# Week 1: Find Clusters of Infected People

**URGENT WARNING**

We have been receiving reports from health facilities that a new, fast-spreading virus has been discovered in the population. To prepare our response, we need to understand the geospatial distribution of those who have been infected. Find out whether there are identifiable clusters of infected individuals and where they are.
</span>

Your goal for this notebook will be to estimate the location of dense geographic clusters of infected people using incoming data from week 1 of the simulated epidemic.

## Imports

In [1]:
```
import cudf
import cuml

import cupy as cp
```

## Load Data

Begin by loading the data you've received about week 1 of the outbreak into a cuDF data frame. The data is located at `'./data/week1.csv'`. For this notebook you will only need the `'lat'`, `'long'`, and `'infected'` columns. Either drop the columns after loading, or use the `cudf.read_csv` named argument `usecols` to provide a list of only the columns you need.

In [ ]:

## Make Data Frame of the Infected

Make a new cuDF data frame `infected_df` that contains only the infected members of the population.

In [ ]:

## Make Grid Coordinates for Infected Locations

Provided for you in the next cell (which you can expand by clicking on the "..." and contract again after executing by clicking on the blue left border of the cell) is the lat/long to OSGB36 grid coordinates converter you used earlier in the workshop. Use this converter to create grid coordinate values stored in `northing` and `easting` columns of the `infected_df` you created in the last step.

```
In [2]: # https://www.ordnancesurvey.co.uk/docs/support/guide-coordinate-syst
ems-great-britain.pdf

def latlong2osgbgrid_cupy(lat, long, input_degrees=True):
    '''
    Converts latitude and longitude (ellipsoidal) coordinates into no
rthing and easting (grid) coordinates, using a Transverse Mercator pr
ojection.

    Inputs:
    lat: latitude coordinate (N)
    long: longitude coordinate (E)
    input_degrees: if True (default), interprets the coordinates as d
egrees; otherwise, interprets coordinates as radians

    Output:
    (northing, easting)
    '''

    if input_degrees:
        lat = lat * cp.pi/180
        long = long * cp.pi/180

    a = 6377563.396
    b = 6356256.909
    e2 = (a**2 - b**2) / a**2

    N0 = -100000 # northing of true origin
    E0 = 400000 # easting of true origin
    F0 = .9996012717 # scale factor on central meridian
    phi0 = 49 * cp.pi / 180 # latitude of true origin
    lambda0 = -2 * cp.pi / 180 # longitude of true origin and central
meridian

    sinlat = cp.sin(lat)
    coslat = cp.cos(lat)
    tanlat = cp.tan(lat)

    latdiff = lat-phi0
    longdiff = long-lambda0

    n = (a-b) / (a+b)
    nu = a * F0 * (1 - e2 * sinlat ** 2) ** -.5
    rho = a * F0 * (1 - e2) * (1 - e2 * sinlat ** 2) ** -1.5
    eta2 = nu / rho - 1
    M = b * F0 * ((1 + n + 5/4 * (n**2 + n**3)) * latdiff -
                  (3*(n+n**2) + 21/8 * n**3) * cp.sin(latdiff) * cp.c
os(lat+phi0) +
                  15/8 * (n**2 + n**3) * cp.sin(2*(latdiff)) * cp.cos
(2*(lat+phi0)) -
                  35/24 * n**3 * cp.sin(3*(latdiff)) * cp.cos(3*(lat+
phi0)))
    I = M + N0
    II = nu/2 * sinlat * coslat
    III = nu/24 * sinlat * coslat ** 3 * (5 - tanlat ** 2 + 9 * eta2)
    IIIA = nu/720 * sinlat * coslat ** 5 * (61-58 * tanlat**2 + tanla
t**4)
    IV = nu * coslat
    V = nu / 6 * coslat**3 * (nu/rho - cp.tan(lat)**2)
    VI = nu / 120 * coslat ** 5 * (5 - 18 * tanlat**2 + tanlat**4 + 1
4 * eta2 - 58 * tanlat**2 * eta2)

    northing = I + II * longdiff**2 + III * longdiff**4 + IIIA * long
diff**6
    easting = E0 + IV * longdiff + V * longdiff**3 + VI * longdiff**5

    return(northing, easting)
```

In [ ]:

*ANSWER OR CODE COPYRIGHTED BY NVIDIA DEEP LEARNING INSTITUTE*

## Find Clusters of Infected People

Use DBSCAN to find clusters of at least 25 infected people where no member is more than 2000m from at least one other cluster member. Create a new column in `infected_df` which contains the cluster to which each infected person belongs.

In [ ]:

*ANSWER OR CODE COPYRIGHTED BY NVIDIA DEEP LEARNING INSTITUTE*

## Find the Centroid of Each Cluster

Use grouping to find the mean `northing` and `easting` values for each cluster identified above.

In [ ]:

*ANSWER OR CODE COPYRIGHTED BY NVIDIA DEEP LEARNING INSTITUTE*

Find the number of people in each cluster by counting the number of appearances of each cluster's label in the column produced by DBSCAN.

In [ ]:

*ANSWER OR CODE COPYRIGHTED BY NVIDIA DEEP LEARNING INSTITUTE*

## Take the Assessment

After completing the work above, visit the *Launch Section* web page that you used to launch this Jupyter Lab. Scroll down below where you launched Jupyter Lab, and answer the question *Week 1 Assessment*. You can view your overall progress in the assessment by visiting the same *Launch Section* page and clicking on the link to the *Progress* page.

There will be additional questions for you to answer after completing the remaining notebooks. On the *Progress* page, if you have successfully answered all the assessment questions, you can click on *Generate Certificate* to receive your certificate in the course.

launch_task_page

## Please Restart the Kernel

```
In [3]: import IPython
        app = IPython.Application.instance()
        app.kernel.do_shutdown(True)

Out[3]: {'status': 'ok', 'restart': True}
```