**Airborne Disease Spread Simulation**

**Introduction**

Diseases or virus can be received by simply breathing, this is called Airborne transmission. Airborne disease can spread when carriers of a disease with certain infections talk, cough, sneeze or spew nasal and mouth secretions into the air. There are Viruses that stays in the air, land or any object for some time. When someone breathes in airborne viruses, they stay inside the body. It can also be taken from different things that’s been inhabited by a virus when the virus travelled through air and got into different objects.

The simulation created was done using different libraries in python such as matplotlib and numpy.

**Function**

**def \_\_init\_\_ (self, params)**

**Description**

Shows the usage of polar plot which shows the virus spread from center to outwards. Annotations that shows the needed text to be displayed that gets updated. Member variables that is going to be updated throughout the program.

**Parameters**

Self -> represent the instance

Params -> Dictionary about the virus

**Def initial\_population(self)**

**Description**

Initialized the plot with a population that is showed in the graph .

**Parameters**

Self -> represent the instance

**def spread\_virus(self, i)**

**Description**

Calculates the number of newly infected people using a serial interval.

**Parameters**

Self -> represent the instance

i -> number place holder

**def assign\_sympotms(self)**

**Description**

Calculates the number of mild cases and severe cases in the new wave of people using information stored in params

**Parameters**

Self -> represent the instance

**def update\_status(self)**

**Description**

updates the color of the plot points when a recovery or death occurs

**Parameters**

Self -> represent the instance

**Def update\_text(self)**

**Description**

Updates the set text on each of the text member variables that tracks the day, amount of infections, recoveries, and deaths.

**Parameters**

Self -> represent the instance

**def animate(self)**

**Description**

Shows the plot to update continuously while the program is running

**Parameters**

Self -> represent the instance

**def gen(self)**

**Description**

Generator that is passed to frames for the animation

**Parameters**

Self -> represent the instance

**def one\_by\_one(self, i, thetas, rs, color)**

**Description**

Plotting points individually which is called one by one

**Parameters**

Self -> represent the instance

i -> number holder

thetas -> used for higher repetition in probability

rs -> used to plot points for the data

color -> used for the color of the plots

**def chunks(self, a\_list, n)**

**Description**

Divides a given list into chunks of equal size

**Parameters**

Self -> represent the instance

a\_list -> dictionary

m-> number placeholder

**Simulation**

The simulation that was created used random generator to show the spread of airborne disease and the result of the spread, which is the death, recoveries and infected. The locations of the susceptible possible infected or recovered is displayed in a plot graph.

**Computation**

For the computation that was used in the program most of the computation done was random generator to show the plausible infection spread of airborne disease to see how the spread occurs and the changes happen to the different plots individually. In the software, we also used addition, subtraction and other numpy mathematical properties.