MANCALA GAME

C++ / UBUNTU 18.04.2 LTS

IMPLEMENTED USING DATA STRUCTURES

BY EKRAMUL SAWRID, PROF. CHI-HIM “TIMMY” LIU

Synopsis

In this project, it includes code the code for the mancala game. It uses two classes, the Board class and the State (tree) class. The search functions included (Breath-first search, Dept-first search, and Minimax search) uses these classes to find a winning state. This project also includes a function that lets you play against the computer that bases it moves on dept-first search. Note that although it complies, runs, and is playable, this function MAY have a bug that still needs working. All other code are working properly according my tests. Human vs human also included.

Details

The board class (Board.h and Board.cpp) is based around manipulating a private array of type size_t.

The tree class (State.h and State.cpp) uses a Board object and has pointers to act like a tree.

The search functions (in Functions.h and Functions.cpp) uses the tree and board class to conduct its searches. Prints the board object of every node (tree class) it searches.

main.cpp and makefile included.

See code files comments and PowerPoint for more details**.**

How to compile and run

To limit the number of files, all functions are called in main.cpp has shown:

```
#include "Board.h"
#include "State.h"
#include "Functions.h"

//EKRAMUL SAWRID
//MANCALA GAME

int main(){
        Board input;
        //input.customize(0,1,2,3,4,5,6,7,8,9,10,11,12,13);
        //input.print();
        tNode* root=new tNode(input);

        //BFS(root);
        //DFS(root);
        //MMS(root);
        //MultiPlay(input);
        //Play(input);

        return 0;
}
```

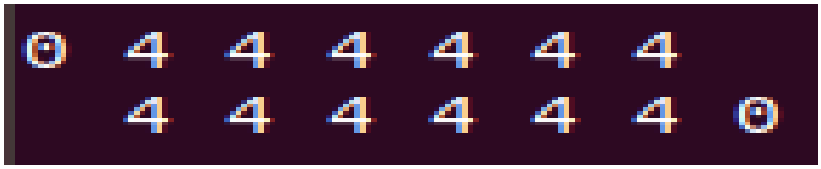
To **compile and run**, type in terminal: make runMain

This runs main.cpp

Example: If you want to do BFS, just uncomment BFS(root), save the file, compile and run.

If you want to **configure the board** or start with different board positions, uncomment input.customize(*parameters of size_t)*. This functions lets you set the number of pieces in each hole of the board. Each number represents the index number. Save file, compile and run.

By default:



After input.customize(0,1,2,3,4,5,6,7,8,9,10,11,12,13):