003-Topic: Logical Arrays & Masks

key

1. Given n=14;
   ii=[1:1:n+1];

By logical arrays produce the following sequence with n+1 elements:

c=[0 1 0 1 0 1 0 1 0 1 0 1 0 1 0]

SOLUTION

```
n=14;
ii=[1:1:n+1];            % ii stands for the c indices
c= mod(ii,2)==0          % Note that n+1 coefficients are created
c =
  0   1   0   1   0   1   0   1   0   1   0
```

2. For the array:

ii= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];

Create a logical array that identifies values equal or smaller than 10 within ii.

SOLUTION

```
ii=[1:1:20];
LA=ii<=10;               % LA=logical array
ii(LA)                   % remove ";" to print on the screen
```

3. For the array:

ii= [1:1:10];

Create a logical array that identifies values smaller than 5 or greater than or equal to 8. Store results on the new array **ii58**.

SOLUTION

```
ii=[1:1:10];
LA=ii<5 | ii>=8;         % LA=logical array
ii58 = ii(LA);           % LA is used as index of ii-array, i.e., a mask
```

4. For the array ii=[3:3:99] create a logical array that identifies multiple-of-6 values and store them in the new ii6 array

SOLUTION

```
ii= [ 3:3:99];
LA=mod(ii,6)==0;
ii6=ii(LA);                   % LA is used as index of ii-array, LA is a mask
```

5. Given

    A = [1:1:100];

Count the multiple-of-seven elements in the A-array. Print the result.

ANSWER

```
A=[1:1:100];
B=mod(A,7)==0;
Count=sum(B);          % sum add up logical values as numbers? interesting

fprintf('There are %d multiple-of-seven values in the range [1,100]',Count);
```

6. By the random function "**rand**" creates the **score** array of 50 values in the range of [0,100] representing the scores of the partial examen 1 of the inge3016 class for 50 students. Create the logical array **passLA** that identifies scores greater or equal to 60. Use this array to separate the passing students from the class set and store them in the new **pass** array.

ANSWER

```
score=rand(1,50).*100;   % creates a set of 'fake' grades for a 50-student class
passLA=score>=60;        %passLA is the logical array
pass=score(passLA);      %pass contains only the student that passed the course
```

7. The inge3016 class has 40 students whose final scores are stored in ng=[78,97,59,84,…88]. Where ng(1) is the score of student 1, ng(2) is the score of student 2 and so on. By **logical arrays** (and **masks,** if necessary) count the number of students who got A, B, C, D, and F.

    ANSWER

```
ng=[78,97,59,84,…88];

A=ng>=90;
B=ng>=80 & ng <90;
```

```
C=ng>=70 & ng<80;
D=ng>=60 & ng<70;       % A, B, C, D and F are logical arrays
cA=sum(A);   cB=sum(B);  cC=sum(C);  cD=sum(D);  cF=sum(F);

% The last statements can be replaced by:
cA=length(ng(A));
cB=length(ng(B));
cC=length(ng(C));
cD=length(ng(D));
cF=length(ng(F));
```

8. For the Puerto Rican population of 4 million, we know the age of each individual stored in the **age** array as **age**=[3, 74, 56, 12, …, 26, 21]. We are interested in counting the number of voters in PR. By the logical array **vote**, identify the number of voters (the voting age is 18). Use a library function to count the number of voters eligible to vote in **age** array.

ANSWER

```
age=[3, 74, 56, 12, …, 26, 21];          % 4 million elements
vote=age>=18;                            % vote is the logical array
nv=numel(age(vote))                      % nv=number of voters

% Last statement can be replaced by:
nv= sum(vote)
```

9. Given A=[10,9,8,4,5,6,7,1,2,3]’ substitute some values of A. Obtain the natural logarithm of the values of A which are equal to their array index (i.e., A(ii)=ii) and store them into the same position in A.

<table>
<tr><th>By logical array, mask, vectorized code</th><th>% By loops and control logic</th></tr>
<tr><td><pre>
% LogicalAndMasks01.m
% Modifies A array
clc, clear

A=[10,9,8,4,5,6,7,1,2,3];
AA=A;

ii=1:numel(A);
B=A(ii)==ii;   % logical array
A(B)=log(A(B));
disp([AA',A']);
</pre></td><td><pre>
% LogicalAndMasks02.m
% Modifies A array
clc, clear

A=[10,9,8,4,5,6,7,1,2,3];
AA=A;
for ii=1:numel(A)
   if A(ii)==ii
      A(ii)=log(A(ii));
   end
end
disp([AA',A']);
</pre></td></tr>
</table>

10. *A number of Matlab's functions are designed to execute a test of some sort, then return a logical array. The output* of these functions or commands is usually used as mask or index. For example, create a magic square with the following command (consult MATLAB help to learn more about the magic function).

```
>> A=magic(5)
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

Find all the primes in the matrix A with Matlab's **isprime** function. After applying the **isprime** function, each position that contains a logical 1 (true) indicates a prime in the corresponding position of the matrix A. Please get a listing of the primes in matrix A.

SOLUTION

```
>> A=magic(5)
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

We'll find all the primes in matrix A with Matlab's **ispime** command.

```
>> B=isprime(A)
B =
     1     0     0     0     0
     1     1     1     0     0
     0     0     1     0     0
     0     0     1     0     1
     1     0     0     1     0
```

Each position in matrix B that contains a logical 1 (true) indicates a prime in the corresponding position of matrix A. We can get a listing of the primes in matrix A.

```
 >> A(B)
ans =
    17
    23
    11
     5
     7
    13
    19
     2
     3
```

11.- **Class Quiz (P2014):** Given the array a=[9,8,7,6,5,4,3,2,1,0]. Square the elements of the a-array whose values are equal to their indexes.

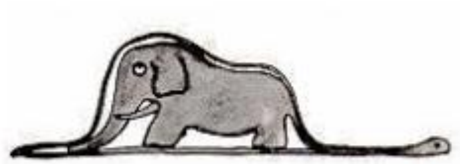
a=[9,8,7,6,5,4,3,2,1,0].
ii=[1,2,3,4,5,6,7,8,9,10];   % to represent the indices of the a array
LA=a==ii   % compares each **a** value to its index
a(LA)=a(LA).^2;   % squared elements are stored in the original array

12.- **Emigrants** (Class quiz). People, male and female professionals, in the range of [24,26] years of age are most likely to emigrate from PR (to the USA) in the next 3 years. Identify the number of potential emigrants and count them within the array age=[4, 45, 67, 13, 24, 93, 105, 7, 19, …,34]. The population of PR is 3.5 millions

% Given

clc,clear
age=rand(1,3.5e6)*100; % simulate actual age array
e=age>=24 & age<=26;
pe=numel(age(e));

13.- **Logical Arrays and El Principito**. By logical arrays find the number of "ñ" (eñes; MATLAB➔ enhe) that appears in the paragraph below.

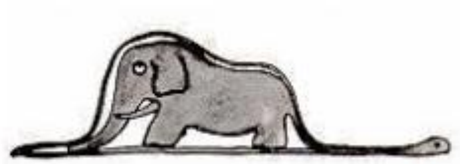


Las personas mayores me aconsejaron abandonar el dibujo de serpientes boas, ya fueran abiertas o cerradas, y poner más interés en la geografía, la historia, el cálculo y la gramática. De esta manera a la edad de seis años abandoné una magnífica carrera de pintor. Había quedado desilusionado por el fracaso de mis dibujos número 1 y número 2. Las personas mayores nunca pueden comprender algo por sí solas y es muy aburrido para los niños tener que darles una y otra vez explicaciones.

```
% principitoProgram
% we only selected a piece from the above paragraph

clc, clear, close

prin4='manera a la edad de seis años abandoné una magnífica carrera de   ';
enhe= 'ñ';

for ii=1:1:numel(prin4)
 enheL(ii)=strcmpi(prin4(ii),enhe); % compare each element of prin4 with ñ
end

% sum the true or 1's:
countenhe= sum(enheL)
```

% `TF = strcmpi(str,str)` compares two character vectors for equality, ignoring any differences in letter case. The character vectors are considered to be equal if the size and content of each are the same. The function returns a scalar logical 1 for equality, or scalar logical 0 for inequality. (see: https://www.mathworks.com/help/matlab/ref/strcmpi.html)

14.- Masks and El Principito. By logical arrays and masks find the number of "ñ" (eñes) that appears in the paragraph below and replace them by "n" (enes). Hint: assume the whole paragraph, starting with the L and ending with the s, is stored in the array principito.



Las personas mayores me aconsejaron abandonar el dibujo de serpientes boas, ya fueran abiertas o cerradas, y poner más interés en la geografía, la historia, el cálculo y la gramática. De esta manera a la edad de seis años abandoné una magnífica carrera de pintor. Había quedado desilusionado por el fracaso de mis dibujos número 1 y número 2. Las personas mayores nunca pueden comprender algo por sí solas y es muy aburrido para los niños tener que darles una y otra vez explicaciones.

```
% principitoProgram

clc, clear, close

prin4='manera a la edad de seis años abandoné una magnífica carrera de   ';
prin5='algo por sí solas y es muy aburrido para los niños tener que      ';

principito=[prin4,prin5];
enhe= 'ñ';

for ii=1:1:(numel(prin4)+numel(prin5))
 enheL(ii)=strcmpi(principito(ii),enhe);    % compare each element with ñ
end

% sum the true or 1's:
countenhe= sum(enheL)

OUTPUT
countenhe =
                    2
```

15.- I want to create a logical array from x and y with the same size as x. Whenever x=y the logical array contain 1's for a n samples, otherwise 0's.

What is the quickest way to do that, other than writing for-loops?

Here is one way of doing it:

```
>> x = 1:1:10
x =
   1 2 3 4 5 6 7 8 9 10
>> y = [2, 5, 7]
```

```
y =
   2 5 7
>> z = ismember(x,y)     % ismember function is logical, produces 0's and 1's
z =
   0 1 0 0 1 0 1 0 0 0

>> whos
 Name        Size    Bytes   Class Attributes
 x           1x10    80      double
 y           1x3     24      double
 z           1x10    10      logical
```

16.- Bonus due to Attendance.

The absences of 39 students are stored in the array ABS=[0,1,3,0,0,0,5,0,0,0,….,1], thirty nine values. Bonus due to attendance are:

- If student has perfect attendance (0 absences), ATT=3% added to the final examination score (**FAL**)
- If student has 1 absence, ATT is equal to 2% added to the final examination score
- If student has 2 or more absences, ATT=0

Show how you can use logical arrays and masks to add **ATT** to **FAL. Take into account that** ABS-array and FAL-array have same number of elements and students are identified by their index. **HINT: Develop three logical arrays using the three conditions above.**

SOLUTION-1:

% Given

```
ABS=[0,1,3,0,0,0,5,0,0,0,….,1],          % 39 VALUES
FAL=[89,78,96,100,99,56….93]             %39 VALUES

tres = ABS == 0;        % tres is a logical array
dos = ABS == 1;         % dos is a logical array
cero= ABS >= 2;         % cero is a logical array

FAL(tres) = FAL(tres) + 3;       % tres is used as index of FAL, therefore tres is now a mask
FAL(dos) = FAL(dos) + 2;         % dos is used as index of FAL, therefore dos is now a mask
FAL(cero) = FAL(cero) + 0;       % cero is used as index of FAL, therefore cero is now a mask
```

SOLUTION-2:

% Given

```
ABS=[0,1,3,0,0,0,5,0,0,0,….,1],          % 39 VALUES
FAL=[89,78,96,100,99,56….93]             %39 VALUES

ATT=[0,2,3];

tres = ABS == 0;        % tres is a logical array
dos = ABS == 1;         % dos is a logical array
cero= ABS >= 2;         % cero is a logical array

FAL(tres) = FAL(tres) + ATT(3);          % tres is used as index of FAL, therefore tres is a mask
FAL(dos) = FAL(dos) + ATT(2);            % dos is used as index of FAL, therefore dos is a mask
FAL(cero) = FAL(cero) + ATT(1);          % cero is used as index of FAL, therefore cero is a mask
```

SOLUTION-3

Some student may be tempted to solve it as follows:

% Given

```
ABS=[0,1,3,0,0,0,5,0,0,0,….,1],          % 39 VALUES
FAL=[89,78,96,100,99,56….93]             %39 VALUES

for ii=1:1:nume(ABS)

        if ABS(ii)==0;
                FAL(ii)=FAL(ii)+3;
        elseif ABS(ii)==1;
                FAL(ii)=FAL(ii)+2;
        endif
end
```

HOWEVER, take into account that this is not the **<u>requested</u>** method. The solution-1 and solution-2 are VECTORIZED , therefore both are faster than solution-1.

17.- **A-Students**: Given the Final Score (**FS**) array containing the final score of 40 students using logical arrays and masks assign the letter grade 'A' to students obtaining a final score, FS>=90. Assign 'Z' to the rest. Count the number of As.

With logical arrays and masks

```
% Assuming FS is already computed:
LG(1:1:numel(FS)) = 'Z';        % LG is a string containing 40 'Z'
FSA = FS>=90;                   % FSA is logical array
LG(FSA) = 'A';                  % 'A' substitute 'Z' for 'A' students
countA = sum(FSA);              % count the number of As
```

With for loops and if constructs:

```
% Assuming FS is already computed:
countA=0;

for ii=1:numel(FS))

        if FS(ii)>=90
            LG(ii)='A';
            countA=countA+1;
        else
            LG(ii)='Z';
        end

end
```

18. By logical arrays (and masks) construct the sequence of the coefficients C in the Simpson Rule formulation:

C = [1 4 2 4 2 4 2 … 4 1]

SOLUTION-1
c=ones(1,n+1);
ii=[1:1:n+1];
b=mod(ii,2)==0;
c(b)=4.*c(b);
bb=mod(ii,2)~=0;
c(bb)=2.*c(bb);
c(1)=1;
c(n+1)=1;
(8 lines)
	SOLUTION-2
	c=ones(1,n+1);
	ii=[1:1:n+1];
	b=mod(ii,2)==0
	c(b)=4.*c(b);
	c(~b)=2.*c(~b);
	c(1)=1;
	c(n+1)=1;
	(7 lines)
		SOLUTION-3
		c=ones(1,n+1);
		b=mod(1:1:n+1,2)==0;
		c(b)=4.*c(b);
		bb=mod(1:1:n+1,2)~=0;
		c(bb)=2.*c(bb);
		c(1)=1;
		c(n+1)=1;
		(7 lines)

SOLUTION-4
c=ones(1,n+1);
b=mod(1:1:n+1,2)==0
c(b)=4.*c(b);
c(~b)=2.*c(~b);
c(1)=1;
c(n+1)=1;
(6 lines)
	SOLUTION-5
	c=ones(1,n+1);
	b=mod(1:1:n+1,2)==0;
	c(b)=4;
	bb=mod(1:1:n+1,2)~=0;
	c(bb)=2;
	c(1)=1;
	c(n+1)=1;
	(7 lines)
		SOLUTION-6
		c=ones(1,n+1);
		b=mod(1:1:n+1,2)==0
		c(b)=4;
		c(~b)=2;
		c(1)=1;
		c(n+1)=1;
		(6 lines)

SOLUTION-7:
c=2.*ones(1,n+1);
b=mod(1:1:n+1,2)==0
c(b)=4;
c(1)=1;
c(n+1)=1;
(5 lines)
	SOLUTION-8:
	c(3:2:n-1)=2;
	c(2:2:n)=4
	c(1)=1;
	c(n+1)=1;
	(4 lines)	

19. Next is an example of logical arrays application to the Midpoint Integration Rule

Midpoint Integration rule: $I=2h(f_2+f_4+...+f_n)$ where n is even

$$\int_{a=0}^{b=3} e^{-\frac{x}{2}}\left(2x-\frac{x^2}{2}\right)dx$$

```
clc, clear;
      a = 0;  b = 3; n = 100; h = (b-a)/n;
      x = [a:h:b];
      f=exp(-x./2).* (2.*x-x.^2./2);
      ii=[1:1:n+1];
```

```
c = mod(ii,2)==0;
t=c.*f;     I=2*h*sum(t);
```

20. By logical arrays (and masks) construct the sequence of the coefficients C in the Simpson Rule formulation:

C = [1 3 3 2 3 3 2… 3 3 1]

```
n=12;
c=3*ones(1,n+1);          % also c(1:1:n+1)=3   % initially all are 3
ii=[1:1:n+1];
b=mod(ii,3)==1;           % also b=mod(1:1:n+1,3)==1
c(b)=(2/3)*c(b);          % also c(b)=2/3
c(1)=1;
c(n+1)=1;
c                         % to print the results
```