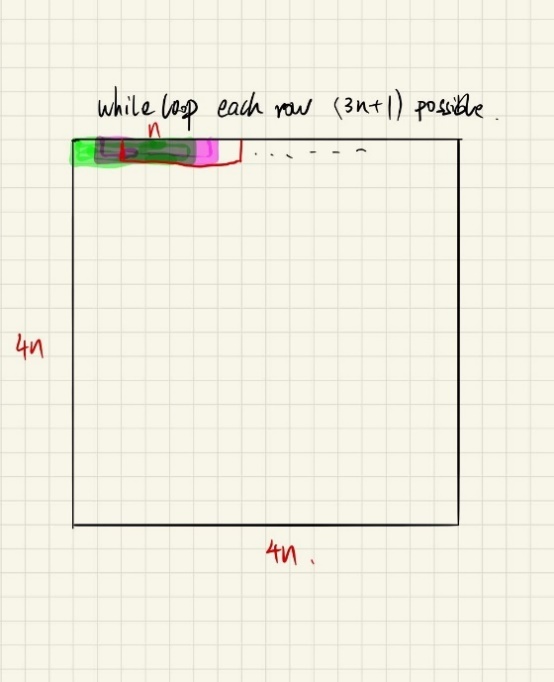
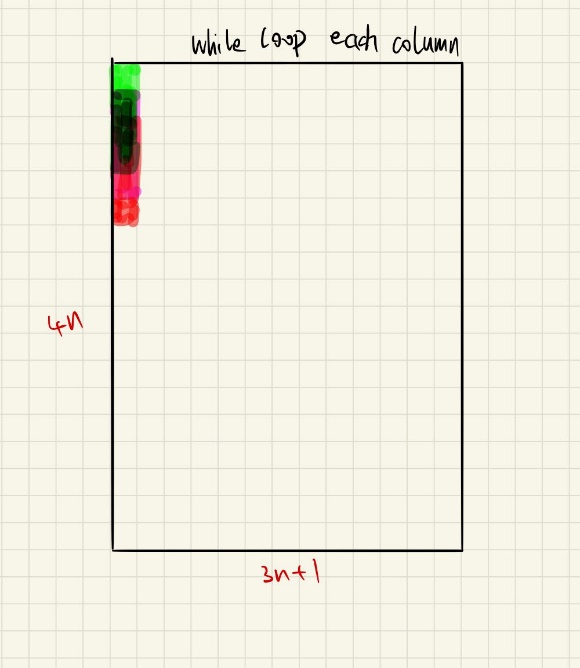
**JIE MEI z5173405**

**Question4**

**First, I use a two-dimensional array to store how many apples there are on each tree. Then we will loop through each row, each row has 3n+1 possibilities, we will use a loop to calculate the total number of apples in n trees. Store each sum into an array of size (3n+1)\*4n, which represents a matrix of (3n+1)\*4n. The time complexity is O(n^2). We will use the similar method above to accumulate the apples of n trees in each column of this new matrix, and we will get a (3n+1)^2 matrix stored in a new array, this array contains All possible combinations. Finally we merge sort this array, we will get a square which contains the largest total number of apples. The time complexity of the above algorithm is O(n^2).**

** **

**Sample code (C file)**

**Generate the origin array all value are 1.**

**Suppose n=4，then we will get the size of an array is 13\*16,like the output.**

#include<stdio.h>

int main(void) {

    //set the value of n is 4

    int n = 4;

    int array[4\*n][4\*n];

    // generate an array

    // all value is 1

    for(int i = 0; i < 4\*n; i++) {

        for(int j = 0; j < 4\*n; j++) {

            array[i][j] = 1;

        }

    }

    //size (3n+1)\*4n

    int A[12\*n\*n+4\*n];

    int sum = 0;

    int origin = n;

    int i = 0;

    int j = 0;

    int z = 0;

    int tmp;

    while(i < 4\*n) {

        sum = 0;

        tmp = j;

        while(j < n) {

            sum = sum + array[i][j];

            j++;

        }

        n++;

        if(n > 4\*origin) {

            i++;

            n = origin;

        }

        A[z] = sum;

        z++;

        if(j > 3\*n+1) j = 0;

        else j = tmp+1;

    }

    // test case

    // print that array like square

    i = 0;

    int k = 1;

    while(i < (3\*n+1)\*4\*n) {

        printf("%d",A[i]);

        if(k == 3\*n+1) {

            k = 0;

            printf("\n");

        }

        k++;

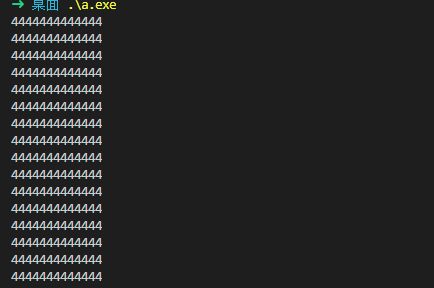
        i++;

    }

    return 0;

}

The program output:



­­