Current timestamp of the game: NNNNNNNN
Resources consumed: 120 (and growing...)

	1.01	**1.02**	**1.03**	**1.05**	**10**	**11**	**12**	**20**	**50**	**75**	**100**	**...**	**1000**
Player 1	25 pos -5 left	30 pos -4 left	60 pos -3 left	40 pos -2 left	50 pos -1 left		2 pos. +1 right	10 pos. +2 right	30 pos. +3 right	2 pos. +4 right	5 pos. +5 right		
Player 2													
Player n													

This is the board:

- on the X-Axis there are all possible positions of the players: it's some sort of exponential scale, starting from 1.01, ending to 1000
- on the Y-Axis there are the players; the number of players can vary from game to game
- After the main board is ready, it takes from a file (snapshots.json) the list of the snapshots
- a snapshot is composed by a timestamp, and a list of properties for every player involved in the game, example for player 1:


```
[
  {
    "timestamp": "2019-10-06 06:49:17",
    "tot_resources": 120,
    "players": [
      {
        "name": "Player 1",
        "segments": {
          "left": [
            {
              "pos": 1.01,
              "size": 25
            },
            {
              "pos": 1.02,
              "size": 30
            },
            {
              "pos": 1.03,
              "size": 60
```

```
            },
            {
              "pos": 1.05,
              "size": 40
            },
            {
              "pos": 10,
              "size": 50
            }
          ],
          "right": [
            {
              "pos": 12,
              "size": 2
            },
            {
              "pos": 20,
              "size": 10
            },
            {
              "pos": 50,
              "size": 30
            },
            {
              "pos": 75,
              "size": 2
            },
            {
              "pos": 100,
              "size": 5
            }
          ]
        }
      }
    ]
...
  }
]
```

- a player is a sort of snake composed two sides (left and right)
- every side can contain a maximum of five segments with position and size
- every game has:
    - Properties
        - total amount of resources consumed (changes at every step of the game)
    - Inputs Data
        - rows from snapshots.json

- the board tracks the total amount of resources consumed
- Players move according to directives given to the board from outside (snapshots.json file)
- **In this particular game the number of winners is the number of players -1, there will be only one loser**
- A player wins if its last segment of left side reaches position 1000
- During the game the agent should choose N players and wait until the end of the game (at the very beginning the algorithm may choose only one runner, then it can increase the number of players)
- If the player chosen by the agent wins, the agent is rewarded like follows: 100 / (<position of the right side head when the player is chosen> -1)
- if the player chosen by the agent loses, there will be a negative reward: -100 points
- So the goal is to find N creditable winners as soon as the agent can, to maximize rewards (according to the formula above)
- Some suggestions:
    - as you can see from snapshots.json near the end of the game, distance between players increases
    - a good approach, maybe, could be analyzing “trends” of players, because all the players -1, before or after, will reach position 1000
- the use of any visual representation of the game is needed