J. Два прямоугольника

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Недавно один известный художник-абстракционист произвел на свет новый шедевр — картину «Два черных непересекающихся прямоугольника». Картина представляет собой прямоугольник *m× n*, разбитый на квадраты *1× 1*, некоторые из которых закрашены любимым цветом автора — черным. Федя — не любитель абстрактных картин, однако ему стало интересно, действительно ли на картине изображены два непересекающихся прямоугольника. Помогите ему это узнать. Прямоугольники не пересекаются в том смысле, что они не имеют общих клеток.

Формат ввода

Первая строка входного файла содержит числа *m* и *n* (*1 ≤ m, n ≤ 200*). Следующие *m* строк содержат описание рисунка. Каждая строка содержит ровно *n* символов. Символ «.» обозначает пустой квадрат, а символ «#» — закрашенный.

Формат вывода

Если рисунок можно представить как два непересекающихся прямоугольника, выведите в первой строке «YES», а в следующих *m* строках выведите рисунок в том же виде, в каком он задан во входном файле, заменив квадраты, соответствующие первому прямоугольнику на символ «a», а второму — на символ «b». Если решений несколько, выведите любое.

Если же этого сделать нельзя, выведите в выходной файл «NO».

Пример 1

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 1  #  . | NO |

Пример 2

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 2 2  ..  ## | YES  ..  ab |

Пример 3

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 3  ### | YES  abb |

Пример 4

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 5  ####. | YES  abbb. |

using System;

class Program

{

static void Main()

{

char RESHOTKA = '#';

char current\_char = 'a';

string[] input = Console.ReadLine().Split();

int n = int.Parse(input[0]);

int m = int.Parse(input[1]);

char[,] Row\_feald = new char[n,m];

char[,] Column\_feald = new char[n,m];

char[,] resultArr = new char[n,m];

string result = "NO";

int counter = 0;

for (int i = 0; i < n; i++)

{

string input1 = Console.ReadLine();

for (int j = 0; j < m; j++)

{

Row\_feald[i, j] = input1[j];

Column\_feald[i, j] = input1[j];

if (input1[j] == '#') counter++;

}

}

if (counter == 1 || counter == 0) Console.WriteLine(result);

else

{

char endchar = 'a';

while (current\_char != 'd')

{

int start\_index\_i = -1;

int start\_index\_j = -1;

int len = 0;

for (int i = 0; i < n; i++)

{

int current\_len = 0;

bool flag = true;

for (int j = 0; j < m; j++)

{

if (Row\_feald[i, j] == RESHOTKA)

{

if (start\_index\_i == -1)

{

start\_index\_i = i;

start\_index\_j = j;

}

if (i == start\_index\_i)

{

Row\_feald[i, j] = current\_char;

endchar = current\_char;

len++;

}

if (j < start\_index\_j || (current\_len == 0 && j > start\_index\_j && i != start\_index\_i))

{

j = m;

i = n;

}

else if (current\_len < len && i != start\_index\_i)

{

Row\_feald[i, j] = current\_char;

current\_len++;

}

else if (current\_len >= len)

{

for (int u = start\_index\_j; u < start\_index\_j + current\_len + 1; u++)

{

Row\_feald[i, u] = '#';

}

j = m;

i = n;

}

}

else

{

if (start\_index\_j != -1 && i == start\_index\_i)

{

j = m;

}

if (j <= start\_index\_j + len - 1 && start\_index\_j != -1 && current\_len != 0 &&

i != start\_index\_i)

{

for (int u = start\_index\_j; u < start\_index\_j + current\_len; u++)

{

Row\_feald[i, u] = '#';

}

j = m;

i = n;

}

if (current\_len != 0 && i != start\_index\_i)

{

j = m;

// i = n;

}

}

}

if (current\_len == 0 && len != 0 && i != start\_index\_i)

{

i = n;

}

}

current\_char++;

}

// for (int i = 0; i < n; i++)

// {

// for (int j = 0; j < m; j++)

// {

// Console.Write(Row\_feald[i, j]);

// }

//

// Console.WriteLine();

// }

if (endchar == 'a' || endchar == 'b')

{

result = "YES";

Console.WriteLine(result);

if (endchar == 'a')

{

int This = -1;

int row = -1;

int column = -1;

int max\_in\_row = 0;

for (int i = 0; i < n; i++)

{

int countMatch = 0;

for (int j = 0; j < m; j++)

{

if (Row\_feald[i, j] == 'a') countMatch++;

if (countMatch > max\_in\_row) max\_in\_row = countMatch;

}

}

if (max\_in\_row == 1)

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < m; j++)

{

if (Row\_feald[i, j] == 'a' && ((i + 1 < n && Row\_feald[i+1, j] != 'a') || i + 1 >= n))

{

Console.Write('b');

}

else Console.Write(Row\_feald[i, j]);

}

Console.WriteLine();

}

}

else

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < m; j++)

{

if (Row\_feald[i, j] == 'a' && ((j + 1 < m && Row\_feald[i, j + 1] != 'a') || j + 1 >= m))

{

Console.Write('b');

}

else Console.Write(Row\_feald[i, j]);

}

Console.WriteLine();

}

}

}

else

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < m; j++)

{

Console.Write(Row\_feald[i, j]);

}

Console.WriteLine();

}

}

}

else

{

current\_char = 'a';

endchar = 'a';

while (current\_char != 'd')

{

int start\_index\_i = -1;

int start\_index\_j = -1;

int len = 0;

for (int j = 0; j < m; j++)

{

int current\_len = 0;

bool flag = true;

for (int i = 0; i < n; i++)

{

if (Column\_feald[i, j] == RESHOTKA)

{

if (start\_index\_j == -1)

{

start\_index\_j = j;

start\_index\_i = i;

}

if (j == start\_index\_j)

{

Column\_feald[i, j] = current\_char;

endchar = current\_char;

len++;

}

if (i < start\_index\_i || (current\_len == 0 && i > start\_index\_i && j != start\_index\_j))

{

i = n;

j = m;

}

else if (current\_len < len && j != start\_index\_j)

{

Column\_feald[i, j] = current\_char;

current\_len++;

}

else if (current\_len >= len)

{

for (int u = start\_index\_i; u < start\_index\_i + current\_len +1; u++)

{

Column\_feald[u, i] = '#';

}

i = n;

j = m;

}

}

else

{

if (start\_index\_j != -1 && j == start\_index\_j)

{

i = n;

}

if (i <= start\_index\_i + len - 1 && start\_index\_i != -1 && current\_len != 0 &&

j != start\_index\_j)

{

for (int u = start\_index\_i; u < start\_index\_i + current\_len; u++)

{

Column\_feald[u, i] = '#';

}

}

if (current\_len != 0 && j != start\_index\_j)

{

i = n;

// j = m;

}

}

}

if (current\_len == 0 && len != 0 && j != start\_index\_j)

{

j = m;

}

}

current\_char++;

}

// Console.WriteLine("dfghj");

//

// for (int i = 0; i < n; i++)

// {

// for (int j = 0; j < m; j++)

// {

// Console.Write(Column\_feald[i, j]);

// }

//

// Console.WriteLine();

// }

if (endchar == 'b')

{

Console.WriteLine("YES");

for (int i = 0; i < n; i++)

{

for (int j = 0; j < m; j++)

{

Console.Write(Column\_feald[i, j]);

}

Console.WriteLine();

}

}

else Console.WriteLine("NO");

}

}

}

}