```
#include<iostream>

using namespace std;

struct point
{
        int x;
        int y;
};

point forbidden;

void tile(char ary[][128], int minX, int maxX, int minY, int maxY , int Q);
bool containForbid(int minX, int maxX, int minY, int maxY);
void flip(char ary[][128], int minX, int maxX, int minY, int maxY);
void print(char ary[][128], int maxX, int maxY);


int main()
{
        char board[128][128];
        int order;
        forbidden.x = 0;
        forbidden.y = 0;

        cout << "Please enter the order of the board." << endl;
        cout << "Note must be a power of two. Max = 128" << endl;
        cin >> order;
        // Error checking for max and min order;
        if ((order*order - 1) % 3 == 0 && order <=128)
        {
                for (int i = 0; i < order; i++)
                {
                        for (int j = 0; j < order; j++)
                        {
                                board[i][j] = ' ';
                        }
                }

                cout << "Please enter where to place the forbiddin square." << endl;
                cin >> forbidden.x >> forbidden.y;
                while (forbidden.x >= order || forbidden.y >= order)
                {
                        cout << "Error x and y must be between 0 and " << order - 1 << endl;
                        cout << "Please try again." << endl;
                        cin >> forbidden.x >> forbidden.y;
                }
                //Error checking for forbidden in bounds
                board[forbidden.y][forbidden.x] = 'F';
                tile(board, 0, order - 1, 0, order - 1,0);
                print(board, order - 1, order - 1);
        }
        else
                cout << "board can not be solved." << endl;



        system("pause");
        return 0;
```

```
}

void tile(char ary[][128], int minX, int maxX, int minY, int maxY,int Q)
{
        if (maxX - minX == 1 && maxY-minY == 1)
        {
                if (containForbid(minX, maxX, minY, maxY))
                {
                        for (int i = minY; i <= maxY; i++)
                        {
                                for (int j = minX; j <= maxX; j++)
                                {
                                        if (forbidden.x != j || forbidden.y != i)
                                        {
                                                ary[i][j] = 'L';
                                        }
                                }
                        }
                        return;
                }
                else
                {
                        if (Q == 1)//top left of set
                        {
                                for (int i = minY; i <= maxY; i++)
                                {
                                        for (int j = minX; j <= maxX; j++)
                                        {
                                                if (j !=maxX || i != maxY)
                                                {
                                                        ary[i][j] = 'L';
                                                }
                                        }
                                }
                        }
                        else if (Q == 2)//top right of set
                        {
                                for (int i = minY; i <= maxY; i++)
                                {
                                        for (int j = minX; j <= maxX; j++)
                                        {
                                                if (j != minX || i != maxY)
                                                {
                                                        ary[i][j] = 'L';
                                                }
                                        }
                                }
                        }
                        else if (Q == 3)//bottem left of set
                        {
                                for (int i = minY; i <= maxY; i++)
                                {
                                        for (int j = minX; j <= maxX; j++)
                                        {
                                                if (j != maxX || i != minY)
                                                {
                                                        ary[i][j] = 'L';
                                                }
                                        }
                                }
                        }
```

```
                        else if (Q == 4)//bottem right of set
                        {
                                for (int i = minY; i <= maxY; i++)
                                {
                                        for (int j = minX; j <= maxX; j++)
                                        {
                                                if (j != minX || i != minY)
                                                {
                                                        ary[i][j] = 'L';
                                                }
                                        }
                                }
                        }
                        return;
                }
        }
        else
        {
                tile(ary, minX, (maxX + minX) / 2, minY, (maxY + minY) / 2, 1);
                tile(ary, (maxX + 1 + minX) / 2, maxX, minY, (maxY + minY) / 2, 2);
                tile(ary, minX, (maxX + minX) / 2, (maxY + 1 + minY) / 2, maxY, 3);
                tile(ary, (maxX + 1 + minX) / 2, maxX, (maxY + 1 + minY) / 2, maxY, 4);
                int empty = 0;
                for (int i = minY; i <= maxY; i++)
                {
                        for (int j = minX; j <= maxX; j++)
                        {
                                if (j != minX || i != minY)
                                {
                                        if (ary[i][j] == ' ')
                                        {
                                                empty++;
                                        }
                                }
                        }
                }
                if (empty == 3)
                {
                        for (int i = minY; i <= maxY; i++)
                        {
                                for (int j = minX; j <= maxX; j++)
                                {
                                        if (j != minX || i != minY)
                                        {
                                                if (ary[i][j] == ' ')
                                                {
                                                        ary[i][j] = 'L';
                                                }
                                        }
                                }
                        }
                }
                else if (empty == 4)
                {
                        if (Q == 1)
                        {
                                flip(ary, (maxX + minX + 1) / 2, maxX, (maxY + minY + 1) / 2,
 maxY);
                        }

                        else if (Q == 2)
```

```
                        {
                                flip(ary, minX, (maxX + minX) / 2, (maxY + minY + 1) / 2,
maxY);
                        }
                        else if (Q == 3)
                        {
                                flip(ary, (maxX + minX + 1) / 2, maxX, minY, (maxY + minY) /
2);
                        }
                        else if (Q == 4)
                        {
                                flip(ary, minX, (maxX + minX) / 2, minY, (maxY + minY) / 2);
                        }
                        for (int i = (maxY + minY) / 2; i <= (maxY + minY) / 2 + 1; i++)
                        {
                                for (int j = (maxX + minX) / 2; j <= (maxX + minX) / 2 + 1;
j++)
                                {
                                        if (ary[i][j] == ' ')
                                        {
                                                ary[i][j] = 'L';
                                        }
                                }
                        }
                }
        }


}

bool containForbid(int minX, int maxX, int minY, int maxY)
{
        bool forbid = false;
        for (int i = minY; i <= maxY; i++)
        {
                for (int j = minX; j <= maxX; j++)
                {
                        if (forbidden.x == j && forbidden.y == i)
                        {
                                forbid = true;
                        }
                }
        }
        return forbid;
}

void flip(char ary[][128], int minX, int maxX, int minY, int maxY)
{
        char temp[128][128];

        int u = maxY;
        int v = maxX;
        for (int i = 0; i< 64; i++)
        {
                for (int j = 0; j < 64; j++)
                {
                        temp[i][j] = 0;
                }
        }
        for (int i = 0; u >= minY; i++, u--)
        {
```

```
                v = maxX;
                for (int j = 0; v >= minX; j++, v--)
                {
                        temp[i][j] = ary[u][v];
                }
        }
        u = minY;
        for (int i = 0; u <= maxY; i++, u++)
        {
                v = minX;
                for (int j = 0; v <= maxX; j++, v++)
                {
                        ary[u][v] = temp[i][j];
                }
        }
        return;
}
void print(char ary[][128], int maxX, int maxY)
{
        for (int i = 0; i <= maxY; i++)
        {
                for (int j = 0; j <= maxX; j++)
                {
                        cout << ary[i][j];
                        if (j < maxX)
                        {
                                cout << ",";
                        }
                }
                cout << endl;
        }
        cout << endl;
        return;
}
/*
Output
Please enter the order of the board.
Note must be a power of two. Max = 128
8
Please enter where to place the forbiddin square.
3
4
L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L
L,L,L,F,L,L,L,L
L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L

Press any key to continue . . .

Output 2
Please enter the order of the board.
Note must be a power of two. Max = 128
16
Please enter where to place the forbiddin square.
20
5
Error x and y must be between 0 and 15
```

```
Please try again.
0
0
F,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L
L,L,L,L,L,L,L,L,L,L,L,L,L,L,L,L

Press any key to continue . . .

Output 3
Please enter the order of the board.
Note must be a power of two. Max = 128
3
board can not be solved.
Press any key to continue . . .
*/
```