NEUROECONOMICS

Submission by Shauryas

CONTRIBUTORS

1. Manasvi Nidugala 220613
2. Shruti Dalvi 221040
3. Palak Mishra 210690
4. Apoorva Gupta 210179
5. Gauravi Chandak 210389

INTRODUCTION

We will start by creating a grid world of size NxN and placing random canteens in it. We will then initialize the population of each type of Macpen and assign them a starting position on the grid. We will then create functions for the movement of Macpen and food collection. The Macpen will move towards the nearest canteen and collect food. After collecting enough food, they will reproduce which results in two new Macpen with an equal share of food from the parent Macpen. We will also create functions for the Ghost Gang's daily visit and its reduction of the Macpen's food supply. We will then implement the behavior of the three types of Macpen: Helpful, Ungrateful, and Tit-for-Tat. We will run the simulation for a fixed number of days and record the population of each type of Macpan on each day. Finally, we plotted the population graphs of each type of Macpan for all the days.

OUR UNDERSTANDING OF THE PS

To begin with, let's correlate the world mentioned in the problem set to a model of the game. We'll be mentioning a few points from the problem set and also be adding our understanding of the same.

The game is a two-dimensional grid, and the characters can either move left, right, up or down. The game comprises Macpan (Plural: Macpen) and the Ghost Gang. The Macpen are the main characters, while the ghost gang is an antagonist.

There are three types of Macpan, as mentioned below:

1. Orange (Helpful) - If the Macpan is in the same cell as another Macpan and has excess food, they can give the food to the other individual.

2. Blue(Ungrateful) - The Macpan will not share food at all.
3. Green(Tit for Tat) - The Macpan will share food with the other pacman based on their history with the grid community

ASSUMPTIONS

- ❖ The game (series of events like reproduction and movement) repeats itself after some interval known as an iteration. One day comprises of NUM_ITERATIONS iterations.
- ❖ Arrival of the ghost gang is marked as the end of a day.
- ❖ The ghost gang snatches GHOST_GANG amount of food from every Macpan.
- ❖ The macpen move only one unit at a time (Left, Right, Up or Down).
- ❖ At every iteration, macpan can either take FOOD_CANTEEN food or might fail to if its away from the canteen.
- ❖ This is the reason why the ghost gang cannot appear after every iteration as the net food intake would become negative.
- ❖ Every canteen has an infinite amount of food.
- ❖ The number of canteens remain consistent in the grid at all times.

APPROACH

We will start by creating a grid world of size NxN and placing random canteens in it. We will then initialize the population of each type of Macpen and assign them a starting position on the grid. We will then create functions for the movement of Macpen and food collection. The Macpen will move towards the nearest canteen and collect food. After collecting enough food, they will reproduce which results in two new Macpen with an equal share of food from the parent Macpen. We will also create functions for the Ghost Gang's daily visit and its reduction of the Macpen's food supply. We will then implement the behavior of the three types of Macpen: Helpful, Ungrateful, and Tit-for-Tat. We will run the simulation for a fixed number of days and record the population of each type of Macpan on each day. Finally, we plotted the population graphs of each type of Macpan for all the days.

IMPLEMENTATION AND STRATEGIES

Every iteration comprises of a few basic activities which are elaborated below:

- Collection of food - Every Macpan either collects food or doesn't. The macpan collects FOOD_CANTEEN units of food from a canteen.
- Sharing of food- A Macpan can share food on the basis of the type of Macpan it is.
- Reproduction - If there is enough food for reproduction, a Macpan can reproduce, i.e., it can split into two, with the amount of food being distributed equally amongst the two.

Moving Strategy

The moving strategy implemented in the code is, every Macpan starts by looking for the closest canteen in the same row and column as its coordinates. In case there is no canteen in the respective '+' region, the macpan will choose a random direction to move.

Sharing Food

Over the course of a single iteration, each macpan helps only once, giving only one unit of food in an iteration . The primary goal of this is to prevent the macpen that need food the most from dying.
In the simulation, firstly we have sorted the macpen on the basis of whether or not they are in excess or in need.
The macpan with food less than or equal to the food taken away by the ghost gang (GHOST_GANG) is considered “needy”, and the macpan with food greater than or equal to ghost gang+2 (GHOST_GANG+2) is considered to be in “excess”.”
The sharing of food is only done if the macpen is in excess. Each ‘excess’ macpan can share only one unit of food, and only once in an iteration. We want to make sure that the macpan that is sharing its food must not die once it shares food, which is why we have written the “ +2”. For eg, if the macpan had only 2 food packets, and gives one of the food packets to a needy macpan, then it will only have one food packet at the end of the day which will be ultimately taken by the ghost gang, rendering it with 0 food packets, and thus it dies.
After doing the sorting, we have arranged the needy macpen in ascending order and the macpen in excess in descending order and we have matched them one by one, and sharing of food occurs. The reason why we have done this pair-wise sharing is in order to simplify the sharing process.

Reproduction

We have ourselves set a threshold for the food required for reproduction of a Macpan (due to the fact that it has not been specified by the problem statement, there is randomness in our final data, as the final results depend on the parameters that we supply to the code). If the food that the macpan has is greater than or equal to the reproduction threshold, then it will reproduce and split into 2 new macpen of the same type as the parent macpan, with no previous history with the community grid. The food undergoes integer division, i.e., if the parent food level is n = odd, each offspring gets (n-1)/2 food.

To deal with sharing food among the Tit-for-Tat macpen, their whole sharing history is stored using a hash map with keys 0 and 1, where 0 is for the times a macron didn’t share food during an interaction, while 1 is for the times it did. The values in the dictionary are updated according to the macpan’s history with the grid community. If there are more no. of 1s in the dictionary, then the macpan will be able to receive food from another green(i.e. tit-for-tat) macpen. However, if the no. of 0s are more, then the macpan will not be able to receive any food from the green macpen as its history is negative with the grid community. By default, we assume that Tit-for-Tat macpen are helpful.

If the macpan is reproducing in the same cell in which a canteen exists, then the canteen will be removed from that place, and a new canteen will be generated at a random place in the cell. The reason why we do this is that if the canteen’s location is not changed, then the newly formed macpen will keep getting food from the same canteen and will not move, thus resulting in infinite multiplication of the macpan. Note that the no. of canteens remains constant throughout the entire iteration.

Environmental Setup

We are assuming the game is set on a N x N grid.
Firstly we start by randomly initializing the location of the canteens
Next we initialize the location of all the helpful macpan, again randomly. The same process is repeated for the ungrateful macpen and the tit-for-tat macpen.

Simulation

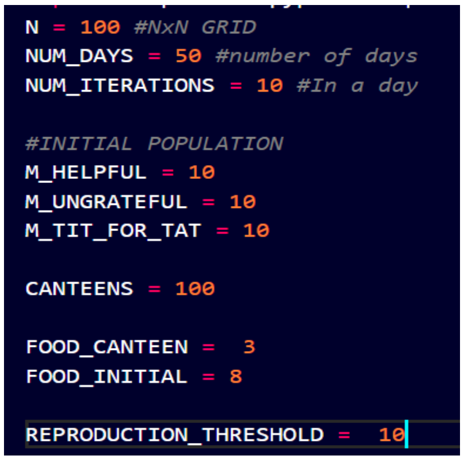
Within the simulation, we firstly check if the canteen exists at the position of the macpan. If it exists, then the macpan’s food is appended by 1. Next, if the reproduction criteria is met, the macpan reproduces and forms 2 macpen with equal amount of food (refer to the reproduction unit for further explanation). After this the macpan moves using the moving strategy described above. If there are more than 2 macpen in the same grid cell, and they meet the sharing criteria, sharing of food occurs(refer to the sharing food unit for further details). At the end of the day, the ghost gang arrives and takes away one food particle from each macpan. If the macpan’s food no. becomes 0, then it dies. This is what happens in each iteration. Every iteration, the food in the canteens is replenished. And each day, the canteen locations are changed and randomised. The positions of canteens are randomly shifted at the end of each day to ensure that the macpen don’t keep on using/moving towards the same canteen. This is done to increase movement in the grid.

INFERENCE AND GRAPHICAL RESULTS

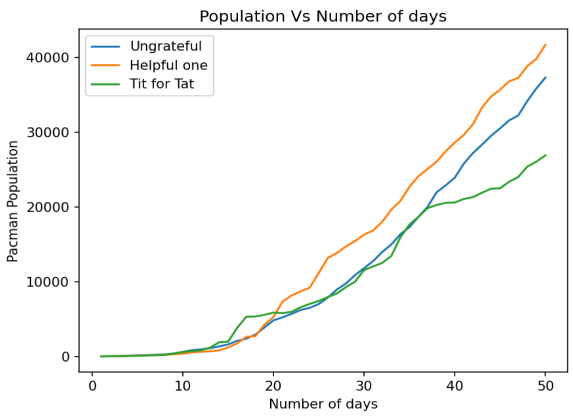
No matter how we change the parameters, the grateful one and the tit-for-tat one behave in a similar manner.

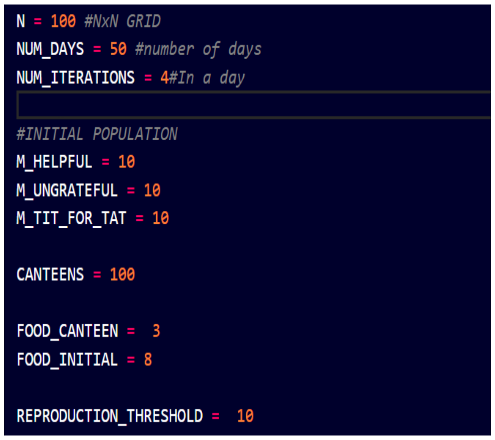
The following changes were made in order to analyze the results:

1) **Change in the number of iterations -**

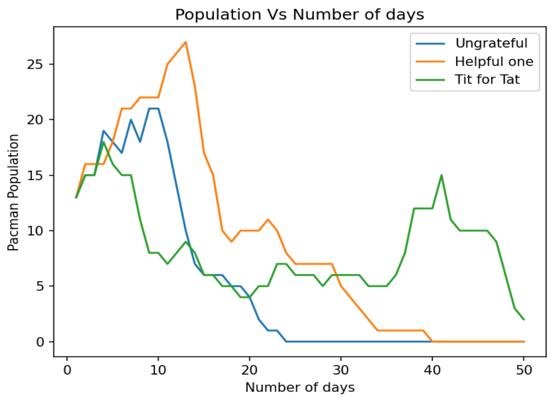


```
N = 100 #NxN GRID
NUM_DAYS = 50 #number of days
NUM_ITERATIONS = 10 #In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 100

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```



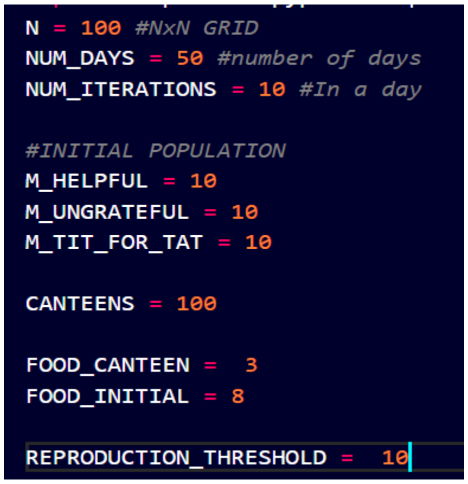


```
N = 100 #NxN GRID
NUM_DAYS = 50 #number of days
NUM_ITERATIONS = 4#In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 100

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```

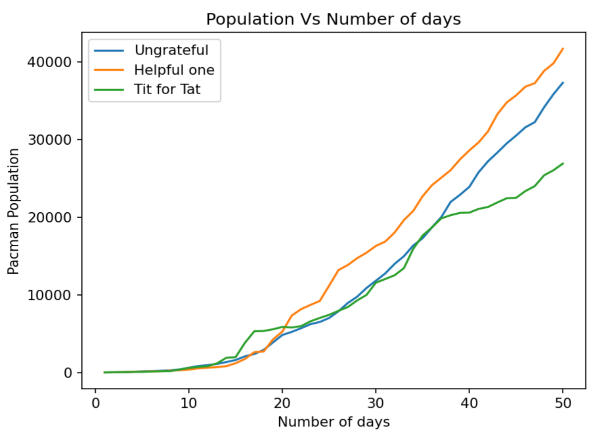


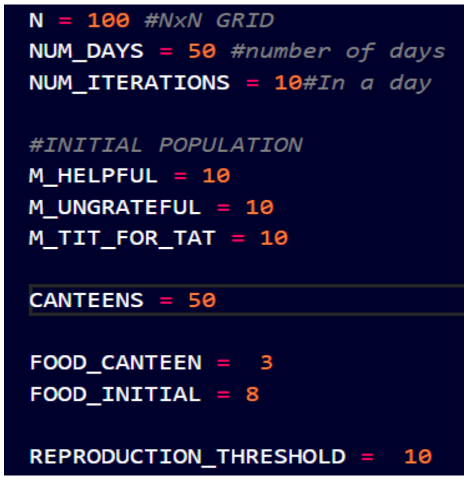
On decreasing the number of iterations, the population starts dying as the arrival of ghost gang overpowers the population growth.

2) On the basis of number of canteens in the area -

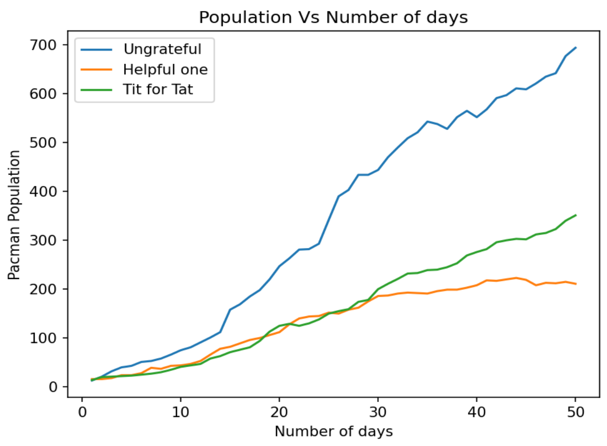


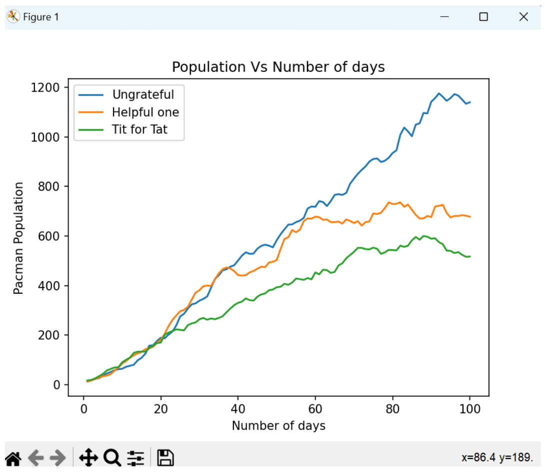
```
N = 100 #NxN GRID
NUM_DAYS = 50 #number of days
NUM_ITERATIONS = 10 #In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 100

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```

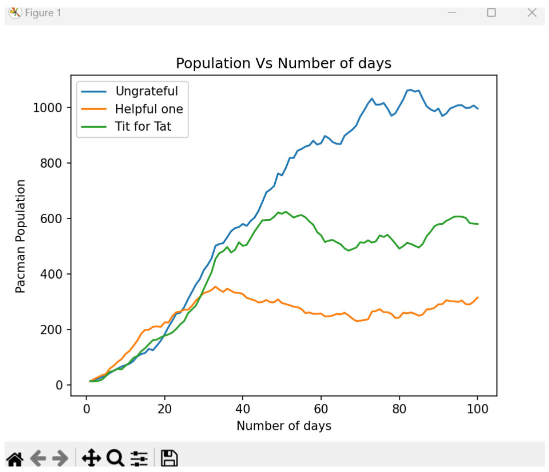




```
N = 100 #NxN GRID
NUM_DAYS = 50 #number of days
NUM_ITERATIONS = 10#In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 50

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```

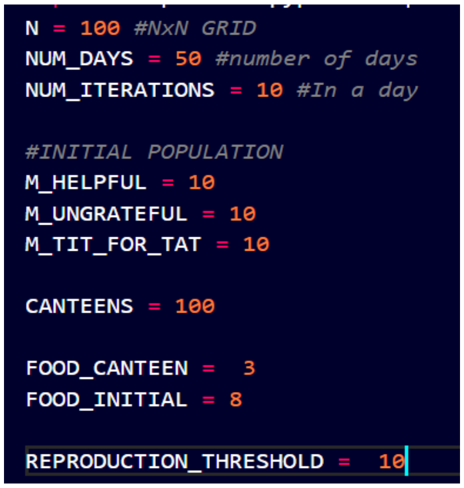




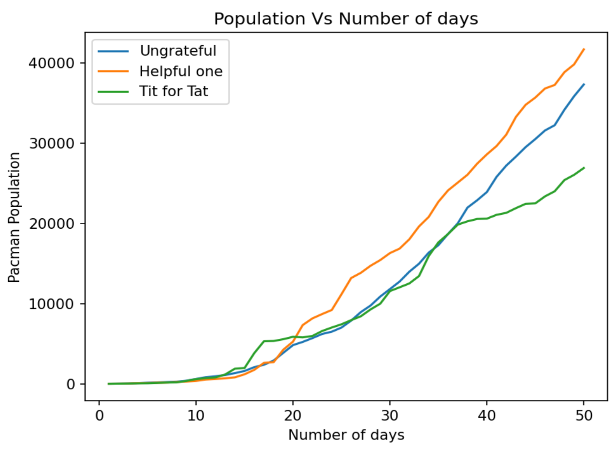


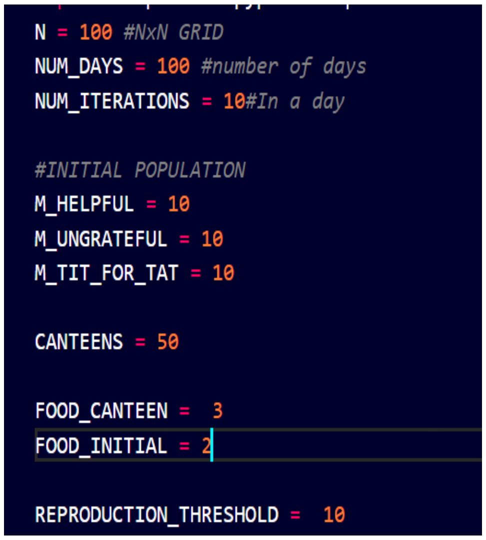
On decreasing the number of canteens the ungrateful one starts winning

3) On decreasing the initial food of the macpen -

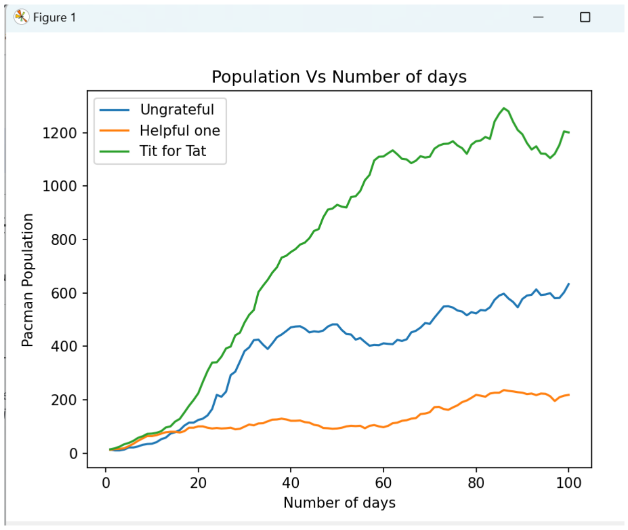


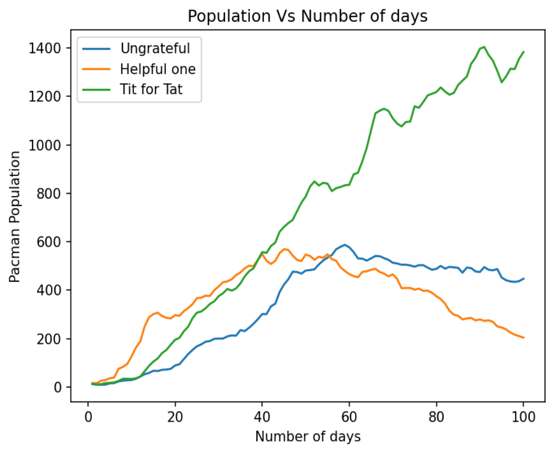
```
N = 100 #NxN GRID
NUM_DAYS = 50 #number of days
NUM_ITERATIONS = 10 #In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 100

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```

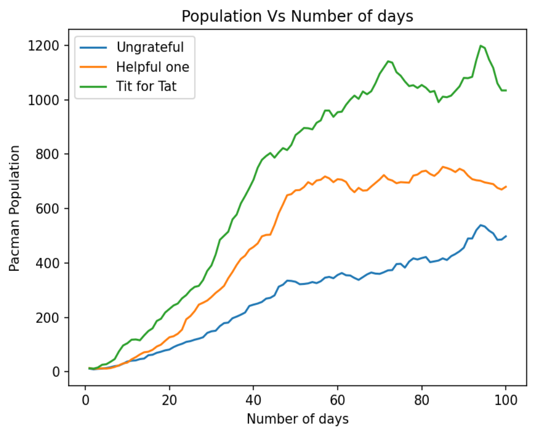


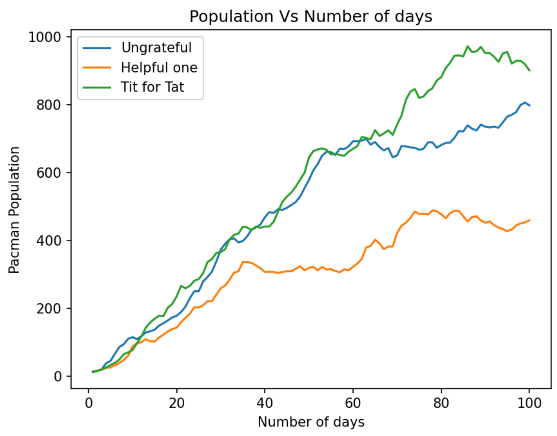


```
N = 100 #NxN GRID
NUM_DAYS = 100 #number of days
NUM_ITERATIONS = 10#In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 50

FOOD_CANTEEN =  3
FOOD_INITIAL = 2

REPRODUCTION_THRESHOLD =  10
```

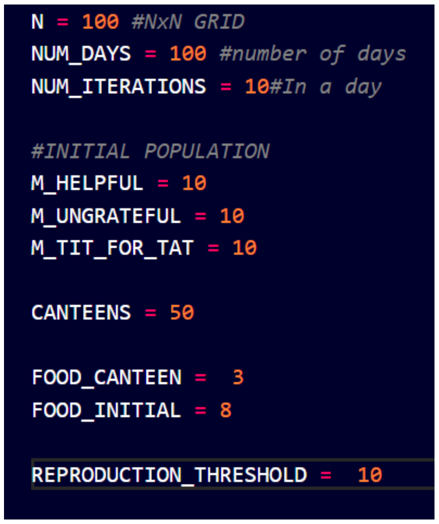


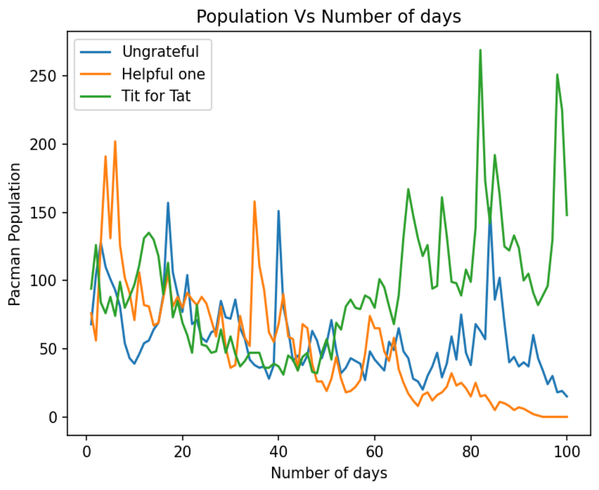






4) On decreasing the threshold of reproduction -


```
N = 100 #NxN GRID
NUM_DAYS = 100 #number of days
NUM_ITERATIONS = 10#In a day

#INITIAL POPULATION
M_HELPFUL = 10
M_UNGRATEFUL = 10
M_TIT_FOR_TAT = 10

CANTEENS = 50

FOOD_CANTEEN =  3
FOOD_INITIAL = 8

REPRODUCTION_THRESHOLD =  10
```



There is a lot of variation post the above change.

RESULTS

We ran the simulation for 1000 days with 1000 macpen in the grid world, out of which 30% were Helpful, 30% were Ungrateful, and 40% were Tit-for-tat. The ghost gang took away 20% of the food every day. The initial food level of each macpan was randomly assigned between 0 and 10. The canteens were randomly placed in the grid world with a fixed amount of food.

From the above graphs, we can observe that the population of all three types of Pacman initially grows but then stabilizes after some time. The most ideal case is if the population of Helpful Macpan is always the highest, followed by Tit-for-tat Macpan and then Ungrateful Macpan. This is because Helpful Macpan shares its food with other Macpan and helps them survive, whereas Ungrateful Macpan does not share food. However, variations in the parameters result into vagaries in the data and the graphs. Certain parameters also result in overlapping of all the three graphs.

BETTER IMPLEMENTATION

Sharing

The sharing strategy that we have implemented is not the most optimal strategy and we are aware of this. This is because of the fact that we have not considered the type of macpen while we have allotted sharing pairs. Expanding on this,

- Orange macpen can share with anyone
- Green macpen can share with all the orange, and nearly half of the green
- Blue macpen do not share with anyone

Considering this, since we have allotted the sharing pairs on the basis of ascending and descending orders, and not considering the type, there are possibilities where this could lead to a negative impact on the community. For eg, a green macpan could be paired with a blue macpan and wouldn't share, and thus its standing in the grid community could potentially go negative and would no longer be able to receive food, and thus would die.

We should try to find a solution that can optimize the pairings such that maximum amount of sharing is possible in between the macpen. However due to the time crunch, we were unable to implement the same.

Moving

Our moving strategy involves checking both the axes of the point where the macpan is in order to find whether or not a canteen is present. If no canteen is present in the “+” region, it will randomly move to a location. While this strategy is a very good one, we are aware that it doesn't account for certain cases, such as diagonal movement of macpen. For example, if a canteen exists diagonal to the macpan and another canteen exists at the very end of the “+” sign, the macpan will choose to go to the 2nd canteen, which is not a very optimal strategy. Due to the scarcity of time, we were unable to code for the same.

CONCLUSION

In conclusion, this project aims to simulate a grid world with Macpen, canteens, and the Ghost gang to analyze the effect of small individual decisions on the population. By implementing the environment in Python, defining the strategies of Macpen, and analyzing the population development over time, we can gain insights into how different types of behavior can affect the survival and growth of a population. The project requires a combination of programming skills, analytical skills, and creativity to come up with effective strategies and visualize the data.

BONUS QUESTION

For the bonus point, we can suggest a similar environment for the above problem statement, which is the simulation of traffic flow in a city. In this environment, we can have different types of vehicles with different driving behaviors such as aggressive, defensive, or neutral. The city can have different road layouts and traffic signals at different intersections. The goal of the simulation is to analyze the impact of individual driving behavior on the overall traffic flow and congestion in the city.

Just like the Pacman world, we can implement a similar simulation where the driving behavior of individual vehicles affects the traffic flow in the entire city. For example, aggressive drivers may cause more accidents and traffic jams, while defensive drivers may cause congestion by driving too slowly. The behaviour of the neutral driver would mostly depend on who it comes across. By analyzing the individual driving behavior and its impact on the overall traffic flow, we can improve the design of road layouts and traffic signals to reduce congestion and improve traffic flow.

We believe that this environment can give similar results to the Pacman world, where the behavior of individual agents affects the overall population development. In both cases, small individual decisions have a significant impact on the entire system, and analyzing these decisions can help us improve the overall performance of the system.