Работа с матрицами в Scilab

Ввод:

```
--> Xn=-3.5;dX=1.5;Xk=4.5;
--> X=Xn:dX:Xk
X =
-3.5000 -2.0000 -0.5000 1.0000 2.5000 4.0000
```

Xn – начало массива, dX – шаг, Xk – конец массва

Пример ввода вектора строки :

```
--> V=[1 2 3 4 5]
V =
1 2 3 4 5
--> W=[1.1,2.3,-0.1,5.88]
W =
1.1000 2.3000 -0.1000 5.8800
```

Пример ввода вектора столбца

```
--> X=[1;2;3]
X =
1
2
3
```

Далее приведен пример задания матрицы и обращение к ее элементам:

```
--> A=[1 2 3;4 5 6;7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
--> A(1,2)^A(2,2)/A(3,3)
ans =  3.5556
```

Важную роль при работе с матрицами играет знак двоеточия «:». Указывая его вместо индекса при обращении к массиву, можно иметь доступ к группам его элементов. Например:

```
-->//Пусть задана матрица A
--> A=[5 7 6 5; 7 10 8 7;6 8 10 9;5 7 9 10]
--> //Выделить из матрицы A второй столбец
--> A(:,2)
ans =


 7
 10
 8
 7
```

```
--> //Выделить из матрицы A третью строку
--> A(3,:)
ans = 6    8    10   9
--> //Выделить из матрицы A подматрицу M
--> M=A(3:4,2:3)
M =
8     10
7     9
--> //Удалить из матрицы A второй столбец
--> A(:,2)=[]
A =
5     8     10
7     7     9
6     10    9
5     9     10
--> //Удалить из матрицы A третью строку
--> A(3,:)=[]
A =
5     8     10
7     7     9
5     9     10
```

Функции:

```
-->D=[1 2;3 4;5 6];
-->matrix(D,2,3)
ans  =
 1.   5.   4.
 3.   2.   6.
-->matrix(D,3,2)
ans  =
 1.   2.
 3.   4.
 5.   6.
-->matrix(D,1,6)
ans  =
 1.   3.   5.   2.   4.   6.
-->matrix(D,6,1)
ans  =
 1.
 3.
 5.
 2.
 4.
 6.
```

`ones(m,n)` создает матрицу единиц из `m` строк и `n` столбцов[8]

`zeros(m,n)` создает *нулевую матрицу*[9] из `m` строк и `n` столбцов

`eye(m,n)` формирует *единичную матрицу*[11] из `m` строк и `n` столбцов

`rand(n1,n2,…nn[,fl])` формирует многомерную матрицу *случайных чисел*, необязательный параметр `p` это символьная переменная, с помощью которой можно задать тип распределения случайной величины (`'uniform'` равномерное, `'normal'` гауссовское); `rand(m,n)` формирует матрицу `m` на `n` случайных чисел; `rand(M)` формирует матрицу случайных чисел, размер которой совпадает с размером матрицы `M`; результат функции `rand()` случайное скалярное число;

`tril(A[,k])` формирует из матрицы `A` нижнюю треугольную матрицу[14] начиная с главной или с `k` й диагонали;

`triu(A[,k])` формирует из матрицы `A` верхнюю треугольную матрицу[15] начиная с главной или с `k` й диагонали;

`sort(X)` выполняет упорядочивание массива `X`, если `X` матрица, сортировка выполняется по столбцам;

`size(V[,fl])` определяет размер массива `V`, если `V` двумерный массив, то `size(V,1)` или `size(v,'r')` определяют число строк матрицы `V`, а `size(V,2)` или `size(V,'c')` число столбцов;

`length(X)` определяет количество элементов массива `X`, если `X` вектор, его длину если `X` матрица, вычисляет общее число ее элементов;

`sum(X[,fl])` вычисляет *сумму* элементов массива `X`, имеет необязательный параметр `fl`; если параметр `fl` отсутствует, то функция `sum(X)` возвращает скалярное значение равное сумме элементов массива; если `fl='r'` или `fl=1`, что то же самое, то функция вернет строку равную поэлементной сумме столбцов матрицы `X`; если `fl='c'` или `fl=2`, то результатом работы функции будет вектор-столбец, каждый элемент которого равен сумме элементов строк матрицы `X`; частный случай применения функции `sum` это вычисление *скалярного произведения векторов*[16];

`prod(X[,fl])` вычисляет *произведение* элементов массива `X`, работает аналогично функции `sum`;

`max(M[,fl])` вычисляет *наибольший элемент* в массиве `M`, имеет необязательный параметр `fl`; если параметр `fl` отсутствует, то функция `max(M)` возвращает максимальный элемент массива `M`; если `fl='r'`, то функция вернет строку максимальных элементов столбцов матрицы `M`; если `fl='c'`, то результатом работы функции будет вектор-столбец, каждый элемент которого равен максимальному элементу соответствующих строк матрицы `M`; функция `[x, nom]=max(M[,fl])` вернет значение максимального элемента `x` и его номер в массиве `nom`;

`min(M[,fl])` вычисляет *наименьший элемент* в массиве `M`, работает аналогично функции `max`;

`mean(M[,fl])` вычисляет *среднее значение* массива `M`, если `M` двумерный массив, то `mean(M,1)` или `mean(M,'r')` определяют среднее значение строк матрицы `M`, а `mean(M,2)` или `mean(M,'c')` среднее значение столбцов;

`median(M[,fl])` вычисляет *медиану*[17] массива `M`, работает аналогично функции `mean`;

`det(M)` **вычисляет** ***определитель***[18] **квадратной матрицы** `M`;

```
-->M=[1 0 2;3 2 1;0 3 1];
-->det(M)
 ans  = 17.
-->Z=[1 2 2;0 1 3;2 4 4];
-->det(Z)
 ans  = 0.
```

Листинг 3.30

`rank(M[,tol])` **вычисление** ***ранга матрицы*** `M`[19] **с точностью** `tol`.

```
-->M=[1 0 2;3 2 1;0 3 1];
-->rank(M)
 ans  = 3.
-->Z=[1 2 2;0 1 3;2 4 4];
-->rank(Z)
 ans  = 2.
```