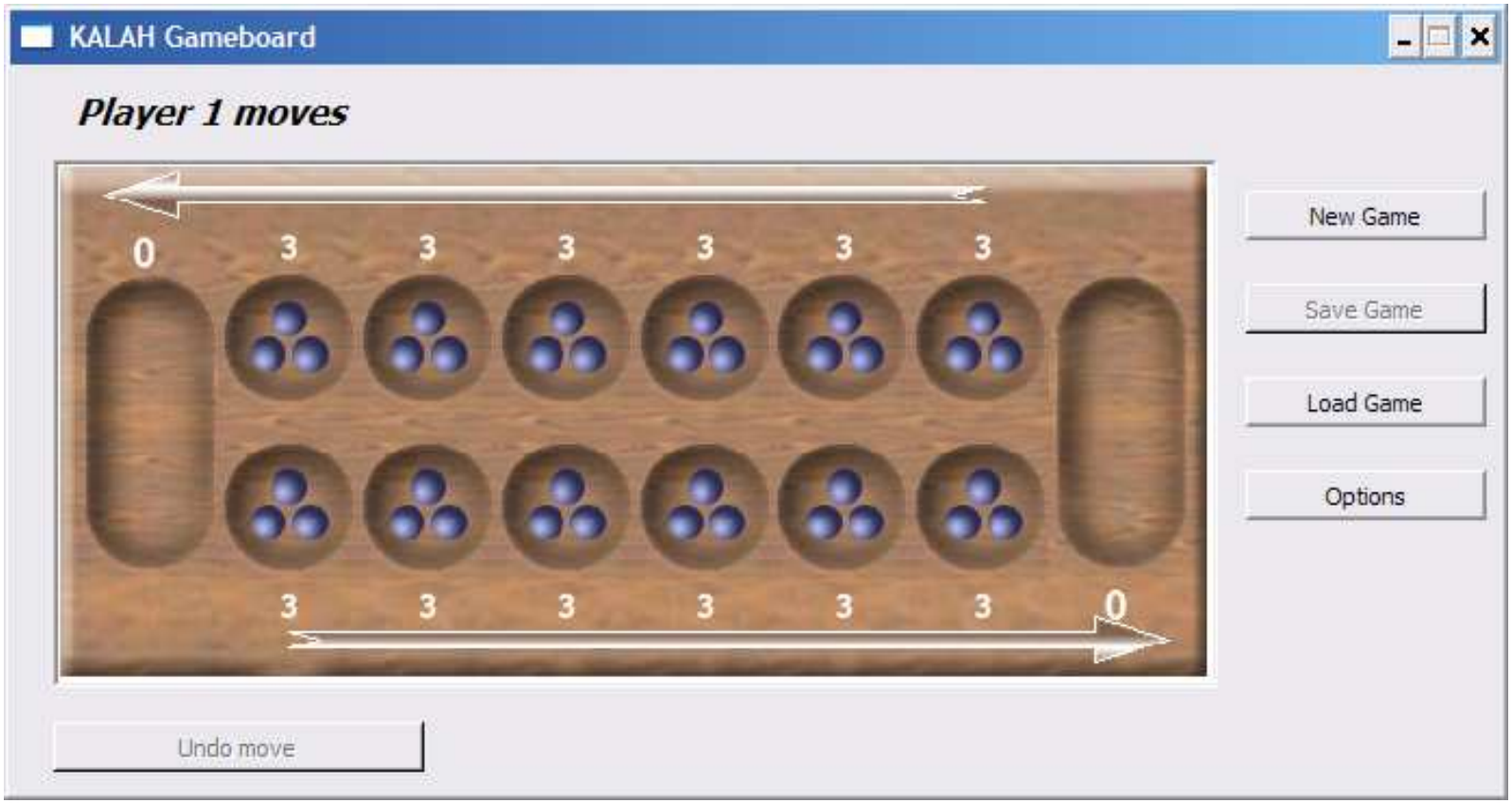
The Summer School “Achievements and Applications of Contemporary Informatics, Mathematics and Physics”

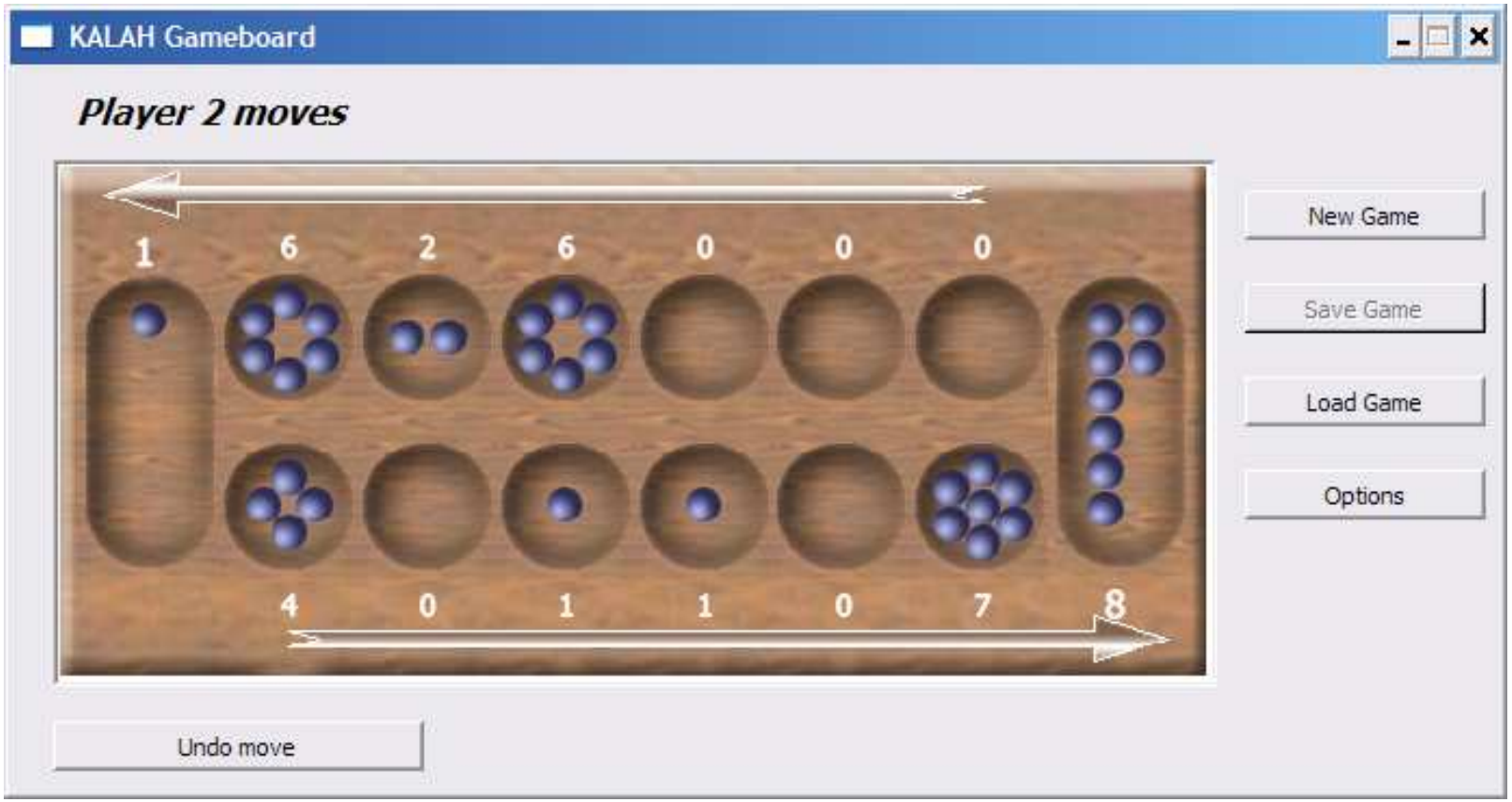
# Search Heuristics

## Practice. Part III
## Kalah Game

# Introducing Kalah Gameboard



# Introducing Kalah Gameboard



# Recall main course ides

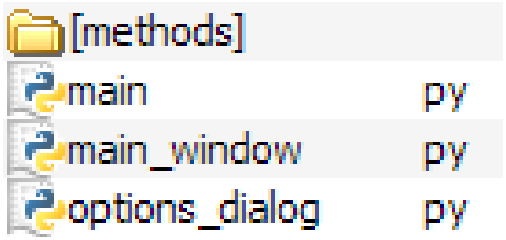
- Searching paradigm:
  - Searching STATE
  - Searching SPACE
  - Searching STRATEGY

    STRATEGY = Method + Neighborhood + Heuristics
- We use this approach for games

  STRATEGY = Minimax + Neighborhood + Heuristics

# Kalah Gameboard architecture

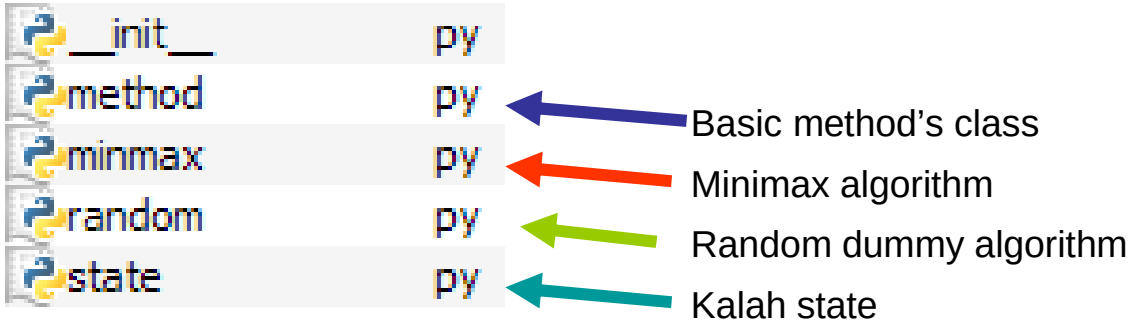
Main folder




main.py – is the project main module.

Run it to work with Kalah Gameboard

Methods package folder



# How to start?

- Download the TSP Problem Solver from https://github.com/olexiim/kalah on GitHub. You can download it as a one zip archive file: https://github.com/olexiim/kalah/archive/master.zip
- Unpack the archive to get full package

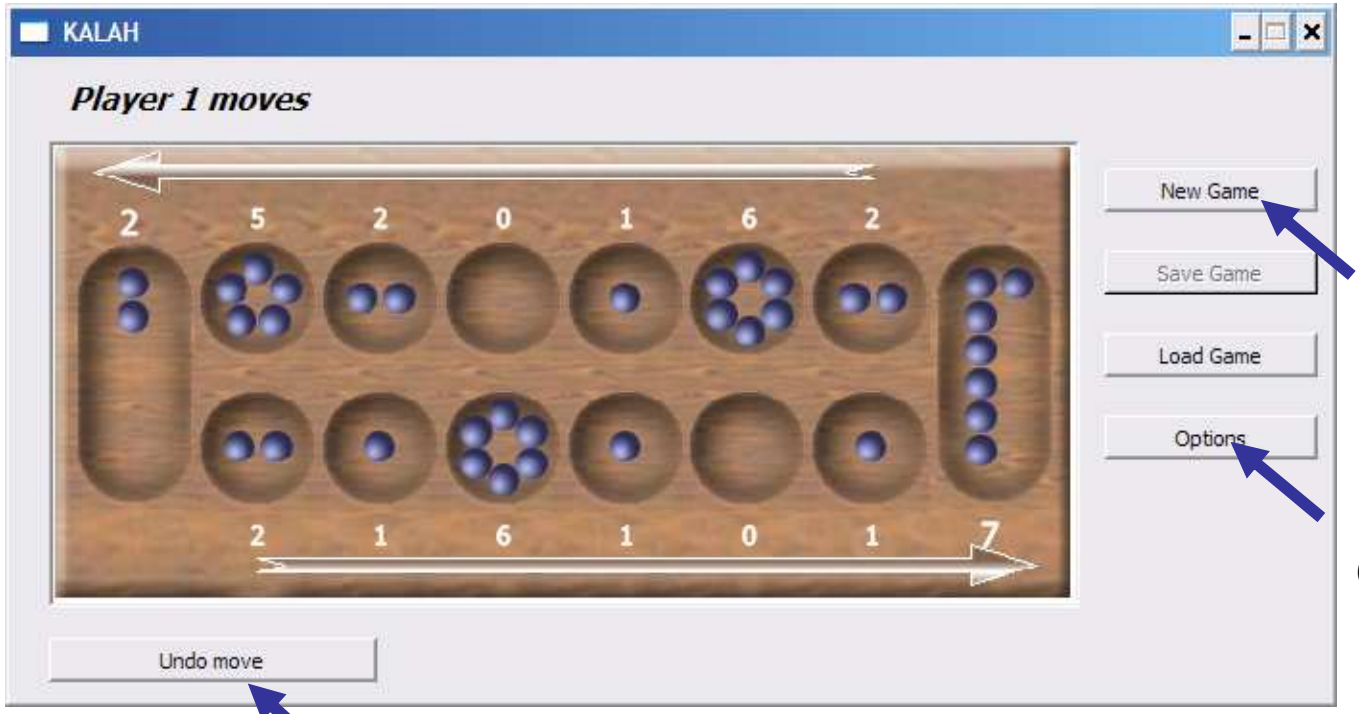
## On Windows

- Run Spyder (or another IDE)
- Open main.py
- Run

## On Linux (Ubuntu)

- Run IPython terminal
- cd to package's folder
- type "run main.py"

# Kalah Gameboard GUI

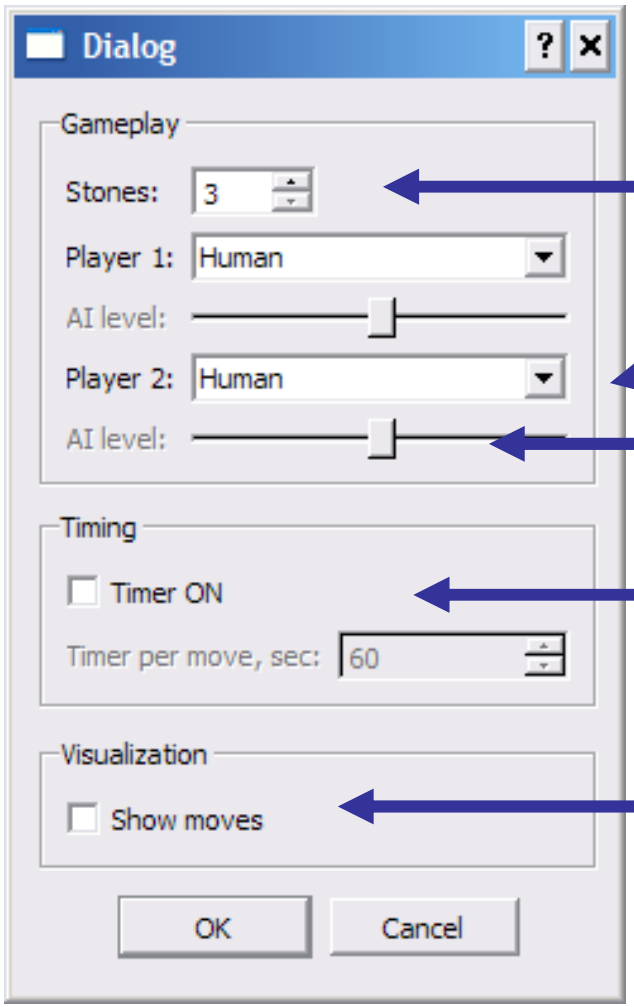



New game

Options dialog

Undo last human player move

# Kalah options




Select initial number of stones (3 ... 6)

Select player

Select computer level (1..5)

Enable time limit (in sec) for each player's move

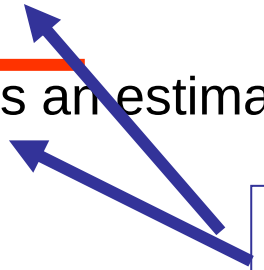
Show animation of moves

# Practice tasks

- Play Kalah game with your friend
  - Became familiar with game rules and gain some basic experience
- Create your own gaming bot
  - By implementing your own strategy
    - Refer to methods/random.py for example
  - By changing heuristic function of Minimax
    - Refer to methods/minmax.py – it has detailed comments

# Working with minimax method

- Make-Move
  - starts to build a searching tree with creation of a root node and generating its neighbors
- Max-Value
  - returns heuristic function value for a node in which the player makes the move
- Min-Value
  - returns heuristic function value for a node where the opponent makes the move
- Terminal-Test
  - checks if a current state could be expanded or not
- Utility
  - calculates an estimation of who will win probably in a specific state



This things you need to change in order to create your own Minimax

# Prepare to contest

- Up to August, 8, 23:59 we accept your bots for playing Kalah
- Later on we will run a contest between your bots
- The winners will gain prizes!
- All details a little bit later on:
  - summerschool.ssa.org.ua/program/acs-stream/search-heuristics

If you have any questions, please do not hesitate to ask your tutor

Work in teams: you can help each other to go deep into problem