%Validation\_nQueens\_Problem\_Solution

%{

Эта программа предназначена:

a) *для проверки правильности решения задачи распределения n-ферзей на произвольной шахматной доски размера* *n x n,*

b) *для проверки правильности произвольной композиции из k ферзей, случайным образом распределенных на шахматной доске размером* *n x n.*

%}

%{

License: *Attribution-NonCommercial-ShareAlike CC BY-NC-SA*

“Эта лицензия позволяет другим редактировать, адаптировать и развивать вашу работу в некоммерческих целях при условии, что они соблюдают требование и лицензируют свои новые творения на идентичных условиях.”.

%}

%{

Автор и разработчик проекта - Григорян Эрос (*EricGrig*), 2020

Буду рад, если какие-то разделы кода или вся программа в целом будут использованы в научных целях или в образовательных целях. При этом буду признателен, если вы сочтете возможным сослаться на мою публикацию. Это элемент культуры и знак взаимного уважения.

Для коммерческого использования любой части программного кода или всей программы в целом требуется письменное согласие автора.

%}

%{

Этот алгоритм подробно описан в статье:

*Grigoryan E., Linear algorithm for solution n-Queens Completion problem,* [*https://arxiv.org/abs/1912.05935*](https://arxiv.org/abs/1912.05935)

Будет правильно, если вы сначала прочтете эту публикацию, прежде чем приступить к анализу исходного кода. Это сделает описание программы более прозрачным и уменьшит количество возможных вопросов.

%}

%{

*Как подготавливаются исходные данные*?

Обозначим размер стороны шахматной доски через *n*.

Пусть существует одномерный обнуленный массив размера *n*.

Если в *i-ой* строке шахматной доски ферзь находится на позиции *j*, то, соответственно, в *i-ую* ячейку массива данных записывается значение *j*.

%}

%{

*Ввод данных*.

Запишем входные данные в массив *Q*.

Имя входного файла *nQueens\_Test\_Solution.mat* следует изменить, если выбран другой файл

%}

inputFileName= 'nQueens\_Test\_Solution.mat';

%inputFileName= 'kQueens\_Test\_Composition.mat';

iInfo=['Input file name: ' inputFileName];

disp(iInfo);

Q=importdata(inputFileName);

% Вывод размера шахматной доски

n=length(Q);

disp(' ');

tStr = sprintf(' The size of a chessboard = %d',n);

disp(tStr);

disp(' ')

% *n2* – размер диагональных контрольных массивов

n2=n\*2;

% Формирование массивов для диагонального контроля.

D1=zeros(1,n2,'uint8');

D2=zeros(1,n2,'uint8');

B=zeros(1,n,'uint8');

%{

Определяим количество строк на шахматной доске, на которых не расположен ферзь, т.е.количество свободных строк. Это число равно общему количеству нулевых ячеек в массиве *Q*.

%}

nZero=sum(Q==0);

qError = 0;

%{

*qError* - индекс правильности решения. Если *qError* == 0, то решение верное, в противном случае, если *qError == 1*, то решение содержит ошибку.

%}

tic

%{

Если *nZero == 0*, то это означает, что входные данные являются решением задачи *n-Queens*, в противном случае - входные данные представляют собой *композицию*.

%}

%{

Суть алгоритма заключается в том, что в обнуленные ячейки контрольных массивов *D1, D2 и B* можно записать *1* только один раз, иначе одно из трех условий задачи будет нарушено.

%}

if nZero==0

for i=1:n

j=Q(i);

r=n+j-i;

t=j+i;

if D1(r)==0 && D2(t)==0 && B(j)==0

D1(r)=1;

D2(t)=1;

B(j)=1;

else

qError=1;

break

end

end

else

for i=1:n

if Q(i)>0

j=Q(i);

r=n+j-i;

t=j+i;

if D1(r)==0 && D2(t)==0 && B(j)==0

D1(r)=1;

D2(t)=1;

B(j)=1;

else

qError=1;

break

end

end

end

end

toc

% Вывод результатов анализа

if qError==0

if nZero==0

disp('Solutions is ok!')

else

tStr = sprintf('Composition size = %d', n-nZero);

disp(tStr);

disp('Composition is ok!')

end

else

tStr = sprintf('Error in solution in row = %d',i);

disp(tStr)

end