**Chess System**

**שדות/תכונות**

Enum ChessResult = {CHESS\_OUT\_OF\_MEMORY, CHESS\_NULL\_ARGUMENT, CHESS\_SUCCESS}

|  |  |  |
| --- | --- | --- |
| **חתימת הפונק** | **תיאור** | **ערכי שגיאה** |
| **ChessSystem** chessCreate(); | יוצרת מערכת חדשה (מקצה זיכרון למצביע למבנה מסוג Chess System) ומחזירה מצביע למבנה.  תחזיר אם לא הצליחה להקצות NULL . |  |
| **void** chessDestroy(ChessSystem chess); | מקבלת מצביע למערכת והורסת את המערכת – מפנה את כל הזכרון של המערכת.  אם התקבל NULL לא תעשה דבר | 1. CHESS\_INVALID\_ID  2. CHESS\_TOURNAMENT\_ALREADY\_EXISTS  3. CHESS\_INVALID\_LOCATION  4. CHESS\_INVALID\_MAX\_GAMES |
| **ChessResult** chessAddTournament(ChessSystem chess, int tournament\_id, int max\_games\_per\_player, cons char \*tournament\_location); | הפונקציה מוסיפה טורניר חדש למערכת. |  |
| ChessResult chessAddGame(ChessSystem chess, int tournament\_id, int first\_player, int second\_player, Winner winner, int play\_time); | תוסיף משחק חדש לטורניר במערכת | 1. CHESS\_INVALID\_ID  2. CHESS\_TOURNAMENT\_ALREADY\_EXISTS  3. CHESS\_TOURNAMENT\_ENDED  4. CHESS\_GAME\_ALREADY\_EXISTS  5. CHESS\_INVALID\_PLAY\_TIME  6. CHESS\_EXCEEDED\_GAMES |
| ChessResult chessRemoveTournament(ChessSystem chess, int tournament\_id); | הסרת טורניר ממערכת ניהול משחקי השחמט | 1. CHESS\_INVALID\_ID 2. CHESS\_TOURNAMENT\_NOT\_EXISTS |
| ChessResult chessRemovePlayer(ChessSystem ches, int player\_id); | הפונקציה תסיר את השחקן ממערכת ניהול משחקי שחמט ותעדכן את המנצח בכל משחק בהתאם | 1. CHESS\_INVALID\_ID 2. CHESS\_PLAYER\_NOT\_EXISTS |
| **ChessResult** chessEndTournament(ChessSystem chess, int tournament\_id); | הפונקציה תסיים את התחרות ותחשב את המנצח של התחרות | 1. CHESS\_INVALID\_ID 2. CHESS\_TOURNAMENT\_NOT\_EXISTS 3. CHESS\_TOURNAMENT\_ENDED |
| **double** chessCalculateAveragePlayTime(ChessSystem chess, int player\_id, ChessResult \*chess\_result) | הפונקציה מחזירה את ממוצע זמני משחק עבור שחקן מסויים | 1.CHESS\_INVALID\_ID  2. CHESS\_PLAYER\_NOT\_EXISTS |
| **ChessResult** chessSavePlayersLevels(ChessSystem chess, FILE\* file); | הפונקציה מחשבת את דירוג כל השחקנים במערכת ומייצאת לקובץ לפי סדר הדירוג בעדיפות ראשונה וסדר התז בעדיפות שנייה | 1.CHESS\_SAVE\_FAILURE |
| chessResult chessSaveTournamentStatistics(ChessSystem chess, char\* path\_file); | הפונקציה תדפיס לקובץ את הסטטיסטיקות עבור **כל טורניר שהסתיים** | 1.CHESS\_NO\_TOURNAMENTS\_ENDED  2. CHESS\_SAVE\_FAILURE |

**Game**

|  |  |  |
| --- | --- | --- |
| **חתימת הפונק** | **תיאור** | **ערכי שגיאה** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**שדות/תכונות**

int player\_1\_id

int player\_2\_id

enum Outcome = {FIRST,SECOND,DRAW}

int time

**Player**

|  |  |  |
| --- | --- | --- |
| **חתימת הפונק** | **תיאור** | **ערכי שגיאה** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**שדות/תכונות**

**Tournament**

|  |  |  |
| --- | --- | --- |
| **חתימת הפונק** | **תיאור** | **ערכי שגיאה** |
|  |  |  |
|  |  |  |
|  |  |  |
| void closeTournament(); | תסגור את הטורניר ותחשב את המנצח, ותעדכן את המנצח |  |
|  |  |  |
|  |  |  |
|  |  |  |

**שדות/תכונות**

int tour\_id

char \*location

Map games

int winner\_id

bool is\_closed