하는 구조체 여기에 갱신 시간을 정한다.

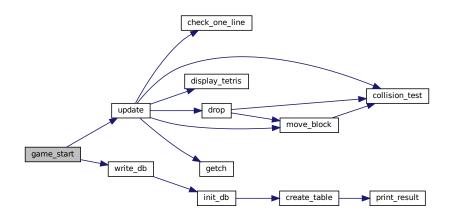
이 함수에서는 x, y, point 값을 초기화 시킨다.

main.c 파일의 38 번째 라인에서 정의되었습니다.

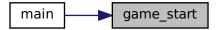
```
39 {
         // signal 설정
static struct itimerval timer;
signal(SIGVTALRM, update);
40
41
42
         timer.it_value.tv_sec = 0;
44
         timer.it_value.tv_usec = 16667;
         timer.it_interval.tv_sec = 0;
timer.it_interval.tv_usec = 16667;
45
46
         setitimer(ITIMER_VIRTUAL, &timer, NULL);
47
48
         while (1)
49
50
              if (game == GAME_END)
51
52
                    signal(SIGVTALRM, SIG_IGN);
                   // save_result (point);
write_db (point);
53
54
                    x = 3;

y = 0;
                    point = 0;
57
58
                    return 1;
59
60
61
         game = GAME_END;
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.8.3.4 init_tetris_table()

```
int init_tetris_table ( )
```

main.c 파일의 92 번째 라인에서 정의되었습니다.

```
93 {
        int i, j;
// 모든 데이터에 0을 넣는다.
for (i = 0; i < 21; i++)
94
95
96
98
             for (j = 0; j < 10; j++)
99
100
                  tetris_table[i][j] = 0;
101
102
103
         // 맨 아래칸에 1 추가 (바닥)
104
         for (j = 1; j < 9; j++)
105
106
              tetris_table[20][j] = 1;
107
         // 양 옆에 1 추가 (벽)
for (i = 0; i < 21; i++)
108
109
110
              tetris_table[i][0] = 1;
111
              tetris_table[i][9] = 1;
112
113
114 }
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.8 src/main.c 파일 참조 45

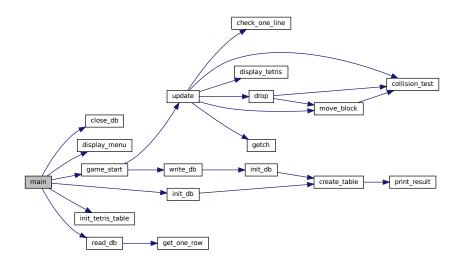
5.8.3.5 main()

```
int main ( )
```

main.c 파일의 64 번째 라인에서 정의되었습니다.

```
int menu = 1;
       menu = display_menu();
67
68
       switch (menu)
69
70
       case 1:
71
           game = GAME_START;
72
           init_tetris_table();
73
           menu = game_start();
74
          break;
75
       case 2:
          printf("Search history\n");
init_db();
76
77
78
           read_db();
79
           close_db();
80
           break;
       case 3:
81
          printf("Record Output\n");
82
           // print_result();
break;
83
85
       case 4:
86
          printf("Quit\n");
87
           break;
88
89
       return 0;
90 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



5.8.3.6 print_color()

```
int print_color ( )
```

5.8.3.7 print_result()

```
int print_result (
     void )
```

5.8.3.8 save result()

```
void save_result (
    int )
```

5.8.3.9 update()

```
int update ( \quad \text{ int } signum \ )
```

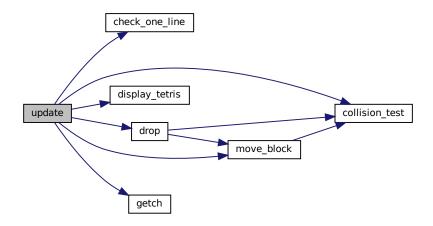
update.c 파일의 174 번째 라인에서 정의되었습니다.

```
175 {
176
         static int downcount = 0;
177
         static int setcount = 0;
         static long speedcount = 0;
178
179
         static int countrange = 5;
static int firststart = 0;
180
181
182
183
184
         srand((unsigned)time(NULL));
185
         if (firststart == 0)
186
187
              block_number = rand() % 7;
if (firststart == 0)
188
189
190
                   firststart++;
191
192
193
         display_tetris();
194
         check_one_line();
195
196
         if (downcount == countrange - 1)
197
198
              point += 1;
              move_block(DOWN);
199
200
201
202
         if (speedcount == 499)
203
              if (countrange != 1)
2.04
205
                  countrange--;
206
207
208
         downcount++;
209
         downcount %= countrange;
         speedcount++;
speedcount %= 500;
210
211
212
213
         if (x == 3 && y == 0)
214
215
              if (collision_test(LEFT) || collision_test(RIGHT) || collision_test(DOWN) ||
        collision_test(ROTATE))
216
217
                   printf("\n Game End! \n");
                  downcount = 0;
setcount = 0;
218
219
                  speedcount = 0;
countrange = 5;
firststart = 0;
220
221
222
223
                   game = GAME_END;
224
        }
```

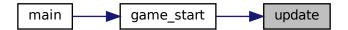
5.8 src/main.c 파일 참조 47

```
226
227
         if (collision_test(DOWN))
228
229
              if (setcount == 9)
230
                   block_number = next_block_number;
next_block_number = rand() % 7;
231
232
233
                   block_state = 0;
234
                   x = 3;
                   y = 0;
235
236
237
              setcount++;
238
              setcount %= 10;
239
240
241
         ch = getch();
242
243
         switch (ch)
244
245
         case 74:
246
         case 106:
247
          move_block(LEFT);
248
             break;
         case 76:
case 108:
249
250
251
           move_block(RIGHT);
break;
252
253
         case 75:
         case 107:
254
          move_block(DOWN);
break;
255
256
257
         case 73:
258
         case 105:
259
             move_block(ROTATE);
260
             break;
         case 65:
case 97:
261
262
             drop();
263
264
             break;
265
         case 80:
266
         case 112:
             downcount = 0;
setcount = 0;
speedcount = 0;
2.67
2.68
269
             countrange = 5;
firststart = 0;
270
271
272
              game = GAME_END;
273
             break;
274
         default:
275
             break:
277
         return 0;
278 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.8.4 변수 문서화

5.8.4.1 game

int game

main.c 파일의 21 번째 라인에서 정의되었습니다.

5.8.4.2 point

int point = 0

main.c 파일의 25 번째 라인에서 정의되었습니다.

5.8.4.3 q

char q

main.c 파일의 30 번째 라인에서 정의되었습니다.

5.8.4.4 tetris_table

int tetris_table[21][10] [extern]

display_tetris.h 파일의 34 번째 라인에서 정의되었습니다.

5.9 src/new.c 파일 참조 49

5.8.4.5 x

int x = 3

main.c 파일의 23 번째 라인에서 정의되었습니다.

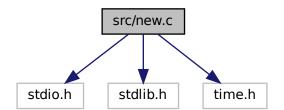
5.8.4.6 y

int y = 0

main.c 파일의 24 번째 라인에서 정의되었습니다.

5.9 src/new.c 파일 참조

#include <stdio.h> #include <stdlib.h> #include <time.h> new.c에 대한 include 의존 그래프



함수

- int getRandomScore ()
- int chooseDoll (int score)
- int new (int a)

5.9.1 함수 문서화

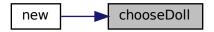
5.9.1.1 chooseDoll()

87 }

```
int chooseDoll (
               int score )
new.c 파일의 25 번째 라인에서 정의되었습니다.
25
        if (score >= 200000) {
26
            printf(". \land \_ \land \land n");
28
            printf(" ( ¬¬¬\n");
printf("ε (_)^^\n");
printf("( ( ·o·)\n");
printf("⊙--⊙ ⊃ \n");
return 10; // 200,000 이상일 경우 10번 인형 획득
29
30
31
32
34
        } else if (score >= 100000) {
          35
36
37
38
39
            return 9; // 100000 이상일 경우 9번 인형 획득
       } else if (score >= 50000) {
   printf(". ^,^\n");
   printf(" \( \cap (^ 0 ^ ) \cap \cap (^n"); \)
}
41
42
43
           44
45
46
47
            return 8; // 50000 이상일 경우 8번 인형 획득
48
       } else if (score >= 30000) {
49
          50
51
53
55
        } else if (score >= 20000) {
56
            printf("( () )\n");
printf("/ ^\n");
printf("人__つつ\n");
return 6; // 2000이 이상일 경우 6번 인형 획득
58
60
61
        | else if (score >= 15000) {
| printf(". ^^ ■");
| printf(" (^w^)/ ");
| printf(" < /");
| printf(" UU");
62
63
65
66
67
            return 5; // 15000 이상일 경우 5번 인형 획득
68
       } else if (score >= 10000) {
69
70
            return 4; // 10000 이상일 경우 4번 인형 획득
72
73
        } else if (score >= 5000) {
74
75
        printf(" (\(__u_u)\n");
return 3; // 5000 이상일 경우 3번 인형 획득
} else if (score >= 3000) {
76
78
            printf(". /) /)\n");
79
80
        81
82
84
            return 1; // 1000 이상일 경우 1번 인형 획득
85
86
        }
```

5.9 src/new.c 파일 참조 51

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.9.1.2 getRandomScore()

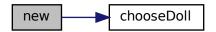
```
int getRandomScore ( )
```

5.9.1.3 new()

```
int new ( \quad \text{int } a \ )
```

new.c 파일의 9 번째 라인에서 정의되었습니다.

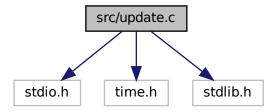
이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



5.10 src/result.c 파일 참조

5.11 src/update.c 파일 참조

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
update.c에 대한 include 의존 그래프
```



매크로

- #define GAME_START 0
- #define GAME_END 1

열거형 타입

• enum COMMAND { DOWN , LEFT , RIGHT , ROTATE }

함수

- char getch ()
- int display_tetris ()
- int update (int signum)
- int move_block (int command)
- int collision_test (int command)
- int drop (void)
- int check_one_line (void)

변수

- int x
- int y
- int block_state = 0
- int block [4][4][4]
- int point
- int tetris_table [21][10]
- int block_number
- int next_block_number
- int game
- int i_block [4][4][4]
- int t_block [4][4][4]
- int s_block [4][4][4]
- int z_block [4][4][4]
- int l_block [4][4][4]
- int j_block [4][4][4]
- int o_block [4][4][4]

5.11.1 매크로 문서화

5.11.1.1 GAME_END

#define GAME_END 1

update.c 파일의 165 번째 라인에서 정의되었습니다.

5.11.1.2 **GAME_START**

#define GAME_START 0

update.c 파일의 164 번째 라인에서 정의되었습니다.

5.11.2 열거형 타입 문서화

5.11.2.1 COMMAND

enum COMMAND

열거형 멤버

DOWN	
LEFT	
RIGHT	
ROTATE	

update.c 파일의 167 번째 라인에서 정의되었습니다.

```
168 {
169 DOWN,
170 LEFT,
171 RIGHT,
172 ROTATE
173 } COMMAND;
```

5.11.3 함수 문서화

5.11.3.1 check_one_line()

```
int check_one_line (
     void )
```

update.c 파일의 468 번째 라인에서 정의되었습니다.

```
469 {
        int i, j;
int ti, tj;
int line_hole;
470
471
472
473
474
        for (i = 19; i > 0; i--)
475
             line_hole = 0;
476
             for (j = 1; j < 9; j++)</pre>
477
478
                  if (tetris_table[i][j] == 0)
480
481
                      line_hole = 1;
482
483
             }
484
485
             if (line_hole == 0)
487
                  point += 1000;
                  for (ti = i; ti > 0; ti--)
488
489
490
                      for (tj = 0; tj < 9; tj++)
491
492
                           tetris_table[ti][tj] = tetris_table[ti - 1][tj];
493
494
495
496
        }
497
498
        return 0;
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.11.3.2 collision_test()

```
int collision_test (
              int command )
update.c 파일의 370 번째 라인에서 정의되었습니다.
371 {
372
        int i, j;
373
        int tempx, tempy;
374
        int oldx, oldy;
375
        int temp_block_state;
376
        int(*block_pointer)[4][4][4];
377
        int temp_tetris_table[21][10];
378
        oldx = tempx = x;
oldy = tempy = y;
379
380
381
        temp_block_state = block_state;
382
383
        switch (command)
384
385
        case LEFT:
386
           tempx--;
387
           break;
388
        case RIGHT:
389
           tempx++;
390
           break:
391
        case DOWN:
392
          tempy++;
393
           break;
394
        case ROTATE:
395
           temp_block_state++;
396
            temp_block_state %= 4;
397
            break;
398
399
400
        switch (block_number)
401
        case 0:
402
403
           block_pointer = &i_block;
404
           break;
405
406
          block_pointer = &t_block;
407
           break;
        case 2:
408
409
           block_pointer = &s_block;
410
           break;
411
412
           block_pointer = &z_block;
413
           break;
414
        case 4:
415
           block pointer = &1 block;
416
           break;
417
        case 5:
418
           block_pointer = &j_block;
419
           break;
420
        case 6:
421
           block_pointer = &o_block;
422
            break;
423
        }
424
        for (i = 0; i < 21; i++)</pre>
425
426
            for (j = 0; j < 10; j++)
427
428
429
                temp_tetris_table[i][j] = tetris_table[i][j];
430
431
432
        for (i = 0, oldy = y; i < 4; i++, oldy++)
433
434
435
            for (j = 0, oldx = x; j < 4; j++, oldx++)
436
437
                if (oldx > 0 && oldx < 9 && oldy < 20 && oldy > 0)
438
                    if ((*block_pointer)[block_state][i][j] != 0)
439
440
                         temp_tetris_table[oldy][oldx] = 0;
441
442
443
        }
444
        for (i = 0; i < 4; i++)
445
446
447
            for (j = 0; j < 4; j++)
448
```

```
449
450

if (temp_tetris_table[tempy + i][tempx + j] != 0 && (*block_pointer)[temp_block_state][i][j]

!= 0)

451

return 1;

452

}

453

}

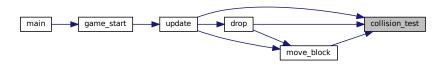
454

455

return 0;

456
}
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.11.3.3 display tetris()

```
int display_tetris ( )
```

display_tetris.c 파일의 13 번째 라인에서 정의되었습니다.

```
14 {
15
        int (*block_pointer)[4][4][4] = NULL;
int block_color; // 인트 블럭 색상 선언
16
17
18
19
        switch (next_block_number)
20
        case 0:
21
22
            block_pointer = &i_block;
2.3
            break;
2.4
        case 1:
25
            block_pointer = &t_block;
26
            break;
        case 2:
28
            block_pointer = &s_block;
29
30
        case 3:
           block_pointer = &z_block;
31
32
           break;
33
        case 4:
34
          block_pointer = &l_block;
35
36
        case 5:
37
           block_pointer = &j_block;
38
            break:
39
        case 6:
40
           block_pointer = &o_block;
41
42
43
        // 터미널 화면을 지우고 다음 블록 출력
44
        system("clear");
45
        printf("Next Block\n");
46
47
48
        //블록 출력 시작
       for (i = 0; i < 4; i++)
49
50
            for (j = 0; j < 4; j++)
51
                 // 블록이 있는 경우 해당 색상으로 출력합니다.
                 if ((*block_pointer)[0][i][j] == 1)
55
                 printf(RED_COLOR " " RESET_COLOR);
}else if ((*block_pointer)[0][i][j] == 2) {
    printf(GREEN_COLOR " " RESET_COLOR);
56
58
                 }else if ((*block_pointer)[0][i][j] == 3){
```

```
printf(YELLOW_COLOR "■ " RESET_COLOR);
60
                     PITHELITEDW_COLOR " RESET_COLOR);
}else if ((*block_pointer)[0][i][j] == 4) {
    printf(BLUE_COLOR " RESET_COLOR);
}else if ((*block_pointer)[0][i][j] == 5) {
    printf(MAGENTA_COLOR " RESET_COLOR);
}
62
63
64
                     }else if ((*block_pointer)[0][i][j] == 6){
65
                           printf(CYAN_COLOR "■ " RESET_COLOR);
66
                      }else if ((*block_pointer)[0][i][j] == 7){
68
                           printf(WHITE_COLOR " " RESET_COLOR);
69
                      }else {
                           printf("□ ");
70
71
72
73
               printf("\n");
74
         // 블록이 없는 경우 공백을 출력합니다.
printf("\n\n"); //두줄 공백 출력하는 것
printf("Tetris Table\n");
75
76
                                                                  printf("\t-
        줘!!\n\t |
- - \n\t - - -
                                  ( ≧ Д ≦ )
         for (i = 2; i < 21; i++)
79
80
               printf("\t");
for (j = 0; j < 10; j++)</pre>
81
82
83
84
                     if (j == 0 || j == 9)
85
86
                           printf("= ");
87
                     else if (i == 20 && (j > 0 && j < 9))
88
89
90
                           printf("=");
91
                     printf(RED_COLOR "■ " RESET_COLOR);

}else if (tetris_table[i][j] == 1){
    printf(GREEN_COLOR "■ " RESET_COLOR);

}else if (tetris_table[i][j] == 2){
    printf(GREEN_COLOR "■ " RESET_COLOR);
                                                                               //블럭포인터로 표현했었었는데, 테트리스테이블로 다시 함.
92
93
94
95
                     }else if (tetris_table[i][j] == 3){
   printf(YELLOW_COLOR " RESET_COLOR);
97
                     }else if (tetris_table[i][j] == 4) {
   printf(BLUE_COLOR " " RESET_COLOR);
98
99
                      100
101
103
                      PINITICIAN_COLOR "■ " RESET_COLOR);
}else if (tetris_table[i][j] == 7){
  printf(WHITE_COLOR "■ " RESET_COLOR);
}else { // 블록이 없는 경우 공백
  printf(" ");
104
105
106
107
108
109
110
                 printf("\n");
111
           printf("\t 신해철 - 그대에게\n\t0:35 — -3:48\n\t
                                                                                                       II
                                                                                                                  \n");
112
113 }
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



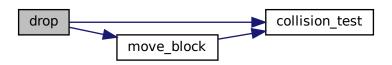
5.11.3.4 drop()

```
int drop (
     void )
```

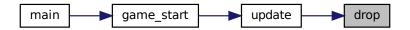
update.c 파일의 459 번째 라인에서 정의되었습니다.

```
460 {
461 while (!collision_test(DOWN))
462 move_block(DOWN);
463
464 return 0;
465 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.11.3.5 getch()

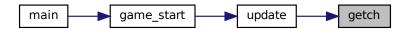
char getch ()

getch.c 파일의 13 번째 라인에서 정의되었습니다.

```
14 {
15
        char ch;
16
        int error;
        static struct termios Otty, Ntty;
17
18
19
        // fflush(stdout);
20
        tcgetattr(0, &Otty);
        Ntty = Otty;
Ntty.c_iflag = 0;
21
22
        Ntty.c_oflag = 0;
Ntty.c_lflag &= ~ICANON;
23
24
25 #if 1
        Ntty.c_lflag &= ~ECHO;
27 #else
28
       Ntty.c_lflag |= ECHO;
29 #endif
       Ntty.c_cc[VMIN] = CCHAR;
Ntty.c_cc[VTIME] = CTIME;
30
32
33 #if 1
34 #define FLAG TCSAFLUSH
35 #else
36 #define FLAG TCSANOW
37 #endif
38
```

```
39     if (0 == (error = tcsetattr(0, FLAG, &Ntty)))
40     {
41         error = read(0, &ch, 1);
42         error += tcsetattr(0, FLAG, &Otty);
43     }
44     return (error == 1 ? (int)ch : -1);
46 }
```

이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.11.3.6 move_block()

update.c 파일의 281 번째 라인에서 정의되었습니다.

```
283
         int i, j;
284
        int newx, newy;
        int oldx, oldy;
int old_block_state;
int(*block_pointer)[4][4][4] = NULL;
285
286
287
288
289
290
        newy = y;
291
292
        old_block_state = block_state;
293
294
        if (collision_test(command) == 0)
295
296
             switch (command)
297
             case LEFT:
298
299
                 newx--;
300
                 break;
301
             case RIGHT:
302
                newx++;
303
304
             case DOWN:
305
               newy++;
break;
306
307
             case ROTATE:
308
                block_state++;
309
                 block_state %= 4;
310
                 break;
311
312
313
        else
314
315
             return 1;
316
317
318
        switch (block_number)
319
320
321
            block_pointer = &i_block;
322
             break;
323
         case 1:
324
            block_pointer = &t_block;
325
            break;
        case 2:
```

```
block_pointer = &s_block;
328
             break;
329
         case 3:
330
             block_pointer = &z_block;
331
             break;
332
        case 4:
333
             block_pointer = &l_block;
334
335
         case 5:
336
             block_pointer = &j_block;
337
            break;
338
        case 6:
339
             block_pointer = &o_block;
340
341
342
             for (i = 0, oldy = y; i < 4; i++, oldy++)
343
344
345
             for (j = 0, oldx = x; j < 4; j++, oldx++)
347
                  if (oldx > 0 && oldx < 9 && oldy < 20 && oldy > 0)
                      if ((*block_pointer)[old_block_state][i][j] != 0)
348
349
                          tetris_table[oldy][oldx] = 0;
350
351
        }
352
353
354
355
356
        for (i = 0, newy = y; i < 4; i++, newy++)
357
358
             for (j = 0, newx = x; j < 4; j++, newx++)
359
360
                  if (newx > 0 \&\& newx < 9 \&\& newy < 20 \&\& newy > 0)
                      if ((*block_pointer)[block_state][i][j] != 0)
    tetris_table[newy][newx] = (*block_pointer)[block_state][i][j];
361
362
363
364
365
366
        return 0;
367 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:

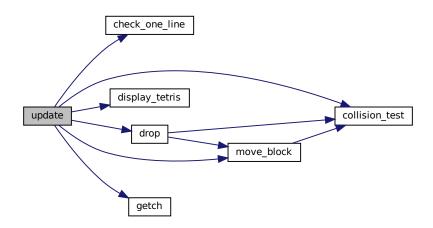


5.11.3.7 update()

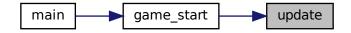
```
int update (
                int signum )
update.c 파일의 174 번째 라인에서 정의되었습니다.
175 {
176
        static int downcount = 0;
177
        static int setcount = 0;
178
        static long speedcount = 0;
        static int countrange = 5;
static int firststart = 0;
179
180
181
182
        char ch;
183
184
        srand((unsigned)time(NULL));
185
186
        if (firststart == 0)
187
             block_number = rand() % 7;
if (firststart == 0)
188
189
190
                 firststart++;
191
192
193
        display_tetris();
194
        check_one_line();
195
196
         if (downcount == countrange - 1)
197
             point += 1;
198
             move_block(DOWN);
199
200
        }
201
202
         if (speedcount == 499)
203
204
             if (countrange != 1)
205
                 countrange--;
206
207
208
        downcount++;
209
        downcount %= countrange;
210
        speedcount++;
211
        speedcount %= 500;
212
213
        if (x == 3 && y == 0)
214
        {
215
             if (collision_test(LEFT) || collision_test(RIGHT) || collision_test(DOWN) ||
       collision_test(ROTATE))
216
                 printf("\n Game End! \n");
downcount = 0;
setcount = 0;
217
218
219
220
                 speedcount = 0;
                 countrange = 5;
firststart = 0;
221
222
                 game = GAME_END;
223
224
225
        }
226
227
         if (collision_test(DOWN))
228
229
             if (setcount == 9)
230
231
                 block_number = next_block_number;
232
                 next_block_number = rand() % 7;
233
                 block_state = 0;
234
                 x = 3;
                 y = 0;
235
236
237
             setcount++;
             setcount %= 10;
238
239
240
241
        ch = getch();
242
243
        switch (ch)
244
245
        case 74:
246
        case 106:
          move_block(LEFT);
247
248
            break;
249
        case 76:
250
        case 108:
            move_block(RIGHT);
```

```
break;
         case 75:
253
254
              move_block(DOWN);
255
256
257
         break;
case 73:
case 105:
258
259
           move_block(ROTATE);
260
               break;
         case 65:
case 97:
261
262
             drop();
263
264
              break;
265
          case 80:
266
          case 112:
           downcount = 0;
setcount = 0;
speedcount = 0;
countrange = 5;
firststart = 0;
267
268
269
270
271
272
               game = GAME_END;
273
               break;
274
          default:
275
              break;
276
          return 0;
278 }
```

이 함수 내부에서 호출하는 함수들에 대한 그래프입니다.:



이 함수를 호출하는 함수들에 대한 그래프입니다.:



5.11.4 변수 문서화

5.11.4.1 block

```
int block[4][4][4] [extern]
```

display tetris.h 파일의 16 번째 라인에서 정의되었습니다.

5.11.4.2 block_number

```
int block_number
```

update.c 파일의 11 번째 라인에서 정의되었습니다.

5.11.4.3 block_state

```
int block_state = 0
```

update.c 파일의 7 번째 라인에서 정의되었습니다.

5.11.4.4 game

```
int game [extern]
```

main.c 파일의 21 번째 라인에서 정의되었습니다.

5.11.4.5 i_block

update.c 파일의 17 번째 라인에서 정의되었습니다.

5.11.4.6 j_block

update.c 파일의 122 번째 라인에서 정의되었습니다.

5.11.4.7 I_block

update.c 파일의 101 번째 라인에서 정의되었습니다.

5.11.4.8 next_block_number

int next_block_number

update.c 파일의 12 번째 라인에서 정의되었습니다.

5.11.4.9 o_block

update.c 파일의 143 번째 라인에서 정의되었습니다.

5.11.4.10 point

```
int point [extern]
```

main.c 파일의 25 번째 라인에서 정의되었습니다.

5.11.4.11 s_block

update.c 파일의 59 번째 라인에서 정의되었습니다.

5.11.4.12 t_block

```
int t_block[4][4][4]

\(\frac{2}{2}\), \(0, 0\), \(\{2}\), \(2, 0\), \(0, 0\), \(\{2}\), \(2, 0\), \(0, 0\), \(\{0}\), \(2, 2\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(2\), \(2\), \(2\), \(0, 0\), \(\{0}\), \(2\), \(0, 0\), \(\{0}\), \(2\), \(0, 0\), \(\{0}\), \(2\), \(0, 0\), \(\{0}\), \(2\), \(0, 0\), \(\{0}\), \(0, 0\), \(\{0}\), \(0, 0\), \(0\), \(0, 0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\), \(0\)
```

update.c 파일의 38 번째 라인에서 정의되었습니다.

5.11.4.13 tetris_table

```
int tetris_table[21][10] [extern]
```

display_tetris.h 파일의 34 번째 라인에서 정의되었습니다.

5.11.4.14 x

```
int x [extern]
```

main.c 파일의 23 번째 라인에서 정의되었습니다.

5.11.4.15 y

```
int y [extern]
```

main.c 파일의 24 번째 라인에서 정의되었습니다.

5.11.4.16 z_block

update.c 파일의 80 번째 라인에서 정의되었습니다.

Index

block	print_result, 25
display_tetris.h, 19	read_db, <mark>26</mark>
update.c, 62	test_db, 27
block_number	tmp, 30
update.c, 63	use_db, <mark>28</mark>
block_state	user, 31
display_tetris.h, 19	write_db, 28
update.c, 63	db.h
BLUE COLOR	close_db, 12
display_tetris.h, 17	drop table, 12
, ,	init db, 13
CCHAR	read_db, 14
getch.c, 37	write_db, 15
check_one_line	display_menu
update.c, 54	display_menu.c, 31
chooseDoll	main.c, 40
new.c, 49	display_menu.c
close db	display_menu, 31
db.c, 22	display_tetris
db.h, 12	display_tetris.c, 33
collision_test	main.c, 41
update.c, 54	
COMMAND	update.c, 56 display tetris.c
update.c, 53	
conn	display_tetris, 33
db.c, 30	i_block, 35
	j_block, 35
create_table	l_block, 35
db.c, 22 CTIME	next_block_number, 35
	o_block, 35
getch.c, 37	s_block, 36
CYAN_COLOR	t_block, 36
display_tetris.h, 17	z_block, 36
dou	display_tetris.h
day RESULT, 10	block, 19
db	block_state, 19
	BLUE_COLOR, 17
db.c, 30	CYAN_COLOR, 17
db.c	GREEN_COLOR, 17
close_db, 22	MAGENTA_COLOR, 18
conn, 30	RED_COLOR, 18
create_table, 22	RESET_COLOR, 18
db, 30	tetris_table, 19
drop_table, 23	WHITE_COLOR, 18
ERR, 21	YELLOW_COLOR, 18
get_one_row, 24	DOWN
host, 30	update.c, 54
init_db, 25	drop
OK, 21	update.c, 57
passwd, 30	drop_table
port, 30	• –

70 INDEX

db.c, 23	MAGENTA_COLOR
db.h, 12	display_tetris.h, 18
ERR	main
db.c, 21	main.c, 44
40.0, 21	main.c
FLAG	display_menu, 40 display_tetris, 41
getch.c, 37	game, 48
	GAME_END, 40
game	GAME_START, 40
main.c, 48	game_start, 42
update.c, 63	init_tetris_table, 44
GAME_END	main, 44
main.c, 40	point, 48
update.c, 53	print_color, 45
GAME_START	print_result, 45
main.c, 40	q, 48
update.c, 53	save_result, 46
game_start	tetris_table, 48
main.c, 42	update, 46
get_one_row	x, 48
db.c, 24	y, 49
getch getch.c, 37	min
update.c, 58	RESULT, 10
getch.c	month
CCHAR, 37	RESULT, 10
CTIME, 37	move_block
FLAG, 37	update.c, 59
getch, 37	
getRandomScore	name
new.c, 51	RESULT, 10
GREEN_COLOR	new
display tetris.h, 17	new.c, 51
,	new.c
host	chooseDoll, 49
db.c, 30	getRandomScore, 51
hour	new, 51 next_block_number
RESULT, 10	display_tetris.c, 35
	update.c, 64
i_block	apadic.c, 04
display_tetris.c, 35	o_block
update.c, 63	display tetris.c, 35
include/db.h, 11	update.c, 64
include/display_tetris.h, 16	OK
init_db db.c, 25	db.c, 21
db.b, 13	
init_tetris_table	passwd
main.c, 44	db.c, 30
main.c, ++	point
j block	main.c, 48
display_tetris.c, 35	RESULT, 10
update.c, 63	update.c, 65
•	port
I_block	db.c, 30
display_tetris.c, 35	print_color
update.c, 64	main.c, 45
LEFT	print_result
update.c, 54	db.c, 25

INDEX 71

	main.c, 45	check_one_line, 54
q		collision_test, 54 COMMAND, 53
,	main.c, 48	display_tetris, 56
		DOWN, 54
read	d_db	drop, 57
	db.c, 26	game, 63
DE	db.h, 14	GAME_END, 53
	ADME.md, 20	GAME_START, 53
ΠEI	D_COLOR display_tetris.h, 18	getch, 58
RE!	SET_COLOR	i_block, 63
	display_tetris.h, 18	j_block, 63
RES	SULT, 9	l_block, 64 LEFT, 54
	day, 10	move_block, 59
	hour, 10	next_block_number, 64
	min, 10	o block, 64
	month, 10	point, 65
	name, 10	RIGHT, 54
	point, 10	ROTATE, 54
DIO	year, 10	s_block, 65
RIG	••••	t_block, 65
DO.	update.c, 54 TATE	tetris_table, 66
ПО	update.c, 54	update, 60
	apatic.o, or	x, 66
s_b	lock	y, 66
	display_tetris.c, 36	z_block, 66
	update.c, 65	use_db db.c, 28
sav	e_result	user
	main.c, 46	db.c, 31
	db.c, 20	
	display_menu.c, 31	WHITE_COLOR
	display_tetris.c, 32	display_tetris.h, 18
	getch.c, 36 main.c, 38	write_db
310/	mam.c, 50	— <u> </u>
src/		db.c, 28
	new.c, 49	db.c, 28 db.h, 15
src/	new.c, 49 result.c, 52	db.h, 15
src/	new.c, 49	db.h, 15 x
src/	result.c, 52 rupdate.c, 52 ock	db.h, 15 x main.c, 48
src/	result.c, 52 result.c, 52 rupdate.c, 52 ock display_tetris.c, 36	db.h, 15 x
src/ src/ t_bl	rew.c, 49 result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65	db.h, 15 x main.c, 48 update.c, 66
src/	rew.c, 49 result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65db	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49
src/ src/ t_bl	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66
src/ src/ t_bl	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year
src/ src/ t_bl	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 _db db.c, 27 is_table display_tetris.h, 19	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10
src/ src/ t_bl	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 _db db.c, 27 is_table display_tetris.h, 19 main.c, 48	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR
src/ src/ t_bl	result.c, 52 result.c, 52 result.c, 52 ock display_tetris.c, 36 update.c, 65 c_db db.c, 27 res_table display_tetris.h, 19 main.c, 48 update.c, 66	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10
src/ src/ t_bl test tetri	result.c, 52 result.c, 52 result.c, 52 ock display_tetris.c, 36 update.c, 65 c_db db.c, 27 res_table display_tetris.h, 19 main.c, 48 update.c, 66	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR
src/ src/ t_bl test tetri	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18
src/ src/ t_bl test tetri	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block
src/ src/ t_bl test tetri	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 _db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30 late main.c, 46	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block display_tetris.c, 36
src/ src/ t_bl test tetri tmp	result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30 late main.c, 46 update.c, 60	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block display_tetris.c, 36
src/ src/ t_bl test tetri tmp	result.c, 49 result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30 late main.c, 46 update.c, 60 late.c	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block display_tetris.c, 36
src/ src/ t_bl test tetri tmp	result.c, 49 result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30 late main.c, 46 update.c, 60 late.c block, 62	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block display_tetris.c, 36
src/ src/ t_bl test tetri tmp	result.c, 49 result.c, 52 update.c, 52 ock display_tetris.c, 36 update.c, 65 db db.c, 27 is_table display_tetris.h, 19 main.c, 48 update.c, 66 db.c, 30 late main.c, 46 update.c, 60 late.c	db.h, 15 x main.c, 48 update.c, 66 y main.c, 49 update.c, 66 year RESULT, 10 YELLOW_COLOR display_tetris.h, 18 z_block display_tetris.c, 36