```
#include <iostream>
#include <fstream>
//labirintus bejarasa
//9: fal
//5: cel
//x, y: kiindulas
//1: ahova lépett
//2: ahova lépett, de zsákutcának bizonyult
//rekurzív implementáció (elhalasztott utasításokkal)
using namespace std;
    ifstream f("matrix.in");
    ofstream g("matrix.out");
void kiiras(int M[][100], int r, int c)
{
    for (int i=1; i<=r; i++)
    {
        for (int j=1; j<=c; j++)
            if (M[i][j]!=1) g<<M[i][j]<<" ";
            else g<<"X"<<" ";
        g<<endl;
    }
}


void step(int M[][100], int r, int c, int x, int y)
{
  if (M[x][y]!=5)
  {
    M[x][y]=1;
    if (x>1 && (M[x-1][y]==0 || M[x-1][y]==5)) step(M, r, c, x-1, y);
    if (y<c && (M[x][y+1]==0 || M[x][y+1]==5)) step(M, r, c, x, y+1);
    if (y>1 && (M[x][y-1]==0 || M[x][y-1]==5)) step(M, r, c, x, y-1);
    if (x<r && (M[x+1][y]==0 || M[x+1][y]==5)) step(M, r, c, x+1, y);
    M[x][y]=2;
  }
  else
  {
    M[x][y]=1;
    kiiras(M, r, c);
    M[x][y]=2;
  }
  return;
}

int main()
{
    cout << "Labirintus" << endl;
    int A[100][100], ssz, osz, x, y;
    f>>ssz>>osz>>x>>y;
    for (int i=1; i<=ssz; i++)
        for (int j=1; j<=osz; j++)
          f>>A[i][j];
    step(A, osz, ssz, x, y);

    f.close();
    g.close();
    return 0;
}
```