solution

April 28, 2019

```
In [1]: from tqdm import tqdm
        import itertools
        import math
        import numpy as np
```

1

-. , - 3Œ3. .

```
In [2]: def get_i_j(square_number):
            return (square_number // 3, square_number % 3)
```

```
In [3]: def check_winning_condition(desk, player, i, j):
            return ((player == desk[i][0] == desk[i][1] == desk[i][2]) or
                    (player == desk[0][j] == desk[1][j] == desk[2][j]) or
                    (player == desk[0][0] == desk[1][1] == desk[2][2]) or
                    (player == desk[0][2] == desk[1][1] == desk[2][0]))
```

2 1-

https://en.wikipedia.org/wiki/Tic-tac-toe#Combinatorics
When considering only the state of the board, and after taking into account board symmetries (i.e. rotations and reflections), there are only 138 terminal board positions.
( ) 138 (..).

```
In [4]: def add_desk(desk, desks):
            if not (
                (tuple(map(tuple, np.rot90(desk, 0)))) in desks or
                (tuple(map(tuple, np.rot90(desk, 1)))) in desks or
                (tuple(map(tuple, np.rot90(desk, 2)))) in desks or
                (tuple(map(tuple, np.rot90(desk, 3)))) in desks or
                (tuple(map(tuple, np.rot90(desk.T, 0)))) in desks or
                (tuple(map(tuple, np.rot90(desk.T, 1)))) in desks or
                (tuple(map(tuple, np.rot90(desk.T, 2)))) in desks or
                (tuple(map(tuple, np.rot90(desk.T, 3)))) in desks
                    ):
                desks.add(tuple(map(tuple, desk)))
```

```
In [5]: desks_set_first = set()

        for perm in tqdm(list(itertools.permutations(range(9)))):
            desk = np.array([[5, 5, 5], [5, 5, 5], [5, 5, 5]])
            for n, step in enumerate(perm):
                player, i, j = (n % 2, *get_i_j(step))
                desk[i][j] = player
                if check_winning_condition(desk, player, i, j) or n == 8:
                    add_desk(desk, desks_set_first)
                    break
```

```
100%|| 362880/362880 [00:25<00:00, 14290.16it/s]
```

```
In [6]: len(desks_set_first)
```

```
Out[6]: 138
```

3 2-

( ) 764 , .

```
In [7]: desks_set_second = set()

        for perm in tqdm(list(itertools.permutations(range(9)))):
            desk = np.array([[5, 5, 5], [5, 5, 5], [5, 5, 5]])
            for n, step in enumerate(perm):
                player, i, j = (n % 2, *get_i_j(step))
                desk[i][j] = player
                add_desk(desk, desks_set_second)
                if check_winning_condition(desk, player, i, j):
                    break
```

```
100%|| 362880/362880 [01:57<00:00, 3096.81it/s]
```

```
In [8]: len(desks_set_second)
```

```
Out[8]: 764
```

```
In [9]: desks_set_first.issubset(desks_set_second)
```

```
Out[9]: True
```