DMSC AI Tournament

Second edition

The day's program

Schedule:

- 13:00 - 13:30: Presentation about the game
- 13:30 - 18:00: Work individually (or in teams of 2) on your AI contestant(s)
- Tournament at 18:00 in Curie/Holmes

How we run this day

- Have fun with the challenge
- It is (most probably) possible to cheat, please show good sportsmanship

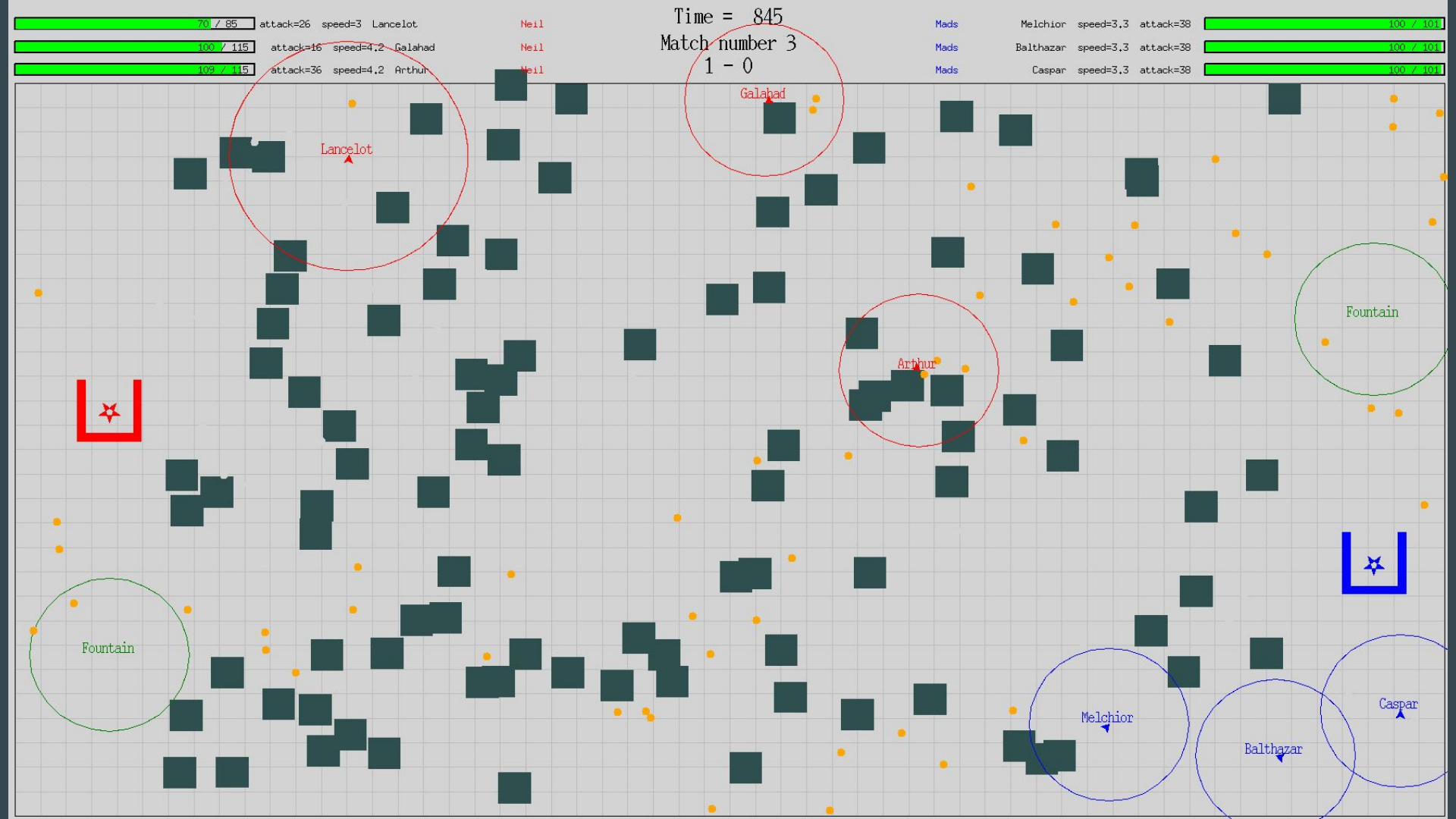
About the game

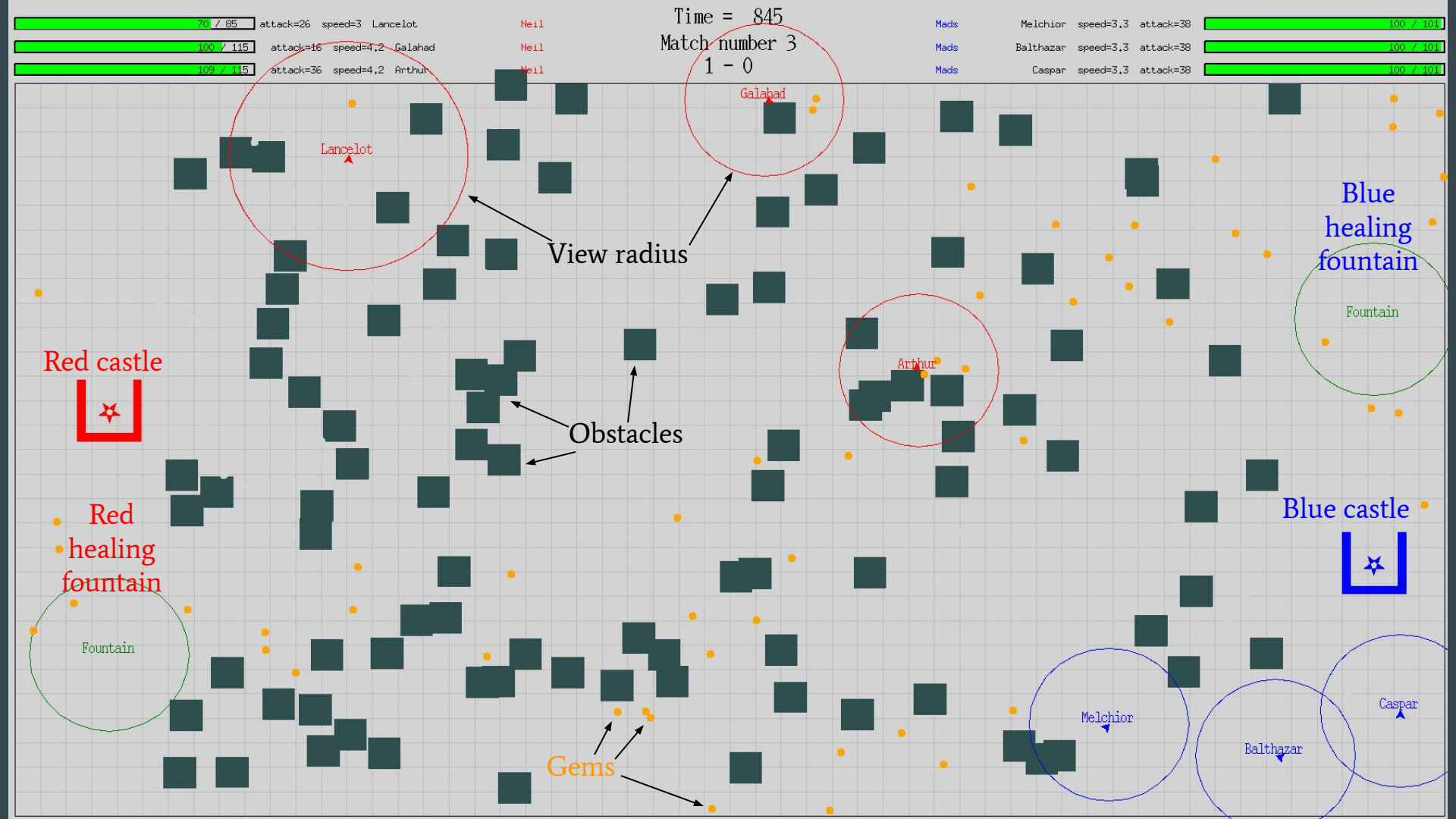
- Inspired by “Heroes of the Storm”

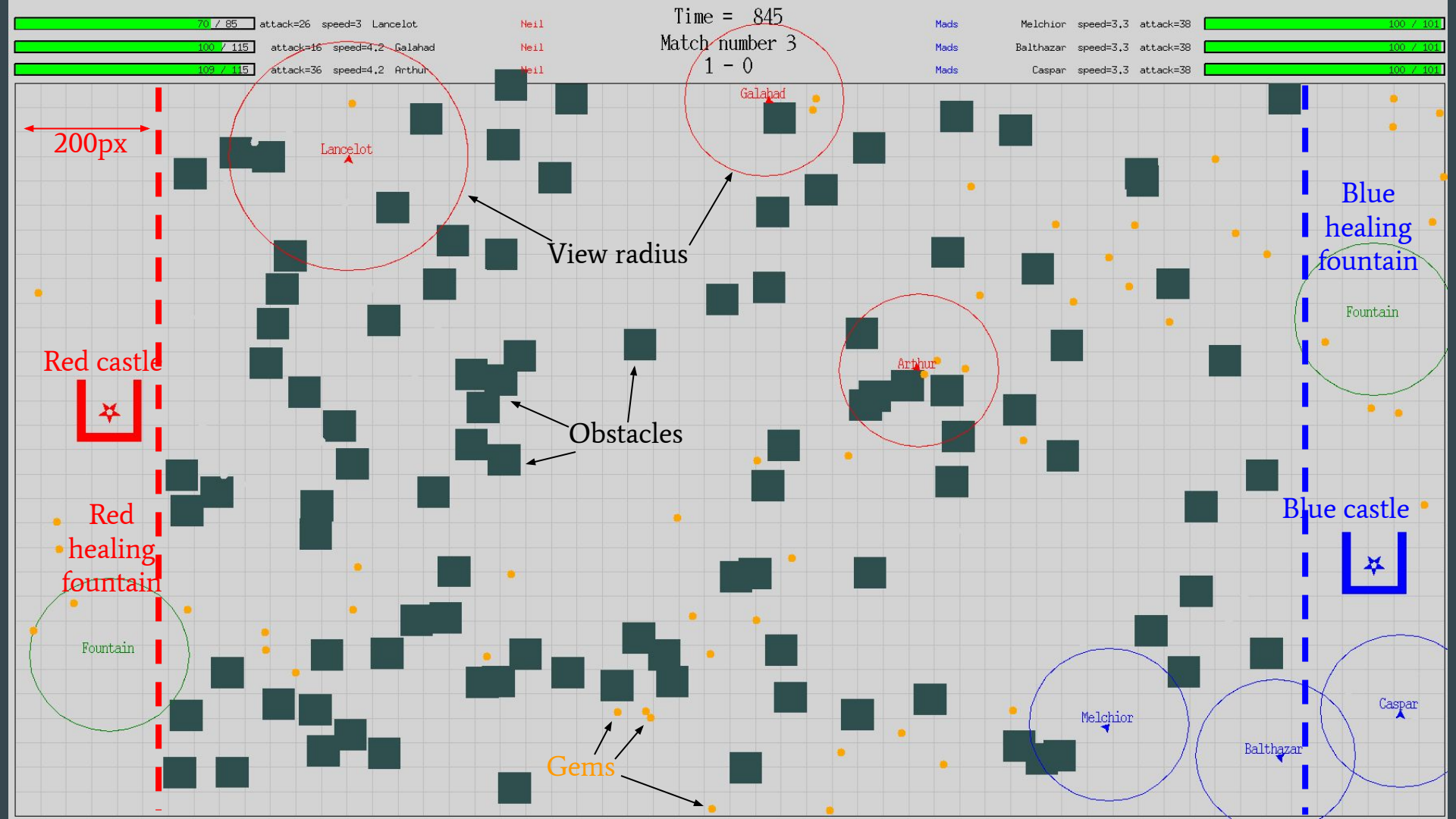
- Implementation (engine & score) heavily inspired by Mads’ “Naval Exercise”
- Python with Turtle graphics (1400 lines, 0 unit tests)
- Should need no dependencies other than `numpy`

  (needs `pyyaml` to run the tournament)


QUEST


70 / 85 attack=26 speed=3 Lancelot Neil
100 / 115 attack=16 speed=4.2 Galahad Neil
109 / 115 attack=36 speed=4.2 Arthur Neil
Time = 845
Match number 3
1 - 0
Mads Melchior speed=3.3 attack=38
Mads Balthazar speed=3.3 attack=38
Mads Caspar speed=3.3 attack=38
100 / 101
100 / 101
100 / 101
Lancelot
Galahad
Arthur
Fountain
Fountain
Melchior
Balthazar
Caspar


70 / 85 attack=26 speed=3 Lancelot Neil
100 / 115 attack=16 speed=4.2 Galahad Neil
109 / 115 attack=36 speed=4.2 Arthur Neil
Time = 845
Match number 3
1 - 0
Mads Melchior speed=3.3 attack=38 100 / 101
Mads Balthazar speed=3.3 attack=38 100 / 101
Mads Caspar speed=3.3 attack=38 100 / 101
Lancelot
Galahad
Arthur
Melchior
Balthazar
Caspar
View radius
Obstacles
Gems
Red castle
Red healing fountain
Fountain
Blue healing fountain
Fountain
Blue castle


70 / 85 attack=26 speed=3 Lancelot Neil
100 / 115 attack=16 speed=4.2 Galahad Neil
109 / 115 attack=36 speed=4.2 Arthur Neil
Time = 845
Match number 3
1 - 0
Mads Melchior speed=3.3 attack=38
Mads Balthazar speed=3.3 attack=38
Mads Caspar speed=3.3 attack=38
100 / 101
100 / 101
100 / 101
200px
Lancelot
Galahad
Arthur
View radius
Obstacles
Red castle
Red healing fountain
Fountain
Blue healing fountain
Fountain
Blue castle
Melchior
Balthazar
Caspar
Gems

Game rules (1/3)

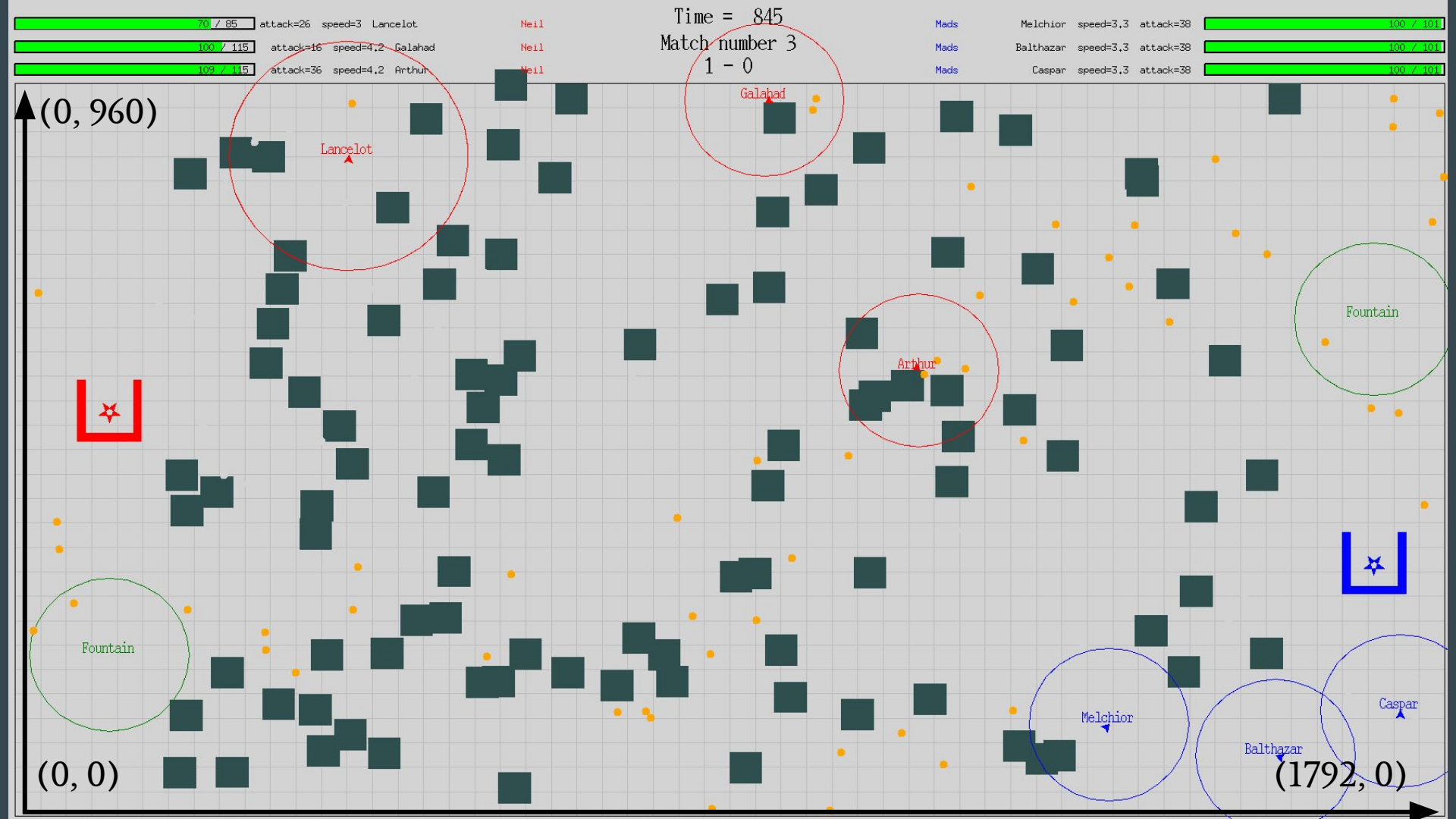
Two ways to win:

1. Capture the flag in the opposing team's castle (no need to bring it home)
2. Kill all members of the opposing team

- Draw after 180s

Game map:

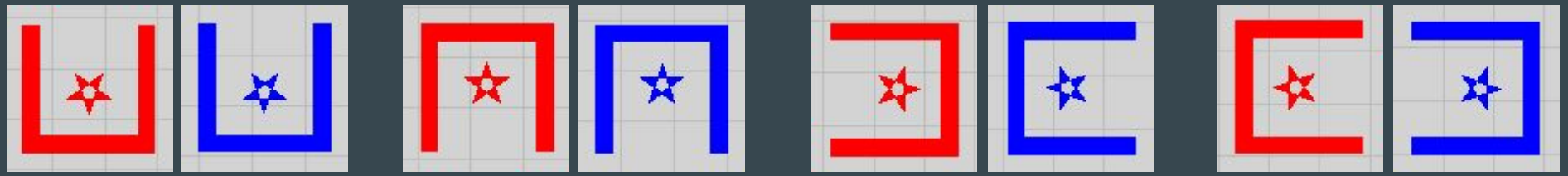
- Dimensions: `nx = 56x32 = 1792 ; ny = 30x32 = 960`
- Coordinate system: lower left = `(0, 0)`, upper right = `(1792, 960)`
- Knights cannot go outside boundaries or through obstacles
- 100 obstacles and 100 gems, randomly positioned at the start of each round


70 / 85 attack=26 speed=3 Lancelot Neil
100 / 115 attack=16 speed=4.2 Galahad Neil
109 / 115 attack=36 speed=4.2 Arthur Neil
Time = 845
Match number 3
1 - 0
Mads Melchior speed=3.3 attack=38 100 / 101
Mads Balthazar speed=3.3 attack=38 100 / 101
Mads Caspar speed=3.3 attack=38 100 / 101
(0, 960)
Lancelot
Galahad
Arthur
Fountain
Fountain
Melchior
Balthazar
Caspar
(0, 0)
(1792, 0)

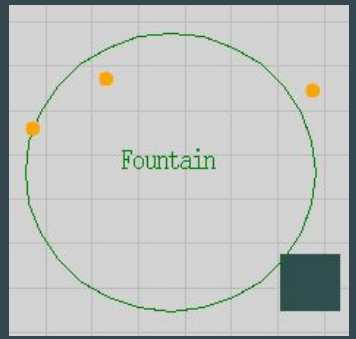
Game rules (2/3)

- **Castles:** 4 possible configurations:

1 2 3 4



- **Fountains of Youth:**
  - Located at `ny / 3 = 320` pixels from the castle
  - Knight will heal at a rate of `15 * dt` when inside the fountain radius
  - Enemy fountains are invisible and have no healing effect



Game rules (3/3)

Fights:

- Whenever two or more knights from opposing teams meet in the same `32x32` cell, they will fight
- Attack from each team is summed up, summed attack force is split up evenly between defenders of a same team
- Knights have a cooldown of `3s` after having fought (light blue health bar)

Gems:

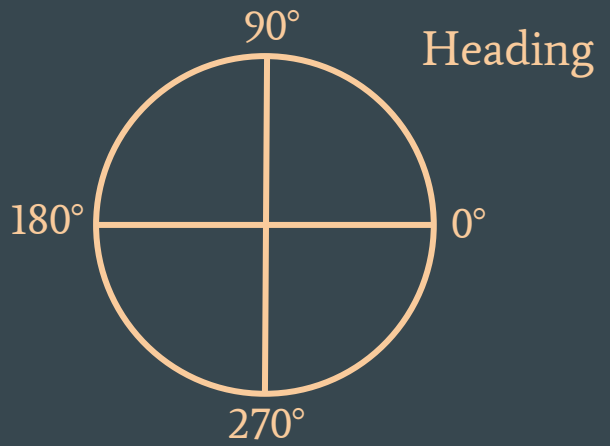
- Picking up a gem will give a random bonus to health, attack or speed to **all** team members

Your character: The Knight

- Moves forwards every time-step a distance = `speed * dt` (`dt=1/30` by default)
- Main properties:
  - `x, y`
  - `attack`
  - `health, max_health`
  - `speed`
  - `cooldown`
  - `heading`
  - `view_radius`
- Knows its local environment within `view_radius` (gems, obstacles, enemies, enemy flag)
- Knows friends info, location of own flag & own fountain at all times

To control the Knight - The AI

- **Info (dict) it receives every time-step:**
  - `local_map`: local_map (ndarray: obstacles = 1, gems = 2, else = 0, no info = -1)
  - `friends`: properties of friends no matter where they are
  - `enemies`: properties of enemies within `view_radius`
  - `gems`: position of gems (but not what kind of bonus they give)
  - `flags`: location of own flag at all times, location of enemy flag if within `view_radius`
  - `me`: my own properties
  - `fountain`: location of my team's fountain of youth
- **Variables to be set by AI:**
  - `heading` (in degrees, 0=right to 360, anti-clockwise)
  - `goto` (x, y position)
  - `left` (degrees from current heading)
  - `right` (degrees from current heading)
  - `stop` (not reset at every time-step)
  - `message`



AI template

Define:

- `__init__()`
- `run()`

Set one of:

- `heading`
- `goto`
- `left`
- `right`
- `stop`

```
4  from ai import BaseAI
5
6  CREATOR = 'JohnDoe'
7
8
9  class TemplateWarrior(BaseAI):
10
11     def __init__(self, *args, **kwargs):
12         super().__init__(*args, creator=CREATOR, kind='warrior', **kwargs)
13         self.previous_position = [0, 0]
14         self.previous_health = 0
15
16     def run(self, t: float, dt: float, info: dict):
17         me = info['me']
18
19         if all(me['position'] == self.previous_position) or (
20                 me['health'] < self.previous_health):
21             self.heading = np.random.random() * 360.0
22
23         elif len(info['flags']) == 2:
24             flag_pos = info['flags'][self.opposing_team]
25             self.goto = flag_pos
26
27         elif len(info['enemies']) > 0 and (me['cooldown'] == 0):
28             target = info['enemies'][0]
29             self.goto = [target['x'], target['y']]
30
31         elif info['gems']:
32             self.goto = [info['gems']['x'][0], info['gems']['y'][0]]
33
34         self.previous_position = me['position']
35         self.previous_health = me['health']
36
```

	Warrior	Scout	Healer
Health	100	70	100
Attack	30	20	10
Speed (max)	60 (150)	45 (90)	60 (210)
View radius	100	150	100
Additional			Heal friends within `view_radius` 

Scout example

```python
from ai import BaseAI


class NeilScout(BaseAI):

    def __init__(self, *args, **kwargs):
        super().__init__(*args, creator='Neil', kind='scout', **kwargs)
        self.previous_position = [0, 0]
        self.previous_health = 0

    def run(self, t, dt, info):
        me = info['me']

        if all(me['position'] == self.previous_position) or (
                me['health'] < self.previous_health):
            self.heading = np.random.random() * 360.0

        elif len(info['flags']) == 2:
            flag_pos = info['flags'][self.opposing_team]
            self.goto = flag_pos

        elif len(info['enemies']) > 0:
            name = list(info['enemies'].keys())[0]
            target = info['enemies'][name]
            self.heading = (self.towards(target['x'], target['y']) + 180) % 360

        elif info['gems']:
            self.goto = [info['gems']['x'][0], info['gems']['y'][0]]

        self.previous_position = me['position']
        self.previous_health = me['health']
```

Tips & recommendations

1. Start making just one AI, and have 3 copies of it in your team
2. Remember: knights can talk to each other! (via `message`)

What's next?

- Clone repo at: `https://github.com/nvaytet/quest`
- Tournament will happen in 2 phases:
  - First phase will be standard 1 v 1 (best of 3)
  - Second phase: randomly team up different people's AIs
  - Most rounds won at the end of the two phases wins
  - Tie will be decided by the most matches won

https://github.com/nvaytet/quest

Summary

Info	Knight properties	Set by AI	AI helper functions
`local_map`	`name`	`heading`	`get_distance(x, y)`: from current knight position
`friends`	`x, y, position`	`goto`	
`enemies`	`attack`	`left`	`towards(x, y)`: heading angle to get to point
`gems`	`health, max_health`	`right`	`heading_from_vector([v_x, v_y])`
`flags`	`speed`	`stop`	`vector_from_heading(heading)`
`me`	`view_radius`	`message`	
`fountain`	`heading, vector`		
	`cooldown`		
	`message`		
	`team`		