Main method start

Get the threads quantity to run. Then open base game text file

Basegame file opened correctly? — No → **ABORT**

Yes ↓

Read the file and save its lines into a game_t* record used as base game

Basegame record filled? — No → **ABORT**

Yes ↓

Close the file

Check if threads quantity is major than game columns, if so set the threads quantity to game columns

Create a shared data record containing best games matrix of size high enough to contain all threads best games with their depth

Create a private data record containing base game, best score for the thread, number of threads, its thread number, a pointer to shared data, and a boolean vector to flag when a game must be saved into best games record

run_threads method

Compute init and end cols to iterate, according to block mapping

For column into init and end, and for every rotation

Allocate the figure for level 0 in the actual column and rotation

Execute find_best_score method for level 1

save_best_game[0] is True? — No → Remove the figure → Destroy game 0 clone

Yes ↑

Save best game for level 0

. . .

Wait for all the threads to complete their job → Get the thread with the best score → **Save best games for detected thread with best scores into txt files and destroy all the memory allocated variables**

find_best_score method

Clone current level game

Deepest level reached? — Yes → Compute score for current game → Score < Best score? — Yes → Set save_best_game True for all levels and replace the best score

No ↓ (Deepest level reached?)

For every column and every rotation

Allocate the figure for current level in the corresponding column and rotation

Execute find_best_score method for next level

save_best_game flag for current level == True — No → Remove the figure

Yes ↑

Save best game for current level

More columns and rotations? — Yes → Allocate the figure for current level in the corresponding column and rotation

No ↓

Destroy current level clone and return score