Transition Matrix of Markov Chain for n=4

	0000	0001	0010	0100	1000	1100	1010	1001	0110	0101	0011	1110	1101	1011	0111	1111
0000	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0001	0	1-3(€/6)	€/6	€/6	€/6	0	0	0	0	0	0	0	0	0	0	0
0010	0	€/6	1-3(€/6)	€/6	€/6	0	0	0	0	0	0	0	0	0	0	0
0100	0	€/6	€/6	1-3(€/6)	€/6	0	0	0	0	0	0	0	0	0	0	0
1000	0	€/6	€/6	€/6	1-3(€/6)	0	0	0	0	0	0	0	0	0	0	0
1100	0	0	0	0	0	1-4€/6	€/6	€/6	€/6	€/6	0	0	0	0	0	0
1010	0	0	0	0	0	(1-€)/6	1-4(1-€)/6	(1-€)/6	(1-€)/6	0	(1-€)/6	0	0	0	0	0
1001	0	0	0	0	0	€/6	€/6	1-4€/6	0	€/6	€/6	0	0	0	0	0
0110	0	0	0	0	0	€/6	€/6	0	1-4€/6	€/6	€/6	0	0	0	0	0
0101	0	0	0	0	0	(1-€)/6	0	(1-€)/6	(1-€)/6	1-4(1-€)/6	(1-€)/6	0	0	0	0	0
0011	0	0	0	0	0	0	€/6	€/6	€/6	€/6	1-4€/6	0	0	0	0	0
1110	0	0	0	0	0	0	0	0	0	0	0	1-3(€/6)	€/6	€/6	€/6	0
1101	0	0	0	0	0	0	0	0	0	0	0	€/6	1-3(€/6)	€/6	€/6	0
1011	0	0	0	0	0	0	0	0	0	0	0	€/6	€/6	1-3(€/6)	€/6	0
0111	0	0	0	0	0	0	0	0	0	0	0	€/6	€/6	€/6	1-3(€/6)	0
1111	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Montecarlo Simulation

€=0.3

Because it is assumed that there are n//2 or n//2+1 type 1 agents, only the highlighted portion of the transition matrix above is considered in the simulation. Using the script in q1b1_script.py, the following output for the Montecarlo simulation was:

```
For n=4, the stationary distribution is [0.213, 0.088, 0.215, 0.197, 0.093, 0.194]
```

And the numerically calculated stationary distribution was:

```
The numerically calculated stationary distribution is: [ 0.20588235  0.08823529  0.20588235  0.20588235  0.08823529  0.20588235]
```

In both distributions, the elements correspond to the states 1100, 1010, 1001, 0110, 0101, and 0011 respectively.

The two stationary distributions are relatively similar with the second and fifth elements being distinctly smaller than the other 4 which are approximately the same size. From this model, it can be concluded that states 1010 and 0101, the states where agents of the same type do not have neighbours of the same type, are significantly less likely to occur than the other states. This suggests that, even with a probability of 'failure', agents of the same type will naturally flock together causing segregation.

Linear Structure

In the scenario where agents live on simple linear structure instead of a cycle, the transition matrix (assuming there are n//2 or n//2+1 type 1 agents) would be:

	1100	1010	1001	0110	0101	0011
1100	1-4€/6	€/6	€/6	€/6	€/6	0
1010	(1-€)/6	1-4(1-€)/6	(1-€)/6	(1-€)/6	0	(1-€)/6
1001	(1-€)/6	€/6	1-2/6	0	€/6	(1-€)/6
0110	(1-€)/6	€/6	0	1-2/6	€/6	(1-€)/6
0101	(1-€)/6	0	(1-€)/6	(1-€)/6	1-4(1-€)/6	(1-€)/6
0011	0	€/6	€/6	€/6	€/6	1-4€/6

Highlighted rows are different than the same rows in the cycle transition matrix. The rest of the full transition matrix (that does not follow the assumption) is the same as the corresponding sections in the full transition matrix for the cyclical model. The script in q1b2_script.py outputted the following for €=0.3:

```
For n=4, the stationary distribution is [0.276, 0.117, 0.12, 0.117, 0.116, 0.254]
The numerically calculated stationary distribution is: [ 0.26923077  0.08823529  0.14253394  0.14253394  0.08823529  0.26923077]
```

There are now 3 distinct classes of states: all agents happy, two agents happy, and no agents happy. The probabilities in the stationary distribution are the highest for all agents happy, followed by two agents happy, and then no agents happy has the lowest. Thus, the population is most likely to end up segregated into distinct clusters of types.

Extensions

A possible way of extending the model would be creating a function to calculate the probability of a failure for any given swap; this could be done to imitate real life considerations such as socio-economic neighbourhood / household status. This would make the model more realistic as not every household would have the same probabilities of agreeing to a given swap, especially if it does not benefit them. This could be done by assigning positions in the state string different values to represent the costs of living in that space and some value (perhaps representing average socio-economic status) to each agent type, and then designing a function to calculate swapping probability using these as parameters.

Additionally, perhaps neighbours could be redefined to be within x amount of spaces instead of adjacent. Being able to adjust the proximity defining neighbours could provide a more realistic portrayal of a neighbourhood and isolation.

After doing this exercise, it can be concluded that as long as agents have some non-negative desire and ability to be near other agents of the same kind, segregation is inevitable in the long run.