```
import itertools

def get_sea(size, alive_cons):
    return [[1 if (i, j) in alive_cons else 0
             for j in xrange(size)]
            for i in xrange(size)]


def get_neighbors(con):
    x, y = con
    neighbors = [(x + i, y + j)
                 for i in xrange(-1, 2)
                 for j in xrange(-1, 2)
                 if not i == j == 0]
    return neighbors


def calculate_alive_neighbors(con, alive_cons):
    return len(filter(lambda x: x in alive_cons, get_neighbors(con)))


def is_alive_con(con, alive_cons):
    alive_neighbors = calculate_alive_neighbors(con, alive_cons)
    if (alive_neighbors == 3 or
            (alive_neighbors == 2 and con in alive_cons)):
        return True
    return False


def nexr_step(alive_cons):
    sea = itertools.chain(*map(get_neighbors, alive_cons))
    next_sea = set([con
                    for con in board
                    if is_alive_con(con, alive_cons)])
    return list(next_sea)


def is_correct_con(size, con):
    x, y = con
    return all(0 <= coord <= size - 1 for coord in [x, y])


def correct_cons(size, cons):
    return filter(lambda x: is_correct_con(size, x), cons)
```

```
def print_sea(sea):
    for line in sea:
        print line
    print


def main():
    size = 5
    sea = [(1, 2), (2, 3), (3, 1), (3, 2), (3, 3)]
    print_sea(get_sea(size, sea))
    for _ in xrange(1000):
        sea = correct_cons(size, next_step(sea))
        print_sea(get_sea(size, sea))
```