MATH3205/7202 – Week 5

A collection optimisation problem is defined on a grid as follows:

- A set of vehicles V will collect in a grid of cells for a number of time periods T
- Each vehicle may start in any square in the first time period, then in each subsequent time period it moves to a neighbouring square (up, down, left or right)
- Each grid square has an associated value
- A grid square is considered "collected" if a vehicle visits the square
- No square may be visited more than once
- The objective is to maximise the sum of the values of the grid squares collected.

For example, the 10 by 10 grid below is shown with two vehicles highlighted, each collecting for 8 time periods.

[3, 3, 3, 4, 3, 4, 0, 2, 4, 3]
[4, 0, 2, 1, 2, 2, 0, 1, 1, 4]
[3, 0, 3, 0, 3, 0, 0, 4, 1, 1]
[4, 4, 1, 4, 2, 3, 1, 4, 3, 4]
[4, 1, 1, 0, 0, 0, 1, 3, 0, 3]
[1, 1, 4, 2, 1, 2, 3, 0, 4, 0]
[3, 4, 0, 3, 1, 2, 0, 0, 0, 4]
[0, 3, 4, 4, 1, 1, 2, 3, 0, 3]
[3, 4, 0, 4, 0, 1, 3, 4, 1, 4]
[2, 1, 1, 0, 0, 0, 3, 4, 0, 0]

1. Formulate the problem of generating the optimal collections as an Integer Program.
2. Using the data generated by the code stub Collect.py, implement your IP formulation in python. You may find the function "Neighbours" useful.
3. If you modify your code so that there are 6 vehicles (V = range(6)) then it may take a very long time to run. Why? How could you do better?