Solution to Hexagonal problem

Program written by : K Sai Sharath

Software used : MATLAB

Input : Number of concentric Hexagons which the user thinks can be appropriate to find the 2000th number satisfying the criteria.

Output : Number corresponding to the 2000th such number.

MAIN

```
%Main program:
% -> Order of Operations
%
% -> Reproduce the hexagonal arrangement of numbers as per the
problem-
%    -statement
%
% -> Identify the 6 neighbhouring numbers of all the numbers based on
the-
%    -logic of nearest 6 numbers
%
% -> check if the center number is infact a factor of the product of
the-
%    -6 neighbhouring numbers
%     1) Find the prime factors of the center number and the
surronding
%     numbers
%     2) If all the prime factors of the center are present in the
prime
%     factors of all the 6 neighbhouring numbers. Then the center
number is a
%     factor of the 6 other numbers.
%
% stop the program when the 2000th number is found

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
function  main(hexes)
tic
clc
clear all
close all
```

```
% Number of concentric number of hexagons that the program has to run
for
hexNumber = hexes;

% Variables to store the coordinates of the numbers in the hexagon
arrangement
mainX = zeros(1,6*sum(1:hexNumber)+1);
mainY = zeros(1,6*sum(1:hexNumber)+1);

% Calculating the actual coordinates and plotting them if necessary
to-
% -visualize

for i=0:hexNumber
    i;
    [coordX,coordY] = hexPoints(i);
    if i==0
        mainX(1) = coordX;
        mainY(1) = coordY;
    else
        mainX(1:6*sum(1:i)+1) = [mainX(1:6*sum(0:i-1)+1),coordX];
        mainY(1:6*sum(1:i)+1) = [mainY(1:6*sum(0:i-1)+1),coordY];
    end
end

% Storing the neighbhours i.e., the 6 numbers surronding every
number-
% -in factors

factors = zeros(6*sum(0:hexNumber-1)+1,6);

% Since the last hexagon will not be having six surrounding
neighbhours-
% we stop finding the neighbhours just a hexagon before the last one.
for i=1:6*sum(0:hexNumber-1)+1
    probe = i;
    [minX,maxX,minY,maxY] = drawHexagon(mainX(probe),mainY(probe));
    a = find(mainX>=minX-0.1 & mainX<=maxX+0.1);
    b = mainY(a)>=minY-0.1 & mainY(a)<=maxY+0.1;
    c = a(b);
    c(c == probe) = [];
    factors(i,:) = c';
end

% Solution function checks if the centre divides the product of the
rest of
% the numbers exactly or not and keeps a count of how many satisfy
the
% criteria.
```

```
[count,eureka] = solution(factors,hexNumber);
toc
end
```

HEX-POINTS

```
% Points on hexagon. Calculates the coorinates of where the numbers
need to
% be.

function [M,N] = hexPoints(hexNumber)

% Number of points on the particular hexagon
if hexNumber == 0

    U = 0;
    V = 0;

    %       plott(U,V,hexNumber);
    %Center coordinates are an exception
    M = U;
    N = V;

else
    theta = 0 : pi/3 : 2*pi;
    theta = theta + pi/2;
    r = ones(1,7);

    %Cartesian coordinates of the vertices of the hexagon is
    %1x7 matrices
    [U,V] = pol2cart(theta,hexNumber*r);
    [M,N] = midpointss(U,V,hexNumber);
    %         plott(M,N,hexNumber);

end

end
```

MID-POINTS

```
% Since after the first hexagon the numbers start occupying the
positions
```

```
% along the edges of the Hexagon in a incrimental fashion. That
observation
% is used to find the coordinates of the numbers and position them
% accordingly.

function [M,N] = midpointss(X,Y,hexNumber)

iteration = 0;
i = 1;
M = zeros(1,6*hexNumber);
N = zeros(1,6*hexNumber);

while i <= 6*hexNumber
    % Number of mid points based on the hexagon number
    midPoints = hexNumber-1;
    % This is used to keep track of calculation occuring on which
line
    iteration = iteration + 1;
    M(i) = X(iteration);
    N(i) = Y(iteration);

    if midPoints ~= 0
        % m+n the ratio is same for all divisions
        denominator = midPoints + 1;
        j=1;

        while midPoints>0
            i = i + 1;

            M(i) =
(j*X(iteration+1)+midPoints*X(iteration))/denominator;
            N(i) =
(j*Y(iteration+1)+midPoints*Y(iteration))/denominator;

            midPoints = midPoints - 1;
            j = j+1;
        end
    end
    i = i+1;
end
end
```

PLOT

```
% Plotting of points to better visualize
function plott(M,N,hexNumber)
```

```
% Plotting '*' everywhere the number is supposed to be

plot(M,N,'*')

if hexNumber ==0
    % Labeling the center
    text(M,N,'1')
else
    % Labeling the points with corresponding numbers to verif
    label = 6*sum(0:(hexNumber-1))+2:6*sum(0:hexNumber)+1;
    for a = 1:length(M)
        text(M(a),N(a),num2str(label(a)));
    end
end

hold on;
axis equal

end
```

DRAW HEXAGON

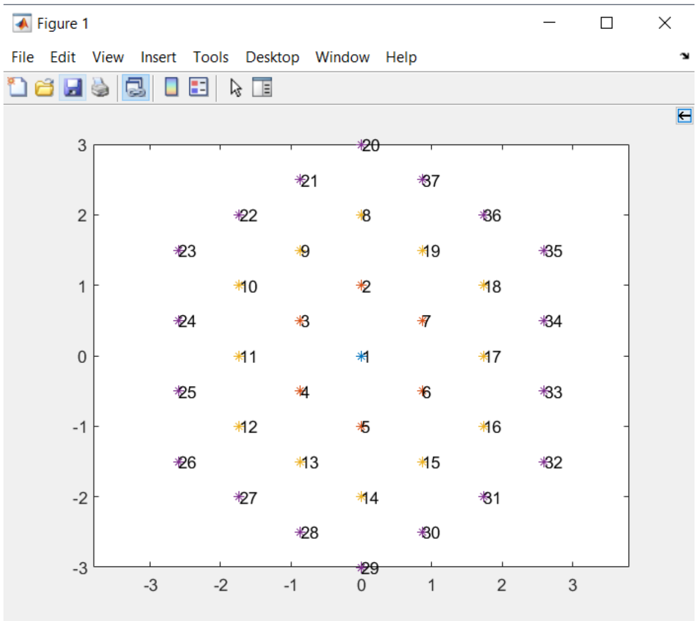
```
%Draw Hexagon i.e., to draw the boudaries of each number
function [minX,maxX,minY,maxY] = drawHexagon(Xcoord,Ycoord)
theta = 0 : pi/3: 2*pi;
theta = theta + pi/2;
r = ones(1,7);

[X,Y] = pol2cart(theta,r);

X = X + Xcoord;
Y = Y + Ycoord;

minX = min(X);
maxX = max(X);

minY = min(Y);
maxY = max(Y);

[theta,r] = cart2pol(X,Y);

%     polar(theta,r);
End
```

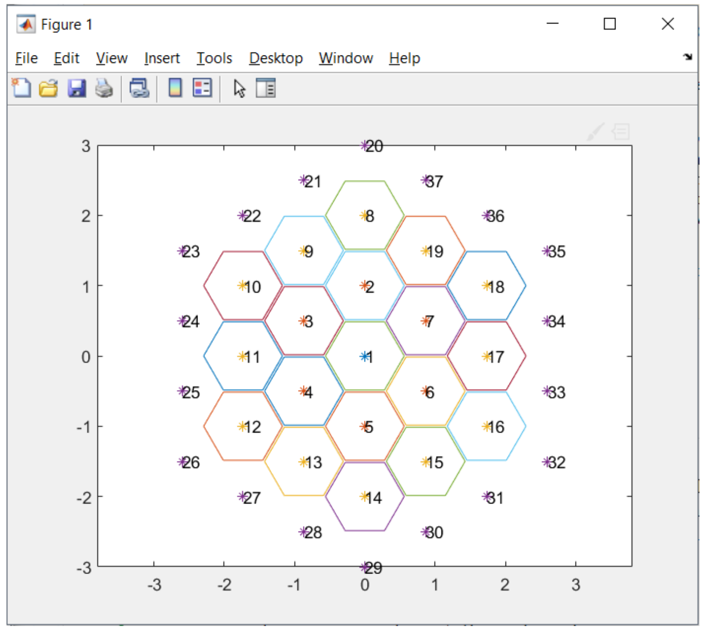
SOLUTION

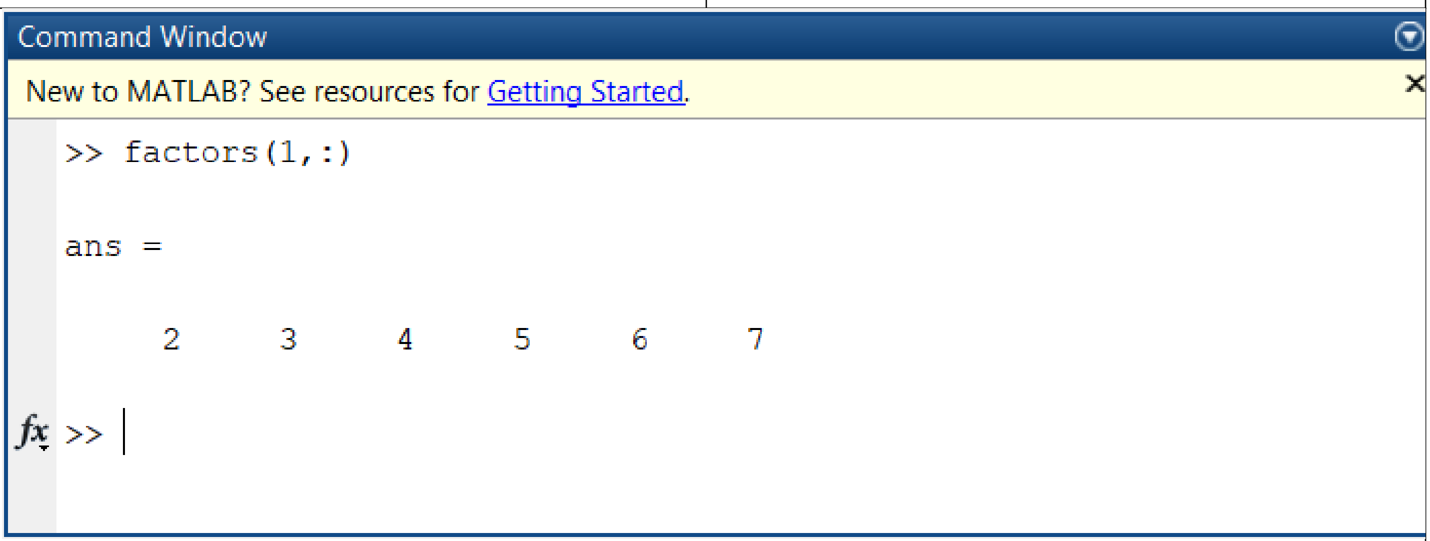
```
function [count,satisfy] = solution(factors,hexNumber)
count =1;
satisfy = zeros(1,2000);
```

```
satisfy(1,1) = 1;
%     for i=1:6*sum(0:hexNumber-1)+1

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% A crude way of checking if the centre is a factor of the product of
the
% rest of the 6 numbers. Results were not conclusive hence tryed
different
% approaches
%
%               if mod(prod(factors(i,:)),i) == 0

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
%2nd method where in  a*b*c mod x = (a mod X) *(b mod X) * (c mod X))
mod X
%
%               if
mod(mod(factors(i,1),i)*mod(factors(i,2),i)*mod(factors(i,3),i)...
%
*mod(factors(i,4),i)*mod(factors(i,5),i)*mod(factors(i,6),i),i) == 0
%
%              count =count+1
%              satisfy(1,count) = i;
%              if count == 2000
%                  disp(i)
%                  break
%              end
%          end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% 3rd Method: The method that worked, find the prume factors and
checks is
% all of the prime factors of center are present in the prime factors
of
% rest of the surrounding numbers.

for i=2:6*sum(0:hexNumber-1)+1
    primefactors = [factor(factors(i,1)),factor(factors(i,2))...
        ,factor(factors(i,3)),factor(factors(i,4)),factor(factors(i,5
))...
        ,factor(factors(i,6))];
    dividend = factor(i);
    for eks=1:length(dividend)
        if ismember(dividend(eks),primefactors) &&
length(dividend)==nnz(dividend)
            for irpselon=1:length(primefactors)
```

```
                if dividend(eks) == primefactors(irpselon)
                    primefactors(irpselon) = 0;
                    dividend(eks) = 1;
                    break
                end
            end
        else
            dividend(eks)=0;
            break
        end
    end

    if length(dividend)==nnz(dividend)
        count = count+1;
        satisfy(1,count) = i;
        if count == 2000 || count ==30
            disp(i)
            break
        end
    end
end
end
```



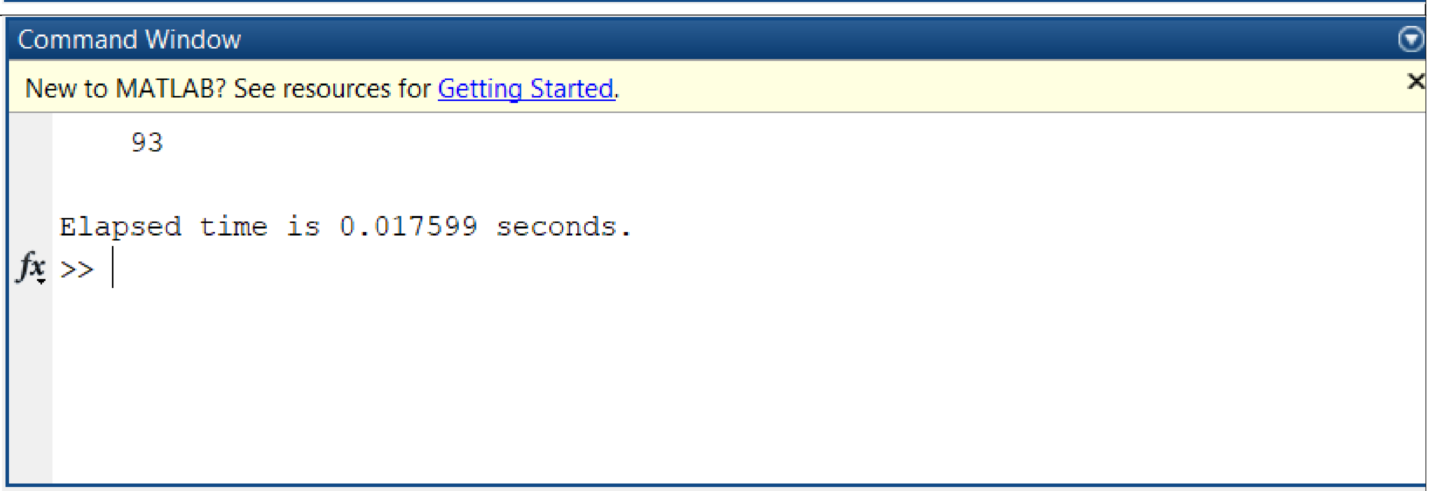




Command Window

New to MATLAB? See resources for Getting Started.

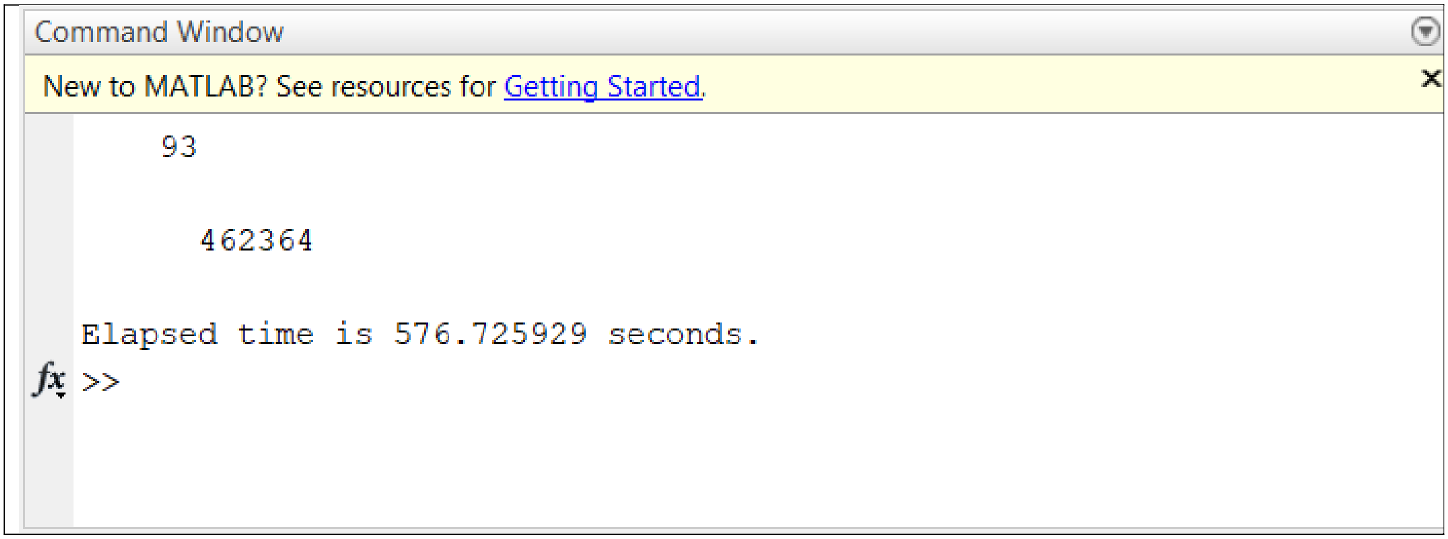
```
>> factors(1,:)

ans =

     2     3     4     5     6     7

>>
```



Command Window

New to MATLAB? See resources for Getting Started.

```
    93

Elapsed time is 0.017599 seconds.
>>
```



93 is the 30th satisfying number.

462364 is the 2000th satisfying number.