Constructing Basestations

27th July 2016 Advance Problem

Ranjith Kumar

Problem description

Four 5G base station towers needs to be installed in a Landscape which is divided as hexagon cells as shown in Fig below, which also contains number of people living in each cell. Need to find four cells to install the 5G towers which can cover maximum number of people combining all four cells, with below conditions

- Only one tower can be placed in a cell
- Each of the four chosen cell should be neighbor to atleast one of the remaining 3 cells.
- All four cells should be connected (like one island)

Refer next slide for some valid combinations

Input range: 1 <= N, M <= 15

Sample input Format for Fig in right

3 4

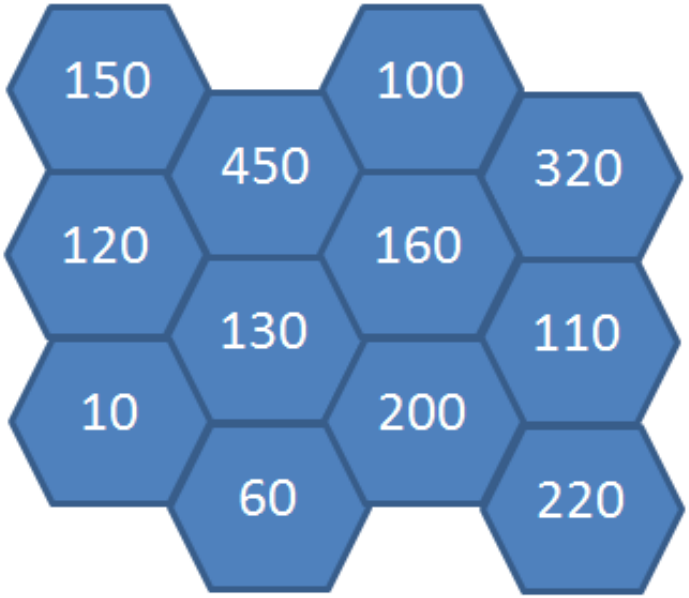
150 450 100 320

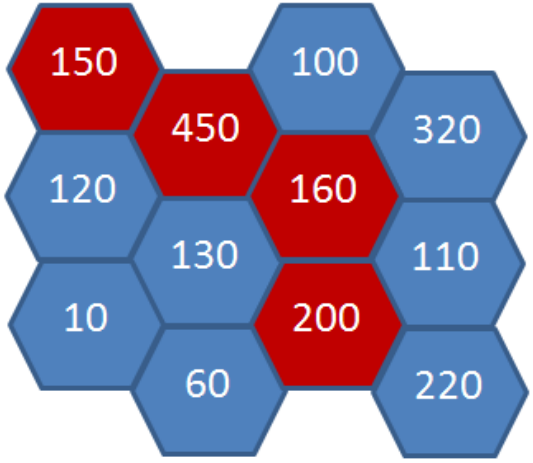
120 130 160 110

10 60 200 220

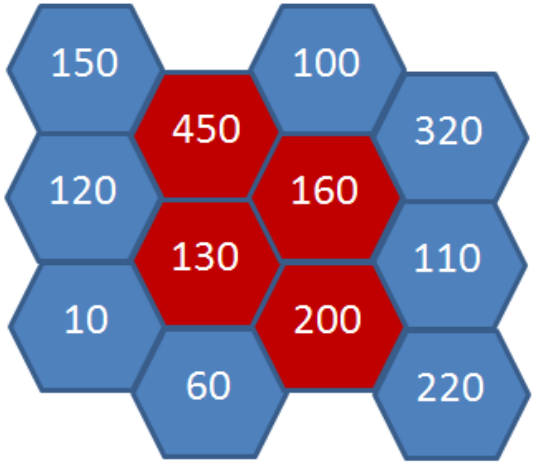
Output

Square of Maximum number of people covered by 4 towers

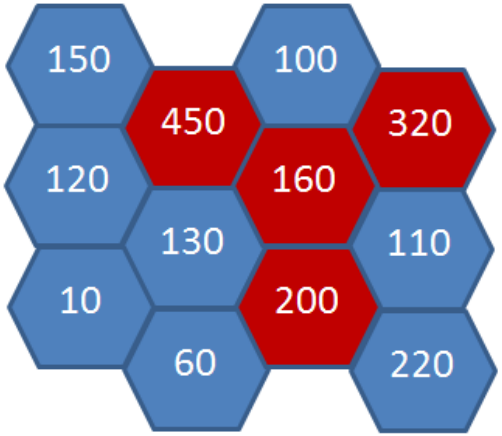




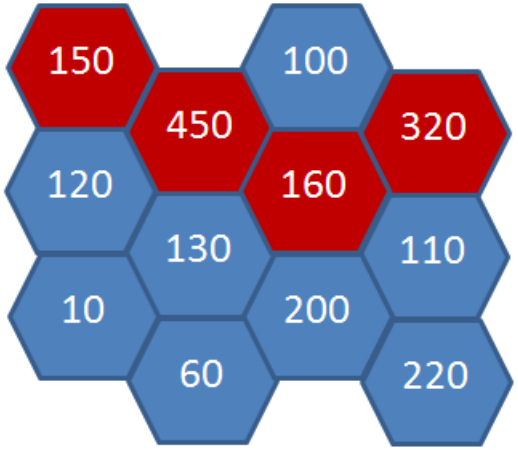
Case 1: sum 960



Case 2: sum 940



Case 3: sum 1130



Case 4: sum 1080

Case 3 has maximum sum, so output is 1130 * 1130 => 1276900

Solutions

- Approach 1:
  - Get logic to find neighbor cells for odd and even cell (w.r.t column)
  - For each cell, do
    - DFS of depth 4
    - combination for remaining number of cells with current cell's neighbor cells only
- Approach 2:
  - Read the input in hexagon format. Get logic to find neighbor cells. In this case logic will be same for both odd and even cell.
  - For each cell, do
    - DFS of depth 4
    - combination for remaining number of cells with current cell's neighbor cells only.
- Approach 3:
  - For each cell give unique number 1, 2, 3, … m*n
  - Generate combination of four numbers from this set and check if these four cells are neighbours.
- Approach 4: (Given by Bhargav Madishetty)
  - Get logic to find neighbor cells for odd and even cell (w.r.t column)
  - For each cell, do
    - DFS of depth 4
    - Calculate Y as shown in figure 3 and also inverted Y.
- Solutions attached for Approach 1 and 2 in Basestation.c, Approach 4 in hexagon.cpp

basestations.c

Hexagon.cpp

input.txt

Common mistakes

- Used same logic to find neighbour cells without differentiating for odd and even cells.
- Some used row index to check even/ odd instead of column index.
- Used only DFS of depth 4 to get the combination, missed the combination which includes more than 2 neighbours of current cell as in case 3 in slide 3.

Similar problem in Sotong (Special Outing)

http://sotong.sec.samsung.net/sotong/cp/cpContestMain.do?contestId=AVYw32R1QwvVldFY