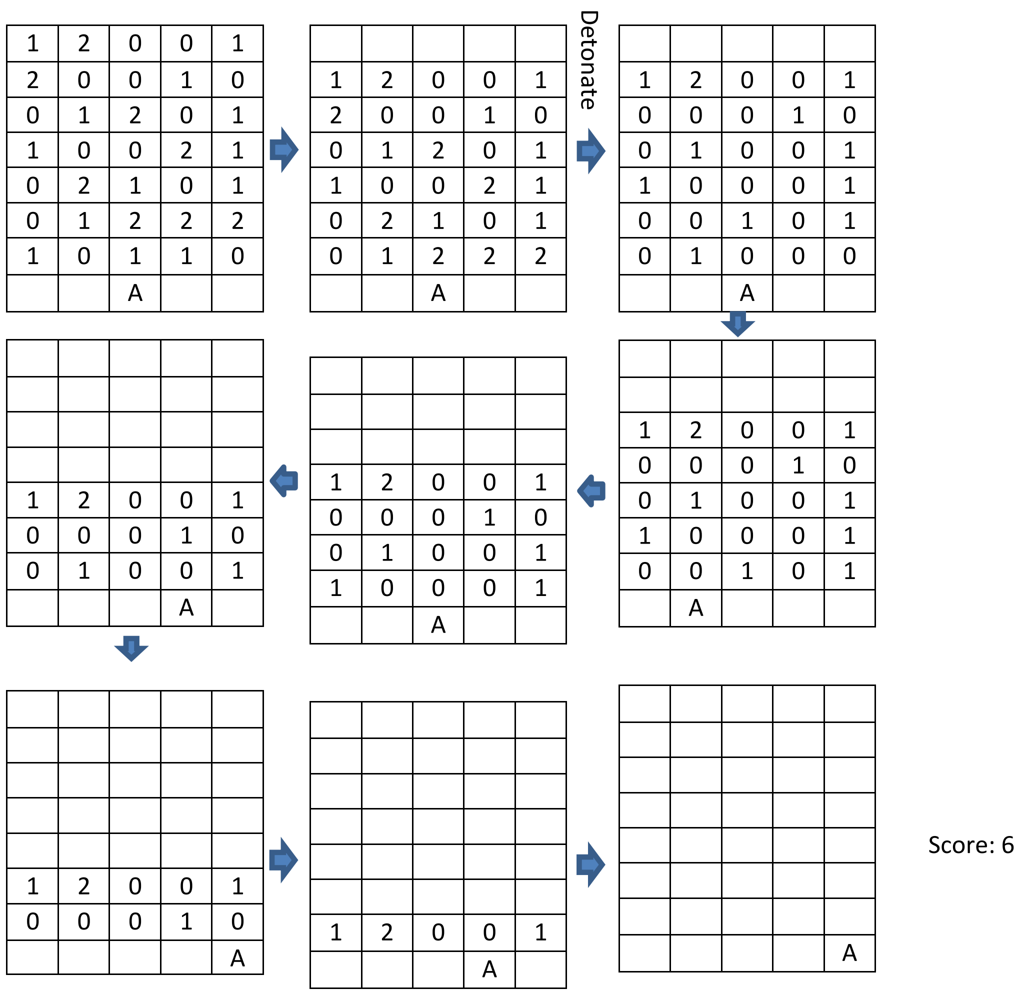
JUNE ADVANCED TEST 2016

QUESTION:

We have a game where an airplane is placed in the middle column of the bottom row. The airplane can move right or left by one step and in every step the row moves down. When the airplane meets ‘1’ (coin) the number of points increase by 1 and when the airplane meets ‘2’ (bomb) the number of points decrease by 1. Whenever the airplane meets the bomb with score 0 the airplane dies and game is over. The user has one detonate option throughout the game where he can detonate all the bombs in the next 5 rows. Find the maximum number of points (coins) that can be collected by the user. Number of rows 1 <= N <= 12. Return -1 if score < 0



1	2	0	0	1
2	0	0	1	0
0	1	2	0	1
1	0	0	2	1
0	2	1	0	1
0	1	2	2	2
1	0	1	1	0
		A		

1	2	0	0	1
2	0	0	1	0
0	1	2	0	1
1	0	0	2	1
0	2	1	0	1
0	1	2	2	2
		A		

Detonate

1	2	0	0	1
0	0	0	1	0
0	1	0	0	1
1	0	0	0	1
0	0	1	0	1
0	1	0	0	0
		A		

1	2	0	0	1
0	0	0	1	0
0	1	0	0	1
1	0	0	0	1
0	0	1	0	1
	A			

1	2	0	0	1
0	0	0	1	0
0	1	0	0	1
1	0	0	0	1
		A		

1	2	0	0	1
0	0	0	1	0
0	1	0	0	1
			A	

1	2	0	0	1
0	0	0	1	0
				A

1	2	0	0	1
			A	

				A

Score: 6

SAMPLE INPUT

The first line denotes the number of test cases and the second line denotes the number of rows. The following lines in every test case have the

```
3
7
1 2 0 0 1
2 0 0 1 0
0 1 2 0 1
1 0 0 2 1
0 2 1 0 1
0 1 2 2 2
1 0 1 1 0
5
1 1 0 0 0
1 2 2 2 1
1 1 2 2 1
2 2 2 1 2
2 2 0 2 0
6
2 2 2 2 2
0 0 0 0 0
0 0 2 0 0
2 0 0 0 2
0 0 0 0 0
1 2 2 2 1
```

SAMPLE OUTPUT

```
 6
 3
-1
```

APPROACHES

APPROACH – 1

The brute force approach would be considering 3 options (move left, move right and current) for all the rows without considering detonate. If we have N rows, the time complexity would be 3^N.

By considering detonate, we can detonate bomb at any step. This means the time complexity would be N*3^N.

Here, the maximum value would of N is 12.

However, if we consider the top down approach, i.e. collecting the coins as the rows move down, we can prune away the tree paths as the score becomes zero.

SAMPLE CODE

```cpp
#include <iostream>
using namespace std;
int a[13][5], b[5][5];

void detonate(int row){
    for (int i = row; i > row - 5; i--){
        for (int j = 0; j < 5; j++){
            b[row - i][j] = 0;
            if (i >= 0 && a[i][j] == 2)
            {
                b[row - i][j] = 2;
                a[i][j] = 0;
            }
        }
    }
}
void unDetonate(int row){
    for (int i = row; i > row - 5; i--){
        for (int j = 0; j < 5; j++)
        {
            if (i >= 0 && b[row - i][j] == 2)
            {
                a[i][j] = 2;
            }
        }
    }
}

void getMaxCoins(int pos, int coins, int n, int &maxCoins){

    if (pos < 0 || pos > 4 || coins < 0) return;

    if (a[n - 1][pos] == 2) coins -= 1;
    else if (a[n - 1][pos] == 1) coins += 1;

    if (n == 1){
        if (coins > maxCoins) maxCoins = coins;
        return;
    }
    else{
        // 3 options
        // move right
        getMaxCoins(pos + 1, coins, n - 1, maxCoins);
        // move left
        getMaxCoins(pos - 1, coins, n - 1, maxCoins);
        // not move
        getMaxCoins(pos, coins, n - 1, maxCoins);
    }
}

int main(){
    int t, n, i, j, k, coins, maxCoins;
    cin >> t;
    for (i = 0; i < t; i++){
        cin >> n;
        maxCoins = -1;
        for (j = 0; j < n; j++){
            for (k = 0; k < 5; k++){
                cin >> a[j][k];
            }
        }
        for (j = 0; j < 5; j++) a[n][j] = 0;
        a[n][2] = 3;
        for (j = n - 1; j > 0 ; j--)
        {
            coins = -1;
            //detonate
            detonate(j);
            getMaxCoins(2, 0, n, coins);
            if (coins > maxCoins) maxCoins = coins;
            unDetonate(j);
            // undetonate

        }
        cout << ((maxCoins < 0) ? -1 : maxCoins) << endl;
    }
}
```