```
//DeductImperfectness.cpp : Defines the entry point for the console application.
//

//#include "stdafx.h"
#include <iostream>
#include <fstream>
#include <stdio.h>
#include <math.h>
#include <vector>
#include <list>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#include <sstream>
#include <string>
#include <cstdio>
#include <time.h>
using namespace std;

int main()
{
# define times 1000

# define dt 1.0

# define Betamu 0.90
# define Betapu 0.90

//# define Betapv 0.833
//# define Betamv 0.773
//# define x 0.10 //0.45

# define mup 0.2933
# define mum  0.211

# define e1 0.550
# define e2 0.650

# define Cv 0.50
# define A 0.50
# define m 0.5

        ostringstream file1;

        file1 << "Betamu= 0.90, Betapu= 0.90, Cv=0.50, A=0.50, x= Betamv,y= Betapv" << ".csv";

        ofstream Data1(file1.str().c_str(), ios_base::out | ios_base::trunc);

        Data1 << "Betamv, Betapv,V,Rs,Rv,P" << endl;



        vector<double>S(70000, 0);

        vector<double>V(70000, 0);

        vector<double>Ipu(70000, 0);

        vector<double>Imu(70000, 0);
```

```
vector<double>Ipv(70000, 0);
vector<double>Imv(70000, 0);
vector<double>Rs(70000, 0);
vector<double>Rv(70000, 0);
vector<double>X(70000, 0);

//double Tau, TTT1;
for (int k = 0; k < 101;k++) {
        double Betapv = k / 100.0;
        for (int p = 0; p < 101;p++) {
                double Betamv = p / 100.0;

S[0] = 0.996;
V[0] = 0.00;
Ipu[0] = 0.001;
Imu[0] = 0.001;
```

```
        Ipv[0] = 0.001;
        Imv[0] = 0.001;

        Rs[0] = 0.0;
        Rv[0] = 0.0;
  X[0]  = 0.1;


                        //double i = 0.0;

                        //while (i < times) {
                                //double xll = x_l[i];

                                //double xtt = tau[i];

        double i = 0.;

        while (i < times) {

  double xx = X[i];

                S[i + dt] = S[i] - xx * S[i]*dt - Betapu * (S[i] - xx * S[i]) * (Ipu[i] + Ipv[i])*dt - Betamu * (S[i]
- xx * S[i]) * (Imu[i] + Imv[i]) * dt;

                V[i + dt] = V[i]+ xx * S[i]*dt - Betapv * (V[i] - e1 * V[i]) * (Ipu[i] + Ipv[i])*dt - Betamv *
(V[i] - e2 * V[i]) * (Imu[i] + Imv[i]) * dt;

                Ipu[i + dt] = Ipu[i]+Betapu * (S[i] - xx * S[i]) * (Ipu[i] + Ipv[i])*dt - mup * Ipu[i] * dt;
                Imu[i + dt] = Imu[i]+Betamu * (S[i] - xx * S[i]) * (Imu[i] + Imv[i])*dt - mum * Imu[i] * dt;

                Ipv[i + dt] = Ipv[i]+Betapv * (V[i] - e1 * V[i]) * (Ipu[i] + Ipv[i])*dt - mup * Ipv[i] * dt;

                Imv[i + dt] = Imv[i]+Betamv * (V[i] - e2 * V[i]) * (Imu[i] + Imv[i])*dt - mum * Imv[i] * dt;

                Rs[i + dt] = Rs[i]+ mup * Ipu[i]*dt + mum * Imu[i] * dt;

                Rv[i + dt] = Rv[i]+ mup * Ipv[i]*dt + mum * Imv[i] * dt;

    X[i + dt] = X[i] + m * X[i] * (1 - X[i]) * (-Cv*V[i] + Ipu[i] + Ipv[i] + Imu[i] + Imv[i] + A) * dt;

     i = i + dt;

    // Data1 << i << "," << S[i] << "," << V[i] << ", " << Ipu[i] + Ipv[i] << "," << Imu[i] + Imv[i] << "," << Rs[i]
+ Rv[i] << "," <<Ipu[i] + Ipv[i]+Imu[i] + Imv[i]<< endl;

     //cout << i << "," << S[i] << "," << V[i] << ", " << Ipu[i] + Ipv[i] << "," << Imu[i] + Imv[i] << "," << Rs[i] +
Rv[i] <<"," <<Ipu[i] + Ipv[i]+Imu[i] + Imv[i]<< endl;

        }
     Data1 << Betamv << "," << Betapv << " ," << V[times] << ", " << Rs[times] <<"," <<Rv[times]<< "," <<
Rs[times]+Rv[times] <<  endl;
     cout << Betamv << "," << Betapv << " ," << V[times] << "," << Rs[times] <<","<<Rv[times]<<"," <<
Rs[times]+Rv[times] << endl;
                }

        }

        Data1.close();
        }
```

Figure Plot Code(Python)

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

Name = '/content/SVIPIMR Case 1 with game 2D Cv e1=e2 model.csv'

cols = ["Cv","e1","P"]

df = pd.read_csv(Name, usecols = cols)
#seismic (For R )
#RdYlGn (For Vaccination)
#PiYG
#PRGn
#BrBG
#PuOr
#RdGy
#RdBu
#RdYlBu
#RdYlGn
#Spectral
#coolwarm
#bwr
pivot_df = df.pivot(index="Cv", columns="e1")

ax = sns.heatmap(pivot_df, cmap = "seismic", square = False, xticklabels =
10, yticklabels = 10, vmin = 0, vmax = 1)

plt.gca().invert_yaxis()

plt.show()
```