Practical Worksheet 01 – Basics of Matlab

Exercise 01

Use the command window to initialize the following variables (the expression is correct if the qnszer is `var = 50`):

- $a = e^{\ln 50}$.
  ```
  >> a = exp(log(50))
  ```
- $b = 100 \sin \frac{\pi}{6}$.
  ```
  >> b = 100 * sin(pi / 6)
  ```
- $c = \lfloor e^{\pi} \rfloor + \lfloor \pi^{e} \rfloor + \lfloor \pi \rfloor + \lfloor e \rfloor$.
  ```
  >> c = floor(exp(pi)) + floor(pi ^ exp(1)) + floor(pi) + floor(exp(1))
  ```
- $d = \frac{2^{\frac{(2*2)!}{2+2}} + 22 + 2^{2^2} - \sqrt{2^2}}{2^{2-\frac{2}{2}}}$.
  ```
  >> d = (2 ^ (factorial(2 + 2) / (2 + 2)) + 22 + 2 ^ 2 ^ 2 - sqrt(2 ^ 2)) / (2 ^ (2 - 2 / 2))
  ```
- $f = 0.5 * (5 + 5)^{\frac{5^0}{0.5}}$.
  ```
  >> f = 0.5 * (5 + 5) ^ (5 ^ 0 / 0.5)
  ```
- $g = 5 * \left(\frac{5}{0.5} + 5^0\right) - 5$.
  ```
  >> g = 5 * (5 / 0.5 + 5 ^ 0) - 5
  ```
- $k = \frac{3^{3!} - 3^{3-3^0}}{3^{3-3^0}} - 30$.
  ```
  >> k = (3 ^ factorial(3) - 3 ^ (3 - 3 ^ 0)) / (3 ^ (3 - 3 ^ 0)) - 30
  ```

Exercise 02

Use the command window to initialize the following arrays

- $A = (1 \quad 2 \quad 3)$; $B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$; $C = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$.
  ```
  >> A = [1 2 3]
  >> B = [1 ; 2 ; 3]
  >> C = [1 2 3 ; 4 5 6 ; 7 8 9]
  ```
- D = a row array containing all the even numbers from 0 to 1000.
  ```
  >> D = 2 : 2 : 1000
  ```
- E = a column array containing all the odd numbers from 0 to 1000.
  ```
  >> E = 1 : 2 : 1000
  ```

- F = a row array containing 100 numbers spread evenly on the interval [3 ; 15].
  ```
  >> F = linspace(3, 15, 100)
  ```
- G = a magic matrix of size 3.
  ```
  >> G = magic(3)
  ```
- H = an identity matrix of size 4.
  ```
  >> H = eye(4)
  ```
- I = a random matrix of 2 rows and 3 columns.
  ```
  >> I = rand(2, 3)
  ```

Exercise 03

- Create the following matrices:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix},\ B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},\ C = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

$$D = \begin{pmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 & 25 \\ 72 & 66 & 60 & 54 & 48 & 42 & 36 & 30 & 24 \\ 0 & 0.125 & 0.25 & 0.375 & 0.5 & 0.625 & 0.75 & 0.875 & 1 \end{pmatrix}.$$

$$E = \begin{pmatrix} 6 & 43 & 2 & 11 & 87 \\ 12 & 6 & 34 & 0 & 5 \\ 34 & 18 & 7 & 41 & 9 \end{pmatrix}.$$

```
A = zeros(2, 5)
B = eye(3)
C = ones(3, 2)
D = [linspace(1,25,9);linspace(72,24,9);linspace(0,1,9)]
E = [6 43 2 11 87; 12 6 34 0 5; 34 18 7 41 9]
```

- Calculate $\boldsymbol{F = E^T}$.
  ```
  F = E'
  ```
- Calculate ***X = D * F***.
  ```
  X = D * F
  ```
- Calculate ***Y = F * D***.
  ```
  X = D * F
  ```
  (Error : incompatible matrix dimensions)
- Create a row vector ***vc*** of size 10 containing the elements of the first and the last row of ***E***, and q column vector ***vd*** of size 6 containing the elements of the second and forth column of ***E***.
  ```
  va = [E(1, :) E(3, :)]
  vb = [E(:, 2); E(:, 4)]
  ```
- Remove the last row and the third column of ***E***.
  ```
  E(end, :) = []
  E(:, 3) = []
  ```
  Ou bien
  ```
  E = E(1:end-1,[1,2,4,5])
  ```

Exercise 04

Write a Matlab script that prompts the user for a positive integer ***N***, then calculates and displays the sum of the digits that make up the integer.

```
a = input('Veuillez saisir un nombre entier S.V.P : ');
s = 0;
while a ~= 0
    s = s + mod(a, 10);
    a = floor(a / 10);
end;
display(s);
```

Exercise 05:

1. Write a Matlab function that takes as input a positive integer, and returns as result an array containing all its divisors.
   Examples : Input → 9 ; output → [1, 3, 9].
   Input → 10 ; output → [1, 2, 5, 10].
   Input → 11 ; output → [1, 11].

   ```
   function v = diviseurs(n)
        v = [];
        for i = 1:n
             if mod(n,i) == 0
                  v = [v i];
             end;
        end;
   ```

2. Write a Matlab function that takes a positive integer as input and determines whether it is a prime number or not. The function should return ***1*** if the number is prime and ***0*** otherwise.

   ```
   function p = premier(n)
        if length(diviseurs(n)) <= 2
             p = 1;
        else
             p = 0;
        end;
   ```

3. Write a Matlab script that asks the user to enter a positive integer ***N***, then if the number is prime displays: "the number (value of N) is prime". Otherwise display "the number (value of N) is not prime".

   ```
   n = input('Veuillez saisir un nombre : );
   if premier(n)
        display(strcat('le nombre ', num2str(n), ' est
   premier'));
   ```

```
    else
        display(strcat('le nombre ', num2str(n), ' n''est
    pas premier'));
    end;
```

Exercise 06:

Explain the result of each of the following commands:

1. `0.1 + 0.2 == 0.3`
   0.1 + 0.2 ≠ 0.3 because of a rounding error. The comparaison result is 0 (false).
2. `1 + 1e16 - 1e16`
   The result is 0 because the "1" is absorbed by the "1e16" and has no impact on the final result.
3. `1 + (1e16 - 1e16)`
   The result is correct in this case because the large numbers cancel each other first, however, that confirms the non-associativity of floating point addition.
4. `realmax * 2`
   That's what overflows look like.
5. `v = rand(1, 10000000);`
   `sum(v(end:-1:1)) == sum(v)`
   Another non-associativity example with random numbers.