**Play board game**

Alice, Bruce, Cole and David are playing a board game. The board game is a 4x4 checkerboard. Each cell in the checkerboard represents a business, and these four people invested different amounts of money into different businesses. To make a better strategy, they recorded their investment on sheets of papers, each paper contains a 2x2 array like this:

|  |  |
| --- | --- |
| A(Alice) | B(Bruce) |
| C(Cole) | D(David) |

The 16 sheets are given below (From Left to Right, Top to Bottom on the checkerboard):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sheet 1 | | Sheet 2 | | Sheet 3 | | Sheet 4 | |
| 3 | 16 | 9 | 5 | 13 | 3 | 9 | 0 |
| 13 | 15 | 18 | 3 | 13 | 9 | 4 | 7 |
| Sheet 5 | | Sheet 6 | | Sheet 7 | | Sheet 8 | |
| 20 | 10 | 14 | 19 | 14 | 9 | 12 | 7 |
| 16 | 20 | 13 | 3 | 1 | 14 | 3 | 0 |
| Sheet 9 | | Sheet 10 | | Sheet 11 | | Sheet 12 | |
| 1 | 12 | 1 | 4 | 17 | 20 | 20 | 10 |
| 17 | 3 | 0 | 11 | 17 | 14 | 5 | 3 |
| Sheet 13 | | Sheet 14 | | Sheet 15 | | Sheet 16 | |
| 1 | 1 | 2 | 15 | 9 | 3 | 8 | 16 |
| 1 | 1 | 5 | 12 | 9 | 12 | 1 | 4 |

Part 1:

Suppose each cell of the checkerboard has a value, which equals the total amount of money they invested into this business. You are required to write a program based on the following instructions:

* Each cell of the checkerboard is a pointer that points to the corresponding 2x2 array in the sheet.
* Use the knowledge concerned with the **pointer** to calculate the value of each cell.
* Sort the cells in the checkerboard in descending order of the total amount of money they invested into the businesses represented by the corresponding cells. This means cells may point to an array not in the original position, and the first cell (cell(0,0) in 4x4 checkerboard) should point to the array with the largest value.
* Print the sorted values of the checkerboard in a 4x4 shape.
* **Hints**: Although a matrix has rows and columns, actually the addresses of its elements are consecutive in memory. In this way, the matrix can be operated as a 1-dimensional array. (By sorting, it refers to sorting the cells in this 1-dimensional array in descending order of the above-mentioned values.) Here is an example code to explain this view of matrix:

|  |
| --- |
| #include <iostream>  using namespace std;  int main () {  int matrix[2][2] = {{1,2},{3,4}};  int len = sizeof(matrix)/sizeof(int); *//total numbers of elements*  for (int i=0;i<len;i++){  cout << \*(matrix[0]+i) << ' '; *//matrix[0] is the address of first element*  }  cout << endl; *//output 1 2 3 4*  return 0; |

Part 2 (Bonus+20%):

After Part1, cells should point to 2x2 arrays in an order different from the one given above (in the question text), and cell(0,0) should point to the 2x2 array with the largest value. Based on the sorted checkerboard cells, now the penalty mechanism of the board game is triggered. A user first selects 3 cells (among cell(0,0) to cell(3,3)) from the console. For instance, if the user input is “0 1”, it means selecting cell(0,1). The program then checks the cells of the checkerboard **one by one** (from left to right, and from top to bottom). The rules are as follows:

* Step1: If the current cell is a selected cell, rotate **clockwise** the 2x2 array it points to by **one step**. If not, do nothing.
* Step2: Each cell has its neighbors. One direction (North, East, South, West) counts one neighbor, and consequently, at most 4 neighbor-cells will be counted. Some cells could have fewer neighbor-cells because they are located at the edge of the board game. Compare the adjacent parts of the current cell with those of its neighbor cells in the **North-East-South-West order**.
  + Note that if your solution compares cells in a different order, we will regard your solution incorrectly as such a solution will produce different results.
* Step 3: The detailed information for comparison: Suppose the current cell has 4 neighbor-cells. Looking at the 2x2 arrays they respectively point to, you should compare the adjacent elements in pairs (as the dotted box in the picture below), and in the **exact** order mentioned at Step 2. In the pair, if the two values are the same, do nothing. Otherwise, the larger one will get the amount of the smaller one in extra, and the smaller one will become zero. After updating the values in the corresponding 2x2 arrays for one pair, the program proceeds to handle the next pair (i..e, the value will change in processing the sequences of pairs according to the order stated in Step 2).

There is a typo in the original bonus part. Here is the revised graph. Then the format is the same as that in part 1.

图示

描述已自动生成

You should check all the cells. After that, same as what you did in part 1, you should calculate the new value of each cell again, and re-sort the checkerboard (again, sort the cells as if they are in a 1-dimensional array and cells with higher values appear earlier). Finally, output the re-sorted 4x4 checkerboard to the console window.