```
import random
import numpy as np
import pandas as pd

# Display all
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

# Heading for each column in DataFrame : Staff Strength - Shift Change Interval
column1 = ['2-'+str(i) for i in [1,3,7,14,21]]
column2 = ['4-'+str(i) for i in [1,3,7,14,21]]
column3 = ['6-'+str(i) for i in [1,3,7,14,21]]

# Dataframe which store the outcome at end of day 7
EndDay7 = pd.DataFrame(index=range(0,4),columns=column1+column2+column3)

# Dataframe which store the outcome at end of day 14
EndDay14 = pd.DataFrame(index=range(0,4),columns=column1+column2+column3)

# Dataframe which store the outcome at end of day 21
EndDay21 = pd.DataFrame(index=range(0,4),columns=column1+column2+column3)

# Function to reset the simulation for a new cycle
def restartsim(staff_pool, staffpershift1,staffpershift2, staffpershift3):

    # Staffs are assigned numbers ranging from 0 to staff_pool
    # staff_pool = total number of staffs
    # staffpershift1 = number of staffs in the 1st shift
    # staffpershift2 = number of staffs in the 2nd shift (40% of the 1st shift)
    # staffpershift3 = number of staffs in the 3rd shift (40% of the 1st shift)

    # roster = a data frame to store the roster, with 21 rows, each row showing the staff for
the day
    # The columns in roster dataframe shows the staffs in each shift: shift number - staff slot
number
    # e.g.: Shift1-2 is the 2nd staff slot for the 1st shift
    # e.g.: Shift2-3 is the 3rd staff slot for the 2nd shift
    roster = pd.DataFrame(index=range(0,21),columns=['Shift1-'+str(i) for i in
range(0,staffpershift1)])
    for i in range (0, staffpershift2):
        roster['Shift2-'+str(i)] = 0
    for i in range (0, staffpershift3):
        roster['Shift3-'+str(i)] = 0

    # Create a staff list with 3 columns,
    # 'infected' column shows if the staff is infected: 0 = susceptible; 1 = infected
```

```
    # 'rest' column shows if the staff is resting after working shift (won't be select for roster):
0 = not resting, 1 = resting;
    # 'ref' column is a reference column with value set to 0.
    stafflist = pd.DataFrame(index=range(0,staff_pool),columns=['infected','rest','ref'])
    stafflist.loc[0:staff_pool, 'infected']=0
    stafflist.loc[0:staff_pool, 'rest']=0
    stafflist.loc[0:staff_pool, 'ref']=0
    return stafflist,roster

# Function to fill up the roster with staff number; ensure that the staffs rest for a minimal
period after working a shift
def fillroster1(staff_pool,f,Nday):
    # staff_pool = total number of staffs
    # f = shift change frequency/interval
    # Nday = total number of staff for 1 day
    # num_cycle = number of shift rotation over the 21 days of simulation;
    num_cycle = int(21/f)

    # Special handling for f = 14 as 21 is not a multiple of 14.
    if f == 14:
        temp = random.sample(stafflist[stafflist.loc[:,'rest']==0].index.values.tolist(),k=Nday)
        for j in range(0,f):
            roster.iloc[j] = temp
        stafflist.loc[temp,'rest']=1
        temp = random.sample(stafflist[stafflist.loc[:,'rest']==0].index.values.tolist(),k=Nday)
        for j in range(f,21):
            roster.iloc[j] = temp

    # For other f values, fill up the roster by randomly drawing Nday non-resting staffs from
the staff list
    else:
        for i in range (0,num_cycle):
            temp = random.sample(stafflist[stafflist.loc[:,'rest']==0].index.values.tolist(),k=Nday)
            for j in range(0,f):
                roster.iloc[f*i+j] = temp
            stafflist['rest']=stafflist['rest']-1
            stafflist['rest']=stafflist[['rest','ref']].max(axis=1)
            stafflist.loc[temp,'rest']=1

# Function to model probabilistic transmission via contact within lab
def contact(p,c1,c2,c3, day,staffpershift1, staffpershift2, staffpershift3):
    # p = probability of disease transmission
    # c1,c2,c3 = contact rate for shift 1,2 and 3 respectively
    # day = number of days after the simulation start
    # staffpershift1 = number of staffs in the 1st shift
    # staffpershift2 = number of staffs in the 2nd shift (40% of the 1st shift)
    # staffpershift3 = number of staffs in the 3rd shift (40% of the 1st shift)
```

```
    # To determine which staff is in 1st shift
    staff_in_shift = roster.loc[day,['Shift1-'+str(i) for i in range(0,staffpershift1)]]

    # Form a dataframe with it
    staff_in_shift_df = stafflist.iloc[list(staff_in_shift)]

    # For Shift 1:
    # If there is at least 1 infected staff
    if stafflist['infected'][staff_in_shift].sum()>0:
        # identify staffs who are infected
        staff_I = staff_in_shift_df[staff_in_shift_df['infected']==1]
        # identify staffs who are susceptible
        staff_S = staff_in_shift_df[staff_in_shift_df['infected']==0]

        for j in list(staff_I.index):
            # The number of successful contact per infected staff, k is determined by Poisson
distribution
            contact.n_infected = min(np.random.poisson(lam=p*c1, size=1),[len(staff_in_shift)-
1])

            # Randomly select k staff from those working in the same shift
            staff_infected = random.sample(list(staff_S.index)+list(staff_I[staff_I.index !=
j].index),k=contact.n_infected[0])
            for i in range(0,len(staff_infected)):
                stafflist['infected'][staff_infected[i]]=1

    # For Shift 2: repeating the same infectious process in Shift 1
    staff_in_shift = roster.loc[day,['Shift2-'+str(i) for i in range(0,staffpershift2)]]
    staff_in_shift_df = stafflist.iloc[list(staff_in_shift)]

    if stafflist['infected'][staff_in_shift].sum()>0:
        staff_I = staff_in_shift_df[staff_in_shift_df['infected']==1]
        staff_S = staff_in_shift_df[staff_in_shift_df['infected']==0]

        for j in list(staff_I.index):
            contact.n_infected = min(np.random.poisson(lam=p*c2, size=1),[len(staff_in_shift)-
1])
            staff_infected = random.sample(list(staff_S.index)+list(staff_I[staff_I.index !=
j].index),k=contact.n_infected[0])
            for i in range(0,len(staff_infected)):
                stafflist['infected'][staff_infected[i]]=1

    # For Shift 3: repeating the same infectious process in Shift 1
    staff_in_shift = roster.loc[day,['Shift3-'+str(i) for i in range(0,staffpershift3)]]
    staff_in_shift_df = stafflist.iloc[list(staff_in_shift)]
```

```
    if stafflist['infected'][staff_in_shift].sum()>0:
        staff_I = staff_in_shift_df[staff_in_shift_df['infected']==1]
        staff_S = staff_in_shift_df[staff_in_shift_df['infected']==0]

        for j in list(staff_I.index):
            contact.n_infected = min(np.random.poisson(lam=p*c3, size=1),[len(staff_in_shift)-
1])
            staff_infected = random.sample(list(staff_S.index)+list(staff_I[staff_I.index !=
j].index),k=contact.n_infected[0])
            for i in range(0,len(staff_infected)):
                stafflist['infected'][staff_infected[i]]=1

# Looping through simulation parameters
for staff_strength in [4,6]:
    for f in [1,3,7,14,21]:
        s = 0
        for staffpershift1 in [5,10,20,30]:
            result = pd.DataFrame(index=range(0,100),columns=[str(i) for i in range(0,22)])
            staffpershift2 = int(staffpershift1*0.4)
            staffpershift3 = int(staffpershift1*0.4)
            Nday = staffpershift1 + staffpershift2 + staffpershift3
            staff_pool = staffpershift1*staff_strength

            p = 0.15 # secondary attack rate
            c1 = 0.40*staffpershift1 # number of contact for shift 1
            c2 = 0.40*staffpershift2 # number of contact for shift 2
            c3 = 0.40*staffpershift3 # number of contact for shift 3

            # n = number of cycle for the same simulation param
            for n in range (0, 100):
                # Call the function to reset simulation
                stafflist, roster = restartsim(staff_pool,staffpershift1, staffpershift2,staffpershift3)

                # Call the function to fill the staff roster
                fillroster1(staff_pool,f,Nday)

                # Let the 1st person in the roster be infected;
                stafflist['infected'][roster.iloc[0][0]]=1

                # Run the simulation for 21 days
                for day in range (0,21):
                    contact(p,c1,c2,c3,day,staffpershift1,staffpershift2,staffpershift3)
                    result[str(day)][n]=stafflist['infected'].sum()/staff_pool

            # Storing the median value of infected staff proportion in dataframe at 7,14,21 days
after the simulation start.
            EndDay7[str(staff_strength)+'-'+str(f)][s] = format(result.median()[6],'.2f')
```

```
EndDay14[str(staff_strength)+'-'+str(f)][s] = format(result.median()[13],'.2f')
EndDay21[str(staff_strength)+'-'+str(f)][s] = format(result.median()[20],'.2f')
s += 1
```