Covid Simulation

code: [https://github.com/adithyarganesh/CSC555_Covid_Simulation (https://github.com/adithyarganesh/CSC555_Covid_Simulation)](https://github.com/adithyarganesh/CSC555_Covid_Simulation)

Created below is an agent based simulation modelling of how a virus can spread in a society during a pandemic. The simulation is created using the **MESA** library to simulate houses, grocery shops, parks and quarantine centers.

Assumptions

1. An agent transitions stages only when at home.
2. The probability value for getting affected increases based on how many infected individuals are in the vicinity. This value is assumed and calculated as `num infected/ total` as I felt a constant value doesn't mimic reality.
3. Beds are cleared in the quarantine center at the end of each day.
4. Users are kept in a common grid location without defining 250 houses for simplicity.
5. Social distancing has been added on top of quarantine.

Source Code

The main libraries used for this simulation are MESA for simulation, Matplotlib for graphs, math and numpy

In [1]:
```
from mesa import Agent, Model
from mesa.time import RandomActivation
from mesa.space import MultiGrid
import math
import numpy as np
import matplotlib.pyplot as plt
```

Covid Simulation Model Class

Firstly, we create a model class where we define the elements needed in a society - Houses, Stores, Parks, Quarantine Centers, Social Distancing norms etc.

The **init** function initializes the model with a grid, predefined slots for areas and finally creates agents with a percent of infected agents. The predefined slots alloted are (0, 0) and (1, 0) for parks, next 5 for grocery and the last one for a common house.

Note: Instead of defining separate houses with individuals, I am leveraging the unique id given to each individual and grouping them by 4.

The **get_individuals** function takes all the allocated grid positions for the agent to be in and returns the number of infected agents in the place along with the total number of agents.

the **clear_beds** function keeps track of all the beds that can be cleared at the end of each day if the agent in the quarantine center is either cured or diseased. It also transitions all the agents at home to the next possible stage based on a list of probabilities.

the **get_infections**, **get_deaths** and the **get_cures** functions return the corresponding statistics from each day.

In [2]:
```
class CovidSimulationModel(Model):

    def __init__(self, num=1000, probability=0.5, dimension=8, quarantine=False,
```

```python
        self.num_agents = num
        self.grid = MultiGrid(dimension, 1, True)
        self.schedule = RandomActivation(self)
        self.healing_space = set()
        self.places = [(i, 0) for i in range(dimension)]
        self.quarantine = quarantine
        self.social_distance = social_distance
        for i in range(self.num_agents):
            agent = VirusAgent(i, self)
            self.schedule.add(agent)
            self.grid.place_agent(agent, (dimension-1, 0))
            infected = np.random.choice([0,1], p=[1-probability,probability])
            if infected == 1:
                agent.state = 1

    def get_individuals(self, pos):
        count = 0
        total = 0
        for temp in self.grid.get_cell_list_contents(pos):
            if temp.state > 0:
                count += 1
            if temp.state > -2:
                total += 1
        return count, total

    def clear_beds(self):
        for agent in self.schedule.agents:
            if agent.pos == (7,0):
                agent.transition()
            if agent.unique_id in self.healing_space and agent.state < 0:
                self.healing_space.remove(agent.unique_id)

    def get_infections(self):
        inf = 0
        for agent in self.schedule.agents:
            if agent.state >= 1:
                inf += 1
        return inf

    def get_deaths(self):
        deaths = 0
        for agent in self.schedule.agents:
            if agent.state == -2:
                deaths += 1
        return deaths

    def get_cures(self):
        cure = 0
        for agent in self.schedule.agents:
            if agent.state == -1:
                cure += 1
        return cure

    def step(self):
        self.schedule.step()
```

Virus Agent Class

The **init** function creates each agent with an a non-infected state and with no immunity or isolation

The `states` are mapped in the following manner:

```
 0   --> Not-infected
 1   --> Asymptomatic
 2   --> Symptomatic
 3   --> Critical
-1   --> Cured
-2   --> Diseased
```

The **move** function is called each day as the agent moves around at home/ goes to the grocery shop or the park with a list of probabilities. Here, if the agent is not currently at home, he gets back home that day.

The **home** function checks if the agent returning home has the virus or not, if he has - the remaining members of the house get infected with the virus.

The **infect** function gets called on the agent when he is out in the park or in a grocery store. Here, based on the number of infected agents present in the same place, a probability value is generated `num. infected/ total agents`. Based on this value, the user may or may not catch the virus.

Each agent undergoes the **transition** stage each day based on the stage the agent is at.

If the model allows for Quarantine, the **quarantine** function gets invoked and based on the number of beds available in the quarantine center and the severity of the virus on the agent, he/she may get quarantined.

In [3]:

```python
class VirusAgent(Agent):

    def __init__(self, unique_id, model):
        super().__init__(unique_id, model)
        self.state = 0
        self.immunity = 0
        self.isolated = 0

    def move(self):
        if self.pos != (7, 0):
            self.home()
        else:
            movement = np.random.choice([1,2,3], p=[0.1, 0.2, 0.7])
            x = 7
            if movement == 1:
                x = np.random.choice([0,1])
            elif movement == 2:
                x = np.random.choice([2,3,4,5,6])
            self.model.grid.place_agent(self, (x, 0))
            if self.state == 0 and x != 7:
                self.infect()

    def home(self):
        self.model.grid.place_agent(self, (7, 0))
        if self.state > 0:
            family = [elem for elem in range(math.floor(self.unique_id/4)*4, mat
            probability = 0.1 if self.model.social_distance else 0.5
            for member in family:
                if self.model.schedule.agents[member].state == 0 and self.model.
                    self.model.schedule.agents[member].state = np.random.choice(

    def infect(self):
        n, total = self.model.get_individuals(self.pos)
        probability = 0 if n == 0 else 0.1 if self.model.social_distance else n/
        if self.state == 0:
            self.state = np.random.choice([0,1], p=[1-probability,probability])

    def transition(self):
        if self.state == 1:
```

```python
            self.state = np.random.choice([1,2], p=[0.75, 0.25])
        elif self.state == 2:
            self.state = np.random.choice([2, 3, -1], p=[0.75, 0.10, 0.15])
        elif self.state == 3:
            self.state = np.random.choice([3, -1, -2], p=[0.75, 0.20, 0.05])
        if self.state == -1:
            self.immunity = 1
            self.isolated = 0

    def quarantine(self):
        if self.state == 2:
            self.isolated = np.random.choice([0,1], p=[0.80, 0.20])
        elif self.state == 3:
            self.isolated = 1
        if self.isolated == 1:
            self.model.healing_space.add(self.unique_id)
            self.model.grid.place_agent(self, (7, 0))

    def step(self):
        if self.unique_id not in self.model.healing_space:
            self.move()
        if len(self.model.healing_space)<100 and self.model.quarantine:
            self.quarantine()

```

The **simulate** function that tracks the metrics for each model run on simulation and plots graphs on several parameters.

In [4]:

```python
def simulate(model):
    days = 0
    infections, cures, deaths, beds = [], [], [],[]

    while(model.get_infections() != 0):
        infections.append(model.get_infections())
        deaths.append(model.get_deaths())
        cures.append(model.get_cures())
        beds.append(len(model.healing_space))
        model.step()
        if days > 1:
            model.clear_beds()
        days += 1


    print("Days for survival", days)
    print("Casualties", model.get_deaths())
    print("Cures", model.get_cures())
    print("Max Quarantine Beds used", max(beds))

    plt.plot(infections)
    plt.plot(deaths)
    plt.plot(cures)
    plt.plot(beds)
    plt.xlabel('Days')
    plt.ylabel('Population')
    plt.gca().legend(('infections','deaths', 'cures', 'beds'))
    plt.show()

```

If the quarantine parameter in the model is set to `True` , the simulation for **Adding Quarantine Centers** task is implemented. If both quarantine and social distancing parameters in the model is set to `True` , the **Enforcing Social Distancing Norms** task is implemented. If both arent mentioned, **Basic implementation** task runs.

Basic Simulation

1. When does the simulation stabilize?

   It can be seen that the simulation stabilizes when the cures, infections and death become a flat line and when there are no infected agents in space. We notice that on average we get the following values

Probability	Days	Max Infections	Cures	Deaths
0.1	59	822	918	82
0.25	54	879	923	77
0.5	57	928	919	81

   ```
           Above table has been generated after taking the mean of the values a
   fter 30 such simulations
   ```

2. Time taken to stabilize

   We see that as the percent of initially infected people increases, the number of days before the curve stabilizes tends to decrease as each day the agent goes through the transformation. And if lots of agents are already affected by the virus, they may undergo the transformation to the final state of diseased/ cured sooner. Corresponding values have been tablulated above.

3. When was the infection at its peak?

   When the initial infection was less, the peak new cases was between days 5 and 10. However, when the number of initially infected agents increased, we notice that the peak new cases in a day was hit much sooner. We also notice that the peak gets reached in 3 to 4 days as the percent of initial infections increases.

4. Deaths by the virus

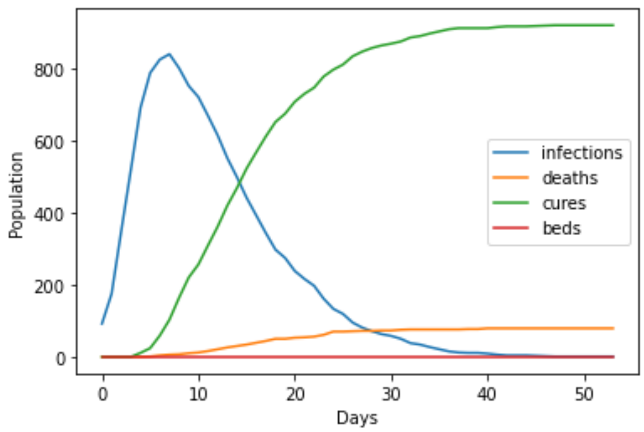
   Since there are only 7 places the agents go to, there is a good chance that he/she gets affected. This makes the chance of death high. From mltiple runs of the simulations, we notice that on average there is a death rate of about 0.08 percent as mentioned in the table above.

5. Graphs

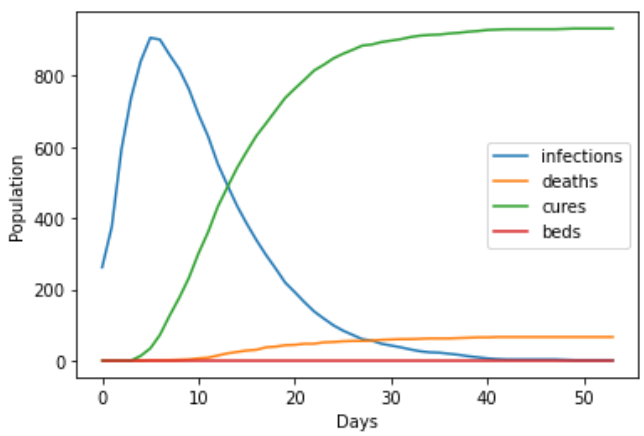
   Please find the graphs for the probability values 0.1, 0.25 and 0.5 along with the metrics below the description

In [5]:
```
simulate(CovidSimulationModel(probability=0.1))
simulate(CovidSimulationModel(probability=0.25))
simulate(CovidSimulationModel(probability=0.50))
```

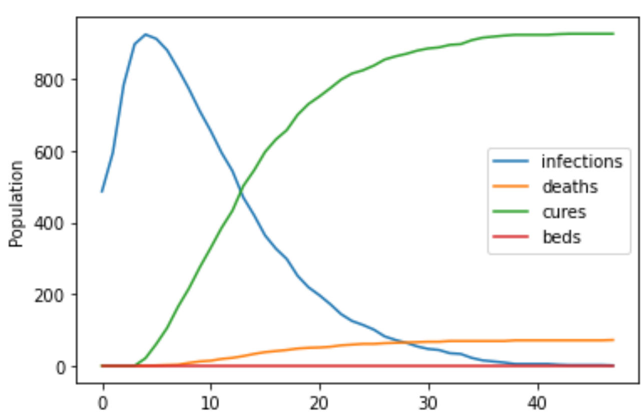
```
Days for survival 54
Casualties 80
Cures 919
Max Quarantine Beds used 0
```




```
Days for survival 54
Casualties 66
Cures 934
Max Quarantine Beds used 0
```




```
Days for survival 48
Casualties 72
Cures 928
Max Quarantine Beds used 0
```



Adding Quarantine Centers

1. When does the simulation stabilize?

   It can be seen that the simulation stabilizes when the cures, infections and death become a flat line and when there are no infected agents in space. We notice that on average we get the following values

Probability	Days	Max Infections	Cures	Deaths	Max Beds
0.1	51	809	919	81	86
0.25	50	874	921	79	86
0.5	49	923	921	79	87

   ```
           Above table has been generated after taking the mean of the values a
   fter 30 such simulations
   ```

2. Time taken to stabilize

   Just like the previous case, we see that as the percent of infected people increases, the time to stabilize the virus decreases. Since we are working with a smaller dataset, we donot see a drastic decrease in the number of days. However, we can conclude that there is an appreciable increase in time as the initial infected percentage decreases. Values are mentioned in the table above.

3. When was the infection at its peak?

   With the quarentine in place, we actually notice that on average for an initial infected value of 0.1, the peak is reached only between 10 and 15 days and with an increase in this value to 0.5 the peak is reached between 5 and 10 days. Max infections has also been tabulated above.

4. Deaths by the virus

   We notice that the deaths are not affected by the addition of quarantine. It hovers around 0.08 irrespective of the initial infected percentage.

5. Graphs

   Please find the graphs for the probability values 0.1, 0.25 and 0.5 along with the metrics below the description

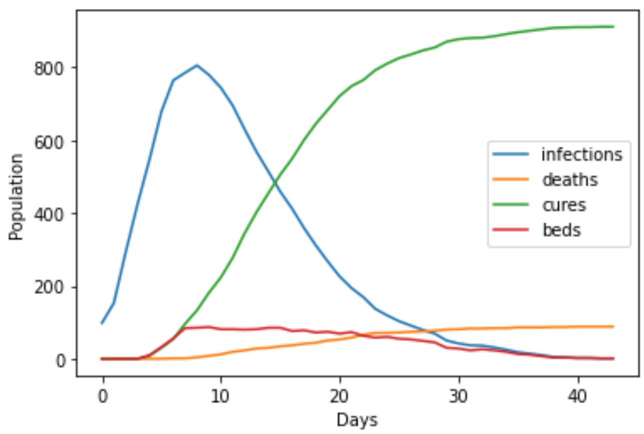
6. Quarantine center fill-up

   The Quarantine center doesnt seem to fill up for all the cases as the probability of an agent being in a critical stage is low. But we notice a steady increase in the number of occupied beds in the quarantine center to slowly increase as the number of initial infected agents increases.

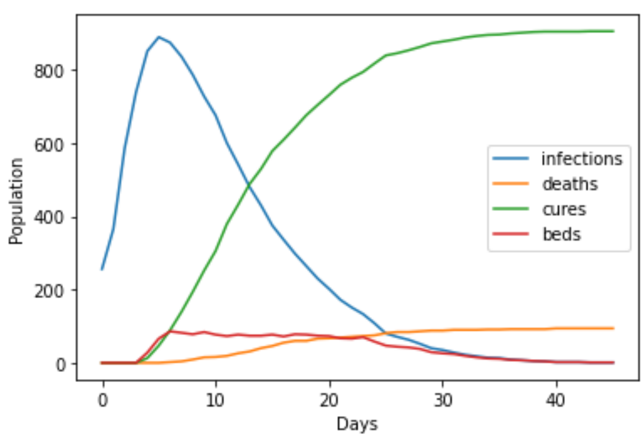
In [6]:

```
simulate(CovidSimulationModel(probability=0.1, quarantine=True))
simulate(CovidSimulationModel(probability=0.25, quarantine=True))
simulate(CovidSimulationModel(probability=0.50, quarantine=True))
```

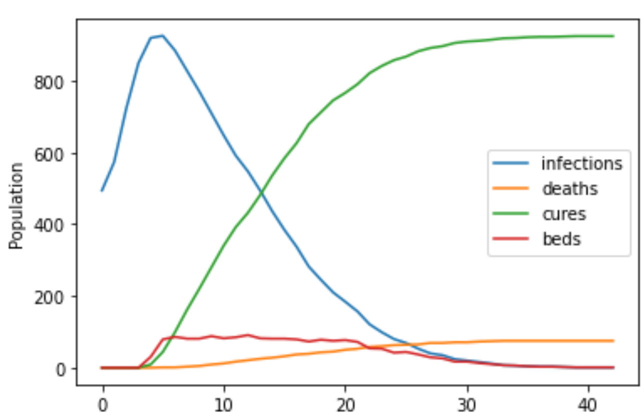
```
Days for survival 44
Casualties 88
Cures 912
Max Quarantine Beds used 87
```




```
Days for survival 46
Casualties 94
Cures 906
Max Quarantine Beds used 86
```




```
Days for survival 43
Casualties 76
Cures 924
Max Quarantine Beds used 91
```



Enforcing Social Distancing Norms

1. When does the simulation stabilize?

   It can be seen that the simulation stabilizes when the cures, infections and death become a flat line and when there are no infected agents in space. We notice that on average we get the following values

Probability	Days	Max Infections	Cures	Deaths	Max Beds
0.1	188	273	912	88	77
0.25	178	371	915	85	83
0.5	167	578	915	85	85

   ```
           Above table has been generated after taking the mean of the values a
   fter 30 such simulations
   ```

2. Time taken to stabilize

   Unlike the previous cases, we see that the stabilization time is much higher. We could thus say that social distancing may indeed help reach a virus free world but it does so by flattenning the peak across multiple days.

3. When was the infection at its peak?

   With the quarentine and social distancing in place, we see that on average for an initial infected value of 0.1, the peak is reached between 10 and 15 days and with an increase in this value to 0.5 the peak is reached between 5 and 10 days just like the previous situation with just quarantine. This is because of our tiny population and regions. This value is very low in comparison to other scenarios.

4. Deaths by the virus

   We notice that the deaths are also not affected by social distancing as the average number of fatalties is aroung 88 for initial probability of 0.1 and it is around the same with a higher initial infected percentage. All values are tabulated above.

5. Graphs

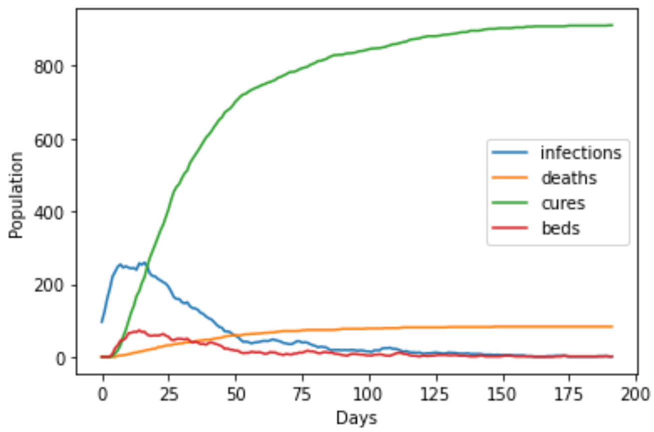
   Please find the graphs for the probability values 0.1, 0.25 and 0.5 along with the metrics below the description

6. Improvements in the situations?

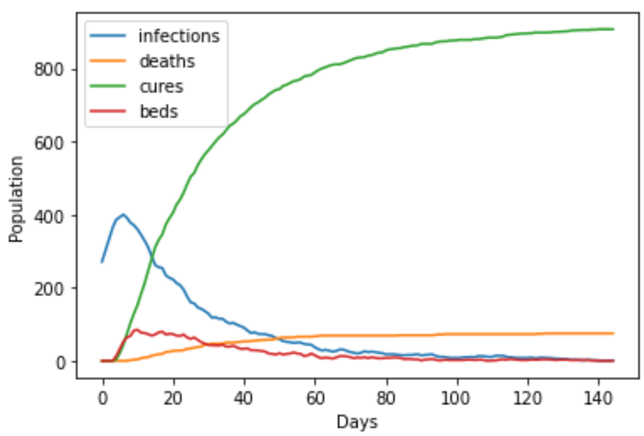
   We could argue that with the addition of social distancing, we notice a much lower peak but a broader graph. This is mainly because we were able to prevent overcrowded quarantine centers with agents rushing to the quarantine centers but instead spreads the time at which each agent is affected so that resources can be efficiently managed.

In [7]:
```
simulate(CovidSimulationModel(probability=0.1, quarantine=True, social_distance=
simulate(CovidSimulationModel(probability=0.25, quarantine=True, social_distance
simulate(CovidSimulationModel(probability=0.50, quarantine=True, social_distance
```

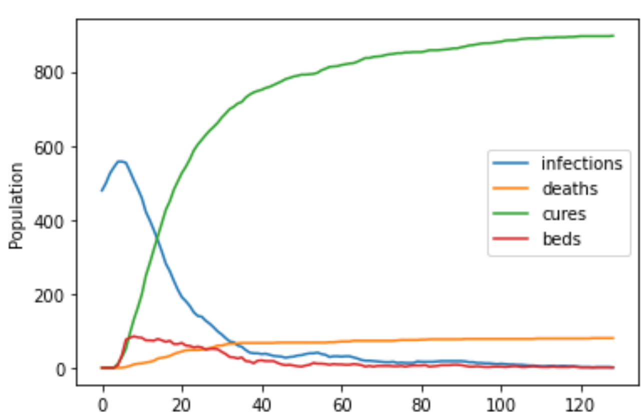
```
Days for survival 192
Casualties 83
Cures 912
Max Quarantine Beds used 73
```




```
Days for survival 145
Casualties 75
Cures 908
Max Quarantine Beds used 85
```




```
Days for survival 129
Casualties 80
Cures 898
Max Quarantine Beds used 85
```



Conclusions

Interesting findings

If one looks at the three tables generated for the 3 situatuations (Basic, Quarantine and Social Norms) we can justify that a similar number of agents reach a cured/ fatal stage but there is a huge difference in time taken to reach the final state. There is lesser unpredictability in the Graph with social distancing incorporated as the slope is much lesser than that determined in basic simulation and a broader graph helps the government manage resources.

From Dr. Fauci's comments that "social distancing may decrease deaths in the country", we notice that that is not the case in our simulation. The reason for that is mainly because we start with about a 100 agents infected and we place on average a minimum of 30 of them in 7 areas. This makes a large number of asymptomatic individuals present in these hotspots negating the affect of quarantine and social distancing. When we change the probability of symptomatic agents to quarantine at home, the new generated graphs showed no decrease in daily deaths as they just happened at a later stage.

Impacts of quarantine and Social distancing

We could make a case that quarantine and social distancing definitely helps. In our simulations, since we have just 7 areas the agent may go to and since we have simplified the case of when he/she might get affected, we do not see a big difference in the time taken to reach a COVID free state. However, with social distancing in place as lesser people catch the virus sooner, there is a safe and efficient way in which the government could manage the situation.

Other things to note about quarantine and social distancing is that the beds occupied are filled more gradually when social distancing is added as tabulated in the tables above.

2 influential factors in spreading the virus

1. The population of the region
2. How strictly the rules are followed by the public and better awareness programmes

Changes to society

1. Minimal outdoor movement
2. Increased virtual events so that there as lesser crowds
3. More stricter social distancing norms

Ref: https://mesa.readthedocs.io/en/master/tutorials/intro_tutorial.html (https://mesa.readthedocs.io/en/master/tutorials/intro_tutorial.html)