ΑΛΓΕΒΡΑ ΠΙΝΑΚΩΝ ΜΕ ΧΡΗΣΗ ΤΗΣ ΓΛΩΣΣΑΣ JULIA

Αναστασία-Ευτέρπη Ψήτου

Στα πλαίσια του μαθήματος

Λογισμός Πινάκων και Εφαρμογές

Διδάσκουσα: Μαριλένα Μητρούλη

Εαρινό Εξάμηνο 2021

Άλγεβρα Πινάκων

Διάνυσμα:

- `a = [2, 3, 4]` ή `a = [2; 3; 4]`

Πίνακας γραμμή:

- `a = [2 3 4]`
- Μέσω του ανάστροφου (`'`) , `a = [2;3;4]'`

Πίνακας στήλη:

- `a = reshape([2 3 4], 3, 1)` , όπου 3 είναι το πλήθος των γραμμών και 1 το πλήθος των στηλών.
- Μέσω του ανάστροφου (`'`) , `a = [2 3 4]'`

Πίνακας n x m:

- `A = [2 3 4; 5 6 7]`

Ισότητα πινάκων:

Έστω $\mathrm{A}, \mathrm{B} \in \mathbb{R}^{3\times 2}$ πίνακες ίδιας διάστασης:

In [1]:
```
#Μαθηματική Ισότητα: aij = bij για κάθε 1≤i≤3, 1≤j≤2.

A = [1 2; 3 4; 5 6]
B = [1 2; 3 4; 5 6]

A == B
```

Out[1]: `true`

Προσοχή: Δεν είναι όμως το ίδιο αντικείμενο, είναι διαφορετικά, παρόλο που αποτελούνται από ίδια στοιχεία.

In [2]:
```
A = [1 2; 3 4; 5 6]
B = [1 2; 3 4; 5 6]

A ≡ B

#ή A === B. (\equiv TAB)
```

Out[2]: `false`

In [3]:
```
#Παρακάτω έχουμε A,B ίδιο αντικείμενο.
A = [1 2; 3 4; 5 6]
B = A

A ≡ B
```

Out[3]: `true`

Ειδικές Μορφές Πινάκων:

Τετραγωνικός:

Στα παραδείγματα που ακολουθούν, χρησιμοποιούμε τη round για να προκύπτουν ακέραιες είσοδοι και έτσι να αποφεύγονται τα σφάλματα στρογγύλευσης.

In [4]:
```
#Τετραγωνικός πίνακας 4x4 με τυχαία στοιχεία στο διάστημα [5,15].

A = round.(10*(rand(4,4)) .+ 5)
```

Out[4]:
```
4×4 Array{Float64,2}:
  9.0   7.0  10.0   9.0
  6.0  13.0   9.0  10.0
  9.0   8.0  12.0   8.0
 11.0  13.0  11.0   6.0
```

Κάτω Τριγωνικός:

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia.

In [5]:
```
A = [1 0 0 0; 2 3 0 0; 4 5 6 0; 7 8 9 10]
```

Out[5]:
```
4×4 Array{Int64,2}:
 1  0  0   0
 2  3  0   0
 4  5  6   0
 7  8  9  10
```

- Με χρήση της συνάρτησης `LowerTriangular`.

In [6]:
```
using LinearAlgebra

#Μετατροπή κάτω πυκνού τριγωνικού πίνακα σε αραιό κάτω τριγωνικό.
A = [1 0 0 0; 2 3 0 0; 4 5 6 0; 7 8 9 10]
B = LowerTriangular(A)
```

Out[6]:
```
4×4 LowerTriangular{Int64,Array{Int64,2}}:
 1  ·  ·   ·
 2  3  ·   ·
 4  5  6   ·
 7  8  9  10
```

In [7]:
```
using LinearAlgebra

#Μετατροπή πληρη πίνακα σε αραιό κάτω τριγωνικό.
A = round.(10*(rand(4,4)) .+ 5)
B = LowerTriangular(A)
```

Out[7]:
```
4×4 LowerTriangular{Float64,Array{Float64,2}}:
 14.0    ·     ·     ·
  5.0   9.0    ·     ·
  9.0  10.0  14.0    ·
  8.0   9.0  13.0  11.0
```

- Υπάρχει και η συνάρτηση `UnitLowerTriangular`, όπου αντικαθιστά τα στοιχεία της διαγωνίου του κάτω τριγωνικού πίνακα με άσσους.

Άνω Τριγωνικός:

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia.

In [8]:
```
A = [1 2 3 4; 0 5 6 7; 0 0 8 9; 0 0 0 10]
```

Out[8]:
```
4×4 Array{Int64,2}:
 1  2  3   4
 0  5  6   7
 0  0  8   9
 0  0  0  10
```

- Με χρήση της συνάρτησης `UpperTriangular`.

In [9]:
```
using LinearAlgebra

#Μετατροπή άνω πυκνού τριγωνικού πίνακα σε αραιό άνω τριγωνικό.
A = [1 2 3 4; 0 5 6 7; 0 0 8 9; 0 0 0 10]
B = UpperTriangular(A)
```

Out[9]:
```
4×4 UpperTriangular{Int64,Array{Int64,2}}:
 1  2  3   4
 ·  5  6   7
 ·  ·  8   9
 ·  ·  ·  10
```

In [10]:
```
using LinearAlgebra

#Μετατροπή πληρη πίνακα σε αραιό άνω τριγωνικό.
A = round.(10*(rand(4,4)) .+ 5)
B = UpperTriangular(A)
```

Out[10]:
```
4×4 UpperTriangular{Float64,Array{Float64,2}}:
 10.0   7.0  8.0  7.0
  ·    11.0  9.0  8.0
  ·     ·    9.0  5.0
  ·     ·    ·    7.0
```

- Υπάρχει και η συνάρτηση `UnitUpperTriangular`, όπου αντικαθιστά τα στοιχεία της διαγωνίου του άνω τριγωνικού πίνακα με άσσους.

Παράδειγμα Εξίσωσης με Αραιό και Πυκνό άνω τριγωνικό πίνακα

In [131…
```
using LinearAlgebra

n = 10000
A = UpperTriangular(rand(n,n))      #Αραιός
B = Matrix(A)                       #Πυκνός
c = ones(n)

@show sizeof(A)
@show sizeof(B)
@time A\c
@time B\c;
```

```
sizeof(A) = 8
sizeof(B) = 800000000
  0.022621 seconds (7 allocations: 78.516 KiB)
  0.067006 seconds (13 allocations: 78.797 KiB)
```

Διαγώνιος:

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia.

In [12]:
```
A = [2 0 0; 0 1 0; 0 0 3]
```

Out[12]:
```
3×3 Array{Int64,2}:
 2  0  0
 0  1  0
 0  0  3
```

- Με χρήση της συνάρτησης `Diagonal`.

In [13]:
```
using LinearAlgebra

#Μετατροπή πυκνού διαγώνιου πίνακα σε αραιό διαγώνιο πίνακα.
A = [2 0 0; 0 1 0; 0 0 3]
B = Diagonal(A)
```

Out[13]:
```
3×3 Diagonal{Int64,Array{Int64,1}}:
 2  ·  ·
 ·  1  ·
 ·  ·  3
```

In [14]:
```
using LinearAlgebra

#Μετατροπή πληρη πίνακα σε αραιό διαγώνιο.
A = round.(10*(rand(4,4)) .+ 5)
B = Diagonal(A)
```

Out[14]:
```
4×4 Diagonal{Float64,Array{Float64,1}}:
 10.0    ·     ·     ·
  ·    10.0    ·     ·
  ·     ·    13.0    ·
  ·     ·     ·    11.0
```

In [15]:
```
using LinearAlgebra

#Δημιουργία διαγώνιου πίνακα μέσω ενός διανύσματος v.
v = [1; 2; 3; 4; 5]
B = Diagonal(v)
```

Out[15]:
```
5×5 Diagonal{Int64,Array{Int64,1}}:
 1  ·  ·  ·  ·
 ·  2  ·  ·  ·
 ·  ·  3  ·  ·
 ·  ·  ·  4  ·
 ·  ·  ·  ·  5
```

- Παρατηρούμε ότι οι παραπάνω συναρτήσεις ουσιαστικά "κόβουν" κάποια στοιχεία των πινάκων, ώστε να αποκτήσουν τη ζητούμενη δομή.

Ταυτοτικός πίνακας: Συμβολίζεται με `I` και οι διαστάσεις του επηρεάζονται από τους γειτονικούς του πίνακες.

In [16]:
```
using LinearAlgebra

#Πίνακας Α διαστάσεων 3x4
A = round.(12*rand(3,4))

@show A == A*I
println("Ο I είναι πίνακας 4x4.\n")

@show A == I*A
println("Ο I είναι πίνακας 3x3.")
```

```
A == A * I = true
Ο I είναι πίνακας 4x4.

A == I * A = true
Ο I είναι πίνακας 3x3.
```

Αναπαράσταση Πινάκων:

- Κατά γραμμές:

In [17]:
```
v1 = [1 2 3]
v2 = [4 5 6]
v3 = [7 8 9]
A = [v1; v2; v3]
```

Out[17]:
```
3×3 Array{Int64,2}:
 1  2  3
 4  5  6
 7  8  9
```

- Κατά στήλες:

In [18]:
```
v1 = [1; 2; 3]
v2 = [4; 5; 6]
v3 = [7; 8; 9]
A = [v1 v2 v3]
```

Out[18]:
```
3×3 Array{Int64,2}:
 1  4  7
 2  5  8
 3  6  9
```

Πράξεις Διανυσμάτων

Εσωτερικό γινόμενο (Inner product)

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia.

In [19]:
```
a = [1; 2; 3]
b = [2; 3; 4]

@show a' * b;
```

```
a' * b = 20
```

- Μέσω των συναρτήσεων `dot(a, b)` ή `a ⋅ b` = `a \cdot TAB b`, όπου a, b διανύσματα.

In [20]:
```
using LinearAlgebra

# a, b διανύσματα
a = [1; 2; 3]
b = [2; 3; 4]

@show dot(a, b)
@show a ⋅ b;
```

```
dot(a, b) = 20
a ⋅ b = 20
```

In [21]:
```
using LinearAlgebra

# a, b πίνακες γραμμής
a = [1 2 3]
b = [2 3 4]

@show dot(a, b)
@show a ⋅ b;
```

```
dot(a, b) = 20
a ⋅ b = 20
```

Ιδιότητες Εσωτερικού γινομένου (Inner product properties)

1. $a^T b = b^T a$

In [22]:
```
using LinearAlgebra

a = 123*rand(456) .+ 78
b = 123*rand(456) .+ 78

@show a' * b == b' * a
@show dot(a,b) == dot(b,a)
@show a ⋅ b == b ⋅ a;
```

```
a' * b == b' * a = true
dot(a, b) == dot(b, a) = true
a ⋅ b == b ⋅ a = true
```

2. $(ka^T)b = k(a^T b) = ka^T b,\ k \in \mathbb{R}$

In [23]:
```
using LinearAlgebra

a = 12*rand(BigFloat,34) .+ 56
b = 12*rand(BigFloat,34) .+ 56
k = rand(BigFloat)

@show (k*a')*b
@show k*(a'*b)
@show k*a'*b
@show k*dot(a,b)
@show dot(k*a,b)

(k*a')*b == k*(a'*b) == k*a'*b
```

```
(k * a') * b = 127165.81202048392727127576128896464886306262443692286839777763280001557870001 25
k * (a' * b) = 127165.81202048392727127576128896464886306262443692286839777763280001557870001 59
k * a' * b = 127165.81202048392727127576128896464886306262443692286839777763280001557870001 25
k * dot(a, b) = 127165.81202048392727127576128896464886306262443692286839777763280001557870001 59
dot(k * a, b) = 127165.81202048392727127576128896464886306262443692286839777763280001557870001 25
```

Out[23]: false

In [135…

```
using LinearAlgebra

a = 12*rand(BigFloat,34) .+ 56
b = 12*rand(BigFloat,34) .+ 56
k = rand(BigFloat)

@show (k*a')*b
@show k*(a'*b)
@show k*a'*b
@show k*dot(a,b)
@show dot(k*a,b)

(k*a')*b == k*(a'*b) == k*a'*b
```

```
(k * a') * b = 72046.09840298225217552185991324323216617053779092993432873313098 5704538275311
k * (a' * b) = 72046.09840298225217552185991324323216617053779092993432873313098 5704538275311
k * a' * b = 72046.09840298225217552185991324323216617053779092993432873313098 5704538275311
k * dot(a, b) = 72046.09840298225217552185991324323216617053779092993432873313098 5704538275311
dot(k * a, b) = 72046.09840298225217552185991324323216617053779092993432873313098 5704538275311
```

Out[135… true

Παρατήρηση: Από τον έλεγχο της παραπάνω ιδιότητας, φαίνεται ότι η θεωρητική τιμή, λόγω των σφαλμάτων στρογγύλευσης που υπεισέρχονται στην αριθμητική κινητης υποδιαστολής, μπορεί να διαφέρει από την υπολογιζόμενη.

In [25]:

```
using LinearAlgebra

a = round.(12*rand(34)) .+ 56
b = round.(12*rand(34)) .+ 56
k = round(123*rand())

@show (k*a')*b
@show k*(a'*b)
@show k*a'*b
@show k*dot(a,b)
@show dot(k*a,b)

(k*a')*b == k*(a'*b) == k*a'*b
```

```
(k * a') * b = 1.064032e7
k * (a' * b) = 1.064032e7
k * a' * b = 1.064032e7
k * dot(a, b) = 1.064032e7
dot(k * a, b) = 1.064032e7
```

Out[25]: true

3. $(a+b)^T c = a^T c + b^T c$

In [26]:
```
using LinearAlgebra

a = 123*rand(Int64,456) .+ 78
b = 123*rand(Int64,456) .+ 78
c = 123*rand(Int64,456) .+ 78

@show (a+b)'*c == a'*c + b'*c
@show dot(a+b,c) == dot(a,c) + dot(b,c);
```

```
(a + b)' * c == a' * c + b' * c = true
dot(a + b, c) == dot(a, c) + dot(b, c) = true
```

Εξωτερικό γινόμενο (Outer product)

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia.

In [27]:
```
a = [1; 2; 3]
b = [2; 3; 4]

a * b'
```

Out[27]:
```
3×3 Array{Int64,2}:
 2  3   4
 4  6   8
 6  9  12
```

Πράξεις Πινάκων

- Πρόσθεση $\mathrm{A}, \mathrm{B} \in \mathbb{R}^{3\times 2}$ πίνακες ίδιας διάστασης:

In [28]:
```
A = round.(10*(rand(3,2)) .+ 5)
B = round.(10*(rand(3,2)) .+ 5)

A + B
```

Out[28]:
```
3×2 Array{Float64,2}:
 26.0  23.0
 22.0  22.0
 15.0  26.0
```

- Πολλαπλασιασμός πίνακα $\mathrm{A} \in \mathbb{R}^{5\times 3}$:

1. Με βαθμωτό:

In [29]:
```
#Βαθμωτό
c = rand()

#Πίνακας
A = round.(10*(rand(5,3)) .+ 5)

#Γινόμενο
B = c*A
```

Out[29]:
```
5×3 Array{Float64,2}:
 1.24592   0.830616  0.726789
 1.1421    0.726789  1.1421
 0.726789  1.1421    1.24592
 0.830616  0.726789  0.622962
 0.934443  0.934443  0.622962
```

2. Με διάνυσμα:

In [30]:
```
#Διάνυσμα
b = [1; 2; 3]

#Πίνακας
A = round.(10*(rand(5,3)) .+ 5)

#Γινόμενο
y = A*b
```

Out[30]:
```
5-element Array{Float64,1}:
 64.0
 71.0
 64.0
 52.0
 39.0
```

3. Με πίνακα $\mathrm{B} \in \mathbb{R}^{3\times 5}$:

In [31]:
```
#Πίνακας 5x3
A = round.(10*(rand(5,3)) .+ 5)

#Πίνακας 3x5
B = round.(10*(rand(3,5)) .+ 5)

#Γινόμενο
C = A * B
```

Out[31]:
```
5×5 Array{Float64,2}:
 313.0  315.0  190.0  266.0  358.0
 354.0  294.0  193.0  266.0  369.0
 307.0  301.0  182.0  287.0  340.0
 224.0  216.0  132.0  198.0  248.0
 334.0  318.0  196.0  281.0  370.0
```

Ιδιότητες Πινάκων

1. Έστω πίνακες $\mathrm{A} \in \mathbb{R}^{4\times 3}$ και $\mathrm{B} \in \mathbb{R}^{3\times 3}$:

In [32]:
```
#A*B ορίζεται
A = round.(10*(rand(4,3)) .+ 5)
B = round.(10*(rand(3,3)) .+ 5)

A*B
```

Out[32]:
```
4×3 Array{Float64,2}:
 230.0  382.0  228.0
 246.0  424.0  256.0
 149.0  232.0  152.0
 197.0  312.0  194.0
```

In [33]:
```
#B*A δεν ορίζεται
A = round.(10*(rand(4,3)) .+ 5)
B = round.(10*(rand(3,3)) .+ 5)

B*A
```

```
DimensionMismatch("A has dimensions (3,3) but B has dimensions (4,3)")
```

2. $AB \neq BA$

In [34]:
```
A = [1 0; 0 2]
B = [1 2; 3 4]

@show A*B
@show B*A
print("Παρατηρούμε ότι AB ≠ BA.")
```

```
A * B = [1 2; 6 8]
B * A = [1 4; 3 8]
Παρατηρούμε ότι AB ≠ BA.
```

3. $AB = [0]_{n\times n} \nRightarrow A = [0]_{n\times n}$ ή $\mathrm{B} = [0]_{n\times n}$

In [35]:
```
A = [1 0; 0 0]
B = [0 0; 0 1]

@show A*B
print("Παρατηρούμε ότι AB = 0 ⇏ A = 0 ή B = 0.")
```

```
A * B = [0 0; 0 0]
Παρατηρούμε ότι AB = 0 ⇏ A = 0 ή B = 0.
```

4. $AB = AC,\ A \neq [0]_{n\times n} \nRightarrow B = C$

In [36]:
```
A = [1 0; 0 0]
B = [1 2; 3 4]
C = [1 2; 0 5]

@show A*B
@show A*C
print("Παρατηρούμε ότι AB = AC, A ≠ 0 ⇏ B = C.")
```

```
A * B = [1 2; 0 0]
A * C = [1 2; 0 0]
Παρατηρούμε ότι AB = AC, A ≠ 0 ⇏ B = C.
```

5. $(AB)C = A(BC)$ $\qquad\qquad (προσεταιριστική)$

In [37]:
```
A = [1 2; 3 4]
B = [4 3; 2 1]
C = [2 1; 3 4]

@show (A*B)*C
@show A*(B*C)
print("Παρατηρούμε ότι (AB)C = A(BC).")
```

```
(A * B) * C = [31 28; 79 72]
A * (B * C) = [31 28; 79 72]
Παρατηρούμε ότι (AB)C = A(BC).
```

6. $A(B+C) = AB + AC \qquad (\alpha\rho\iota\sigma\tau\varepsilon\rho\acute{\eta}\ \varepsilon\pi\iota\mu\varepsilon\rho\iota\sigma\tau\iota\kappa\acute{\eta})$

In [38]:
```
A = [1 2; 3 4]
B = [4 3; 2 1]
C = [2 1; 3 4]

@show A*(B+C)
@show A*B + A*C
print("Παρατηρούμε ότι A(B+C) = AB + AC.")
```

```
A * (B + C) = [16 14; 38 32]
A * B + A * C = [16 14; 38 32]
Παρατηρούμε ότι A(B+C) = AB + AC.
```

7. $(A+B)C = AC + BC \qquad (\delta\varepsilon\xi\iota\acute{\alpha}\ \varepsilon\pi\iota\mu\varepsilon\rho\iota\sigma\tau\iota\kappa\acute{\eta})$

In [39]:
```
A = [1 2; 3 4]
B = [4 3; 2 1]
C = [2 1; 3 4]

@show (A+B)*C
@show A*C + B*C
print("Παρατηρούμε ότι (A+B)C = AC + BC.")
```

```
(A + B) * C = [25 25; 25 25]
A * C + B * C = [25 25; 25 25]
Παρατηρούμε ότι (A+B)C = AC + BC.
```

8. $k(AB) = (kA)B = A(kB), k \in \mathbb{R}$

In [40]:
```
A = [1 2; 3 4]
B = [4 3; 2 1]
k = 5

@show k*(A*B)
@show (k*A)*B
@show A*(k*B)
print("Παρατηρούμε ότι k(AB) = (kA)B = A(kB).")
```

```
k * (A * B) = [40 25; 100 65]
(k * A) * B = [40 25; 100 65]
A * (k * B) = [40 25; 100 65]
Παρατηρούμε ότι k(AB) = (kA)B = A(kB).
```

9. $I_n A = A I_n = A$

In [41]:
```
using LinearAlgebra

A = [1 2; 3 4]

@show I*A
@show A*I
@show A
print("Παρατηρούμε ότι IₙA = AIₙ = A.")
```

```
I * A = [1 2; 3 4]
A * I = [1 2; 3 4]
A = [1 2; 3 4]
Παρατηρούμε ότι IₙA = AIₙ = A.
```

10. $0A = [0]_{n \times n}$

In [42]:
```
A = [1 2; 3 4]

@show 0*A
print("Παρατηρούμε ότι 0A = [0]ₙₓₙ.")
```

```
0A = [0 0; 0 0]
Παρατηρούμε ότι 0A = [0]ₙₓₙ.
```

Δυνάμεις Πινάκων

Έστω πίνακας $\mathrm{A} \in \mathbb{R}^{3 \times 3}$ και $p \in \mathbb{N}$:

In [43]:
```
A = [1 2 3; 4 5 2; 1 3 1]
p = 6

A^p
```

Out[43]:
```
3×3 Array{Int64,2}:
  53142   81479  46044
 106270  162955  92108
  53140   81474  46056
```

Ιδιότητες:

Έστω πίνακας $\mathrm{A} \in \mathbb{R}^{3 \times 3}$ και $p, q \in \mathbb{N}$:

1. $A^p A^q = A^{p+q} = A^{q+p} = A^q A^p$

In [44]:
```
A = [1 2 3; 4 5 2; 1 3 1]
p = 2
q = 4

@show A^p * A^q
@show A^(p+q)
@show A^(q+p)
@show A^q * A^p

A^p * A^q == A^(p+q) == A^(q+p) == A^q * A^p
```

```
A ^ p * A ^ q = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
A ^ (p + q) = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
A ^ (q + p) = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
A ^ q * A ^ p = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
```

Out[44]:
```
true
```

2. $(A^p)^q = A^{pq}$

In [45]:
```
A = [1 2 3; 4 5 2; 1 3 1]
p = 2
q = 3

@show (A^p)^q
@show A^(p*q)

(A^p)^q == A^(p*q)
```

```
(A ^ p) ^ q = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
A ^ (p * q) = [53142 81479 46044; 106270 162955 92108; 53140 81474 46056]
```

```
true
```

Συμμετρικός Πίνακας

Δημιουργία συμμετρικού πίνακα:

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia:

In [46]:
```
A = [1 2 3; 2 4 2; 3 2 5]
```

Out[46]:
```
3×3 Array{Int64,2}:
 1  2  3
 2  4  2
 3  2  5
```

- Με χρήση της συνάρτησης `Symmetric` :

Η συνάρτηση αυτή δέχεται έναν πίνακα $\mathrm{A} \in \mathbb{R}^{n \times n}$ και τον μετατρέπει σε συμμετρικό.
Τα στοιχεία που βρίσκονται πάνω από την κύρια διαγώνιο αντικαθιστούν συμμετρικώς τα στοιχεία που βρίσκονται κάτω από αυτή.

In [47]:
```
A = [1 0 2 0 3; 0 4 0 5 0; 6 0 7 0 8; 0 9 0 1 0; 2 0 3 0 4]
```

Out[47]:
```
5×5 Array{Int64,2}:
 1  0  2  0  3
 0  4  0  5  0
 6  0  7  0  8
 0  9  0  1  0
 2  0  3  0  4
```

In [48]:
```
using LinearAlgebra

#A = [1 0 2 0 3; 0 4 0 5 0; 6 0 7 0 8; 0 9 0 1 0; 2 0 3 0 4]
Symmetric(A)
```

Out[48]:
```
5×5 Symmetric{Int64,Array{Int64,2}}:
 1  0  2  0  3
 0  4  0  5  0
 2  0  7  0  8
 0  5  0  1  0
 3  0  8  0  4
```

Σε περίπτωση που θέλουμε τα στοιχεία που βρίσκονται κάτω από την κύρια διαγώνιο να αντικαταστήσουν τα στοιχεία που βρίσκονται πάνω από αυτή, τότε καλούμε τη συνάρτηση ως εξής:
`Symmetric(όνομα_πίνακα, :L)`

In [49]:
```
using LinearAlgebra

#A = [1 0 2 0 3; 0 4 0 5 0; 6 0 7 0 8; 0 9 0 1 0; 2 0 3 0 4]
Symmetric(A, :L)
```

Out[49]:
```
5×5 Symmetric{Int64,Array{Int64,2}}:
 1  0  6  0  2
 0  4  0  9  0
 6  0  7  0  3
 0  9  0  1  0
 2  0  3  0  4
```

Έστω $\mathrm{A}, \mathrm{B} \in \mathbb{R}^{n \times n}$ συμμετρικοί πίνακες:

- $\mathrm{A} + \mathrm{B}$ συμμετρικός πίνακας.

In [50]:
```
using LinearAlgebra
A = 123*rand(123,123) .+ 123
B = 123*rand(123,123) .+ 123

#issymmetric επιστρέφει true αν ο πίνακας είναι συμμετρικός, διαφορετικά false.
issymmetric(Symmetric(A) + Symmetric(B))
```

Out[50]: true

- $\mathrm{A} \cdot \mathrm{B}$ δεν είναι απαραίτητα συμμετρικός πίνακας.

In [51]:
```
using LinearAlgebra
A = [1 2; 3 4]
B = [1 2; 3 4]

issymmetric(Symmetric(A) * Symmetric(B))
```

Out[51]: true

In [52]:
```
using LinearAlgebra
A = round.(123*rand(123,123) .+ 123)
B = round.(123*rand(123,123) .+ 123)

issymmetric(Symmetric(A) * Symmetric(B))
```

Out[52]: false

Για να είναι ο $\mathrm{A} \cdot \mathrm{B}$ συμμετρικός πίνακας, πρέπει να ισχύει ότι $\mathrm{AB} = \mathrm{BA}$.

In [53]:
```
using LinearAlgebra

A = round.(12*rand(2,2) .+ 34)
B = round.(12*rand(2,2) .+ 34)

while A*B != B*A
    A = round.(12*rand(2,2) .+ 34)
    B = round.(12*rand(2,2) .+ 34)
end

issymmetric(Symmetric(A) * Symmetric(B))
```

Out[53]: true

Ανάστροφος Πίνακας

- Δημιουργείται μέσω του `'`, δηλαδή `A'` ή μέσω της συνάρτησης `transpose(A)`.

In [54]:
```
A = [1 2; 3 4]
A'
```

Out[54]:
```
2×2 Adjoint{Int64,Array{Int64,2}}:
 1  3
 2  4
```

- Ο συμμετρικός πίνακας ισούται με τον ανάστροφο του.

In [55]:
```
A = 10*rand(3,3) .+ 3
B = Symmetric(A)

B == B'
```

Out[55]:
```
true
```

Ιδιότητες Αναστρόφου

1. $(A^t)^t = A$

In [56]:
```
A = [1 2; 3 4]

@show (A')'
(A')' == A
```

```
(A')' = [1 2; 3 4]
```

Out[56]:
```
true
```

2. $(aA)^t = aA^t$

In [57]:
```
A = [1 2; 3 4]
a = 5

@show (a*A)'
@show a*A'
(a*A)' == a*A'
```

```
(a * A)' = [5 15; 10 20]
a * A' = [5 15; 10 20]
```

Out[57]:
```
true
```

3. $(A+B)^t = A^t + B^t$

In [58]:
```
A = [1 2; 3 4]
B = [5 6; 7 8]

@show (A + B)'
@show A' + B'
(A + B)' == A' + B'
```

```
(A + B)' = [6 10; 8 12]
A' + B' = [6 10; 8 12]
```

Out[58]:
```
true
```

4. $(AB)^t = B^t A^t$

In [59]:
```
A = [1 2; 3 4]
B = [5 6; 7 8]

@show (A*B)'
@show B'*A'
(A*B)' == B'*A'
```

```
(A * B)' = [19 43; 22 50]
B' * A' = [19 43; 22 50]
```

Out[59]: true

- Οι πίνακες AA^t, A^tA, $A + A^t$ είναι συμμετρικοί.

In [60]:
```
using LinearAlgebra
A = round.(42*rand(999,999)) .+ 12345

#issymmetric επιστρέφει true αν ο πίνακας είναι συμμετρικός, διαφορετικά false.

#A*A' == συμμετρικός (true)
@show issymmetric(A*A')

#A'*A == συμμετρικός (true)
@show issymmetric(A'*A)

#A + A' == συμμετρικός (true)
@show issymmetric(A + A');

#=
Διαφορετικά θα μπορούσαμε να χρησιμοποιήσουμε τις σχέσεις:
    A*A' == Symmetric(A*A') ή A*A' == (A*A')'
=#
```

```
issymmetric(A * A') = true
issymmetric(A' * A) = true
issymmetric(A + A') = true
```

- $A^t = -A \Rightarrow A$ $\alpha ντισυμμετρικός$

In [61]:
```
#Α κατεπιλογήν αντισυμμετρικός πίνακας
A = [0 1 -2; -1 0 3; 2 -3 0]

A' == -A
```

Out[61]: true

- Κάθε πίνακας γράφεται σαν άθροισμα ενός συμμετρικού και ενός αντισυμμετρικού.

In [62]:
```
A = round.(42*rand(999,999)) .+ 12345

A == (A+A')/2 + (A-A')/2
```

Out[62]: true

In [63]:
```
A = round.(42*rand(999,999)) .+ 12345

#B αντισυμμετρικός
B = A - A'
(A-A')' == A'- A == -(A-A')
```

Out[63]:
```
true
```

- Έστω $\mathrm{A} \in \mathbb{R}^{10^6 \times 2}$. Θέλουμε να υπολογίσουμε τον πίνακα $(\mathrm{A}^{\mathrm{T}}\mathrm{A}\mathrm{A}^{\mathrm{T}}) \in \mathbb{R}^{2 \times 10^6}$ με την οικονομικότερη διαχείριση θέσεων μνήμης. Πώς είναι προτιμότερο να εκτελεσθεί ο πολ/μός;

In [64]:
```
A = 42*rand(10^6,2) .+ 12345

@time (A'*A)*A'
@time A'*(A*A')
```

```
  0.006262 seconds (5 allocations: 15.259 MiB)
OutOfMemoryError()
```

Ερμιτιανοί Πίνακες

Δημιουργία Ερμιτιανού πίνακα:

- Χωρίς τη χρήση κάποιας εσωτερικής συνάρτησης της Julia:

In [65]:
```
A = [1 im 2+im; -im 5 3-im; 2-im 3+im 4]

#im = i (imaginery number, μιγαδικός αριθμός)
```

Out[65]:
```
3×3 Array{Complex{Int64},2}:
 1+0im  0+1im  2+1im
 0-1im  5+0im  3-1im
 2-1im  3+1im  4+0im
```

- Με χρήση της συνάρτησης `Hermitian`:

Η συνάρτηση αυτή δέχεται έναν πίνακα $\mathrm{A} \in \mathbb{C}^{n \times n}$ και τον μετατρέπει σε Ερμιτιανό.
Τα στοιχεία που βρίσκονται πάνω από την κύρια διαγώνιο αντικαθιστούν συμμετρικώς τα στοιχεία που βρίσκονται κάτω από αυτή με τα συζυγή τους.
Επίσης να σημειωθεί ότι τα στοιχεία που βρίσκονται στη διαγώνιο μετατρέπονται σε πραγματικούς αριθμούς.

In [66]:
```
A=[1 0 2+2im 0 3-3im; 0 2+5im 0 5 0; 6-6im 0 7+25im 0 8+8im; 0 9 0 1 0; 2+2im 0 3-3im 0 4+35im]
```

Out[66]:
```
5×5 Array{Complex{Int64},2}:
 1+0im  0+0im  2+2im   0+0im  3-3im
 0+0im  2+5im  0+0im   5+0im  0+0im
 6-6im  0+0im  7+25im  0+0im  8+8im
 0+0im  9+0im  0+0im   1+0im  0+0im
 2+2im  0+0im  3-3im   0+0im  4+35im
```

In [67]:
```
using LinearAlgebra

#A = [1 0 2+2im 0 3-3im; 0 2+5im 0 5 0; 6-6im 0 7+25im 0 8+8im; 0 9 0 1 0; 2+2im 0 3-3im 0 4]
Hermitian(A)
```

Out[67]:
```
5×5 Hermitian{Complex{Int64},Array{Complex{Int64},2}}:
 1+0im  0+0im  2+2im  0+0im  3-3im
 0+0im  2+0im  0+0im  5+0im  0+0im
 2-2im  0+0im  7+0im  0+0im  8+8im
 0+0im  5+0im  0+0im  1+0im  0+0im
 3+3im  0+0im  8-8im  0+0im  4+0im
```

Σε περίπτωση που θέλουμε τα στοιχεία που βρίσκονται κάτω από την κύρια διαγώνιο να αντικαταστήσουν τα στοιχεία που βρίσκονται πάνω από αυτή με τα συζυγή τους, τότε καλούμε τη συνάρτηση ως εξής:
`Hermitian(όνομα_πίνακα, :L)`

In [68]:
```
using LinearAlgebra

#A = [1 0 2+2im 0 3-3im; 0 2+5im 0 5 0; 6-6im 0 7+25im 0 8+8im; 0 9 0 1 0; 2+2im 0 3-3im 0 4]
Hermitian(A, :L)
```

Out[68]:
```
5×5 Hermitian{Complex{Int64},Array{Complex{Int64},2}}:
 1+0im  0+0im  6+6im  0+0im  2-2im
 0+0im  2+0im  0+0im  9+0im  0+0im
 6-6im  0+0im  7+0im  0+0im  3+3im
 0+0im  9+0im  0+0im  1+0im  0+0im
 2+2im  0+0im  3-3im  0+0im  4+0im
```

Ανάστροφος Συζυγής

Έστω $\mathrm{A} \in \mathbb{C}^{5\times 4}$:

In [69]:
```
#Δημιουργία πίνακα με στοιχεία μιγαδικούς αριθμούς καλώντας την rand με είσοδο ComplexF64/F32.
A = round.(10*rand(ComplexF64, 5,4)) .+ 4

#=
Δεύτερος τρόπος:
A = round.(10*rand(5,4)) + im*rand(5,4) .+ 4
=#
```

Out[69]:
```
5×4 Array{Complex{Float64},2}:
 10.0+5.0im  10.0+5.0im   4.0+1.0im  12.0+5.0im
  4.0+6.0im   8.0+8.0im   9.0+4.0im   6.0+8.0im
 13.0+8.0im   6.0+5.0im   7.0+0.0im  11.0+10.0im
 11.0+3.0im  11.0+5.0im  12.0+3.0im  10.0+10.0im
 14.0+5.0im  14.0+6.0im  13.0+3.0im  10.0+6.0im
```

- $\overline{\overline{A}}$ ο συζυγής πίνακας του A μέσω τις συνάρτησης `conj` :

In [70]:
```
Ā = conj(A)
```

Out[70]:
```
5×4 Array{Complex{Float64},2}:
 10.0-5.0im  10.0-5.0im   4.0-1.0im  12.0-5.0im
  4.0-6.0im   8.0-8.0im   9.0-4.0im   6.0-8.0im
 13.0-8.0im   6.0-5.0im   7.0-0.0im  11.0-10.0im
 11.0-3.0im  11.0-5.0im  12.0-3.0im  10.0-10.0im
 14.0-5.0im  14.0-6.0im  13.0-3.0im  10.0-6.0im
```

- $(\overline{A})^t = A^*$ ο ανάστροφος συζυγής πίνακας του A:

In [71]:
```
transpose(Ā)
```

Out[71]:
```
4×5 Transpose{Complex{Float64},Array{Complex{Float64},2}}:
 10.0-5.0im  4.0-6.0im  13.0-8.0im   11.0-3.0im   14.0-5.0im
 10.0-5.0im  8.0-8.0im   6.0-5.0im   11.0-5.0im   14.0-6.0im
  4.0-1.0im  9.0-4.0im   7.0-0.0im   12.0-3.0im   13.0-3.0im
 12.0-5.0im  6.0-8.0im  11.0-10.0im  10.0-10.0im  10.0-6.0im
```

Σημαντικό: Σε περίπτωση που θέλουμε τον ανάστροφο ενός πίνακας $\mathrm{A} \in \mathbb{C}^{m \times n}$ τότε χρησιμοποιούμε τη συνάρτηση `transpose`, καθώς αν χρησιμοποιήσουμε το `'`, τότε θα πάρουμε τον ανάστροφο συζυγή του A.

In [72]:
```
#Ανάστροφος συζυγής του A
A'
```

Out[72]:
```
4×5 Adjoint{Complex{Float64},Array{Complex{Float64},2}}:
 10.0-5.0im  4.0-6.0im  13.0-8.0im   11.0-3.0im   14.0-5.0im
 10.0-5.0im  8.0-8.0im   6.0-5.0im   11.0-5.0im   14.0-6.0im
  4.0-1.0im  9.0-4.0im   7.0-0.0im   12.0-3.0im   13.0-3.0im
 12.0-5.0im  6.0-8.0im  11.0-10.0im  10.0-10.0im  10.0-6.0im
```

In [73]:
```
#Ανάστροφος του A
transpose(A)
```

Out[73]:
```
4×5 Transpose{Complex{Float64},Array{Complex{Float64},2}}:
 10.0+5.0im  4.0+6.0im  13.0+8.0im   11.0+3.0im   14.0+5.0im
 10.0+5.0im  8.0+8.0im   6.0+5.0im   11.0+5.0im   14.0+6.0im
  4.0+1.0im  9.0+4.0im   7.0+0.0im   12.0+3.0im   13.0+3.0im
 12.0+5.0im  6.0+8.0im  11.0+10.0im  10.0+10.0im  10.0+6.0im
```

In [74]:
```
#Ανάστροφος συζυγής το Ā = ανάστροφος του A
Ā'
```

Out[74]:
```
4×5 Adjoint{Complex{Float64},Array{Complex{Float64},2}}:
 10.0+5.0im  4.0+6.0im  13.0+8.0im   11.0+3.0im   14.0+5.0im
 10.0+5.0im  8.0+8.0im   6.0+5.0im   11.0+5.0im   14.0+6.0im
  4.0+1.0im  9.0+4.0im   7.0+0.0im   12.0+3.0im   13.0+3.0im
 12.0+5.0im  6.0+8.0im  11.0+10.0im  10.0+10.0im  10.0+6.0im
```

Ιδιότητες συζυγούς ανάστροφου

1. $(A^*)^* = A$

In [75]:
```
A = 97*rand(ComplexF64, 998,999) .+ 12345

(A')' == A

#=
Διαφορετικά:
transpose(conj(transpose(conj(A)))) == A
=#
```

Out[75]:
```
true
```

2. $(aA)^* = \bar{a}A^*$

In [76]:
```
A = 86*rand(ComplexF64, 998,999) .+ 12345
a = rand(ComplexF64)

transpose(conj(a*A)) == conj(a)*A'
```

Out[76]: true

3. $(A + B)^* = A^* + B^*$

In [77]:
```
A = 42*rand(ComplexF64, 998,999) .+ 12345
B = 42*rand(ComplexF64, 998,999) .+ 12345

(A + B)' == A' + B'
```

Out[77]: true

4. $(AB)^* = B^*A^*$

In [78]:
```
A = round.(53*rand(ComplexF64, 998,999)) .+ 12345
B = round.(53*rand(ComplexF64, 999,998)) .+ 12345

(A*B)' == B' * A'
```

Out[78]: true

- Ένας πίνακας λέγεται Ερμιτιανός εάν $A = A^*$.

In [79]:
```
B = 64*rand(ComplexF64, 999,999) .+ 12345
A = Hermitian(B)

A == A'
```

Out[79]: true

- Έστω $\mathrm{A} \in \mathbb{C}^{n\times n}$. Τότε AA^*, A^*A, $A + A^*$, $i(A - A^*)$ είναι Ερμιτιανοί:

In [80]:
```
using LinearAlgebra
A = round.(64*rand(ComplexF64, 999,999)) .+ 12345

#ishermitian επιστρέφει true αν ο πίνακας είναι Ερμιτιανός, διαφορετικά false.

#A*A' == Ερμιτιανός (true)
@show ishermitian(A*A')

#A'*A == Ερμιτιανός (true)
@show ishermitian(A'*A)

#A + A' == Ερμιτιανός (true)
@show ishermitian(A + A')

#im*(A - A') == Ερμιτιανός (true)
@show ishermitian(im*(A - A'));
```

```
#=
Διαφορετικά θα μπορούσαμε να χρησιμοποιήσουμε τις σχέσεις:
  A*A' == Hermitian(A*A') ή   A*A' == adjoint(A*A')
όπου adjoint(πίνακα) το γνωστό adj(A).
=#
```

```
ishermitian(A * A') = true
ishermitian(A' * A) = true
ishermitian(A + A') = true
ishermitian(im * (A - A')) = true
```

Αντίστροφος Πίνακας

- Μπορούμε να βρούμε τον αντίστροφο A^{-1} ενός πίνακα $\mathrm{A} \in \mathbb{R}^{n \times n}$ μέσω της συνάρτησης `inv`.

In [81]:
```
#A πίνακας τέτοιος ώστε να υπάρχει ο A⁻¹.
A = [2 3; 4 5]

A⁻¹ = inv(A)
```

Out[81]:
```
2×2 Array{Float64,2}:
 -2.5   1.5
  2.0  -1.0
```

In [82]:
```
#A = [2 3; 4 5]

inv(A)*A == A*inv(A) == I
```

Out[82]:
```
true
```

Προτάσεις:

1. Αν ο πίνακας $\mathrm{A} \in \mathbb{R}^{n \times n}$ είναι αντιστρέψιμος τότε ο A^{-1} αντιστρέψιμος και ισχύει:

- $(A^{-1})^{-1} = A$
- $(A^{-1})^{*} = (A^{*})^{-1}$

In [83]:
```
A = round.(123*rand(123,123) .+ 123)

round.(inv(inv(A))) == A
```

Out[83]:
```
true
```

In [84]:
```
A = 123*rand(ComplexF64,123,123) .+ 123

inv(A)' == inv(A')
```

Out[84]:
```
true
```

2. Αν $\mathrm{A}, \mathrm{B} \in \mathbb{R}^{n \times n}$ τότε ο AB αντιστρέψιμος και ισχύει: $(AB)^{-1} = B^{-1}A^{-1}$

In [85]:
```
using LinearAlgebra
A = 123*rand(123,123) .+ 123
B = 123*rand(123,123) .+ 123

@show round.(inv(A*B) * (A*B)) == round.((A*B) * inv(A*B)) == I

@show round.(inv(A*B)) == round.(inv(B) * inv(A));
```

```
round.(inv(A * B) * (A * B)) == round.((A * B) * inv(A * B)) == I = true
round.(inv(A * B)) == round.(inv(B) * inv(A)) = true
```

3. Αν ο πίνακας $\mathrm{A} \in \mathbb{R}^{n \times n}$ αντιστρέψιμος τότε: $(kA)^{-1} = \frac{1}{k}A^{-1}, \forall k \in \mathbb{R}$ (ή $\mathbb{C}$), $k \neq 0$

In [86]:
```
using LinearAlgebra
A = 123*rand(123,123) .+ 123
k = 123*rand()

round.(inv(k*A)) == round.(1/k * inv(A))
```

Out[86]: true

Ορίζουσα Πίνακα

- Μπορούμε να υπολογίσουμε την Ορίζουσα ενός πίνακα $\mathrm{A} \in \mathbb{R}^{n \times n}$ μέσω της συνάρτησης `det`.

In [87]:
```
using LinearAlgebra

#Ορίζουσα 2x2 πίνακα
@show det([1 2; 3 4])

#Ορίζουσα 12x12 πίνακα
A = 32*rand(12, 12) .+ 23
@show det(A);
```

```
det([1 2; 3 4]) = -2.0
det(A) = 8.537990237090564e15
```

Ιδιότητες Οριζουσών

1. $det(A) = det(A^t)$

In [88]:
```
using LinearAlgebra

A = 12*rand(999,999) .+ 12323
det(A) == det(A')
```

Out[88]: true

2. $det(aA) = a^n det(A)$

In [89]:
```
using LinearAlgebra

A = 12*rand(999,999) .+ 12323
a = 123232323
```

```
det(a*A) == (a^999)*det(A)
```

Out[89]: true

3. $det(AB) = det(A)det(B)$

In [90]:
```
using LinearAlgebra

A = 12*rand(999,999) .+ 12323
B = 332*rand(999,999) .+ 345

det(A*B) == det(A)*det(B)
```

Out[90]: true

4. Εάν δύο γραμμές ή στήλες του A είναι ίδιες $\Rightarrow det(A) = 0$

In [91]:
```
#=
1η και 4η γραμμή του πίνακα A είναι ίδιες
(ή μέσω της συνάρτησης repeat: A = repeat([1 2; 3 4], 2, 2))
=#
using LinearAlgebra

A = [1 2 3 4 5; 3 4 0 9 3; 1 4 0 5 9; 1 2 3 4 5; 3 0 9 8 2]
det(A)
```

Out[91]: 0.0

5. Εάν ο B προκύπτει από εναλλαγή δύο γραμμών ή στηλών του $\mathrm{A} \Rightarrow det(B) = -det(A)$

In [92]:
```
#Εναλλαγή 1ης και 3ης γραμμής (ή μέσω της συνάρτησης view: B = view(A, :, [3,2,1,4,5]))
using LinearAlgebra

A = [1 2 3 4 5; 3 4 0 9 3; 1 4 0 5 9; 0 2 9 3 4; 3 0 9 8 2]
B = [1 4 0 5 9; 3 4 0 9 3; 1 2 3 4 5; 0 2 9 3 4; 3 0 9 8 2]

@show det(A)
@show det(B);
```

```
det(A) = 95.99999999999967
det(B) = -95.99999999999967
```

6. Η ορίζουσα άνω ή κάτω τριγωνικού πίνακα ισούται με το γινόμενο των διαγώνιων στοιχείων του.

In [93]:
```
#=
Η συνάρτηση diag επιστρέφει σε μορφή διανύσματος τα διαγώνια στοιχεία του πίνακα A.
Η συνάρτηση prod επιστρέφει το γινόμενο των στοιχείων του διανύσματος diag(A).
=#

using LinearAlgebra

A = UpperTriangular(234*rand(123,123) .+ 112321)
det(A) == prod(diag(A))
```

Out[93]: true

7. $det(A^{-1}) = (det(A))^{-1}$

In [94]:
```
using LinearAlgebra

A = 12*rand(999,999) .+ 12323

det(inv(A)) == det(A)^-1
```

Out[94]: true

8. $det(A) \neq 0 \Rightarrow A\ αντιστρέψιμος$

In [95]:
```
using LinearAlgebra

A = 12*rand(999,999) .+ 12323

round.(A*inv(A)) == round.(inv(A)*A) == I
```

Out[95]: true

Υποπίνακες και Διαμερίσεις Πινάκων

Υποπίνακες: `όνομα_πίνακα(k:l, m:n)`, όπου k,l και m,n δείκτες για τις γράμμες και για τις στήλες αντίστοιχα.

In [96]:
```
#A[:, 2:4] υποπίνακας
A = [1 2 3 4; 5 6 7 8; 9 0 1 2; 3 4 5 6]
A[:, 2:4]
```

Out[96]:
```
4×3 Array{Int64,2}:
 2  3  4
 6  7  8
 0  1  2
 4  5  6
```

Διαμερίσεις

1. Ένας απλός τρόπος δημιουργίας Πίνακα Block $\mathrm{X} = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$ είναι να ορίσουμε τα block A, B, C, D και έπειτα να τα εισάγουμε σε ένα πίνακα X.

In [97]:
```
#Παράδειγμα δημιουργίας Πίνακα Block
using LinearAlgebra

A = rand(4,4)
B = zeros(4,2)
C = I
D = ones(4,2)
X = [A B; C D]
```

Out[97]:
```
8×6 Array{Float64,2}:
 0.742689  0.745612  0.537609   0.70032   0.0  0.0
 0.221108  0.465434  0.172277   0.725332  0.0  0.0
 0.479425  0.579938  0.0168729  0.451202  0.0  0.0
 0.981194  0.287095  0.646325   0.273321  0.0  0.0
```

```
1.0       0.0       0.0       0.0       1.0  1.0
0.0       1.0       0.0       0.0       1.0  1.0
0.0       0.0       1.0       0.0       1.0  1.0
0.0       0.0       0.0       1.0       1.0  1.0
```

2. Μέσω του πακέτου `BlockArrays`

In [98]:
```
#Χρειάζεται να κατεβάσετε το πακέτο BlockArrays
#Copy-Paste τις παρακάτω εντολές

import Pkg; Pkg.add("BlockArrays")
```

```
   Updating registry at `C:\Users\Anastasia-Efterpi\.julia\registries\General`
   Updating registry at `C:\Users\Anastasia-Efterpi\.julia\registries\JuliaComputingRegistry`
  Resolving package versions...
No Changes to `C:\Users\Anastasia-Efterpi\.julia\environments\v1.5\Project.toml`
No Changes to `C:\Users\Anastasia-Efterpi\.julia\environments\v1.5\Manifest.toml`
```

- Για τη δημιουργία ενός πίνακα Block χρησιμοποιούμε τη συνάρτηση `BlockArray`.

In [99]:
```
#=
Στο παράδειγμα που ακολουθεί έχουμε έναν πίνακα Block 5 x 6 με τυχαία στοιχεία στο (0,1) - rand.
Το [3,2] και το [3,1,2] αναφέρονται στις διαστάσεις των block.
Πιο συγκεκριμένα το [3,2] αναφέρεται στις γραμμές και το [3,1,2] στις στήλες.
Δηλαδή έχουμε έναν πίνακα block [3,2] x [3,1,2].
=#

using BlockArrays

BlockArray(rand(5,6), [3, 2], [3,1,2])
```

Out[99]:
```
2×3-blocked 5×6 BlockArray{Float64,2}:
 0.349566  0.336245  0.563108  │  0.130336  │  0.70523   0.48214
 0.900588  0.805724  0.177269  │  0.415726  │  0.678865  0.616177
 0.748778  0.145427  0.656565  │  0.0875394 │  0.9792    0.830488
──────────────────────────────┼────────────┼──────────────────────
 0.753973  0.630478  0.124616  │  0.968048  │  0.360319  0.456504
 0.958075  0.999137  0.188414  │  0.16826   │  0.851502  0.115523
```

Αρχικοποίηση πίνακα block

- Δομή: `BlockArray(undef, block_type, block_sizes...)`
  - `block_type` = Matrix{Float64}, Array{Int32,2}, ...
  - `block_sizes` = i, j, όπου i, j διανύσματα ακεραίων πχ. $i = [i_1, i_2, i_3, \dots]$, $j = [j_1, j_2, j_3, \dots]$, όπου $i_m \in \mathbb{N}$, $j_n \in \mathbb{N}$

In [100…
```
#=
Έχετε στο μυαλό σας την εξής απλή δομή:
BlockArray{Type}(undef_blocks, [i₁, i₂,...], [j₁, j₂,...])
=#

using BlockArrays

BlockArray{Float64}(undef_blocks, [1,2], [3,2])
```

```
# ή BlockArray(undef_blocks, Array{Float64,2}, [1,2], [3,2])
```

Out[100…
```
2×2-blocked 3×5 BlockArray{Float64,2}:
 #undef  #undef  #undef  │  #undef  #undef
 ─────────────────────────┼─────────────────
 #undef  #undef  #undef  │  #undef  #undef
 #undef  #undef  #undef  │  #undef  #undef
```

Τοποθέτηση στοιχείων στον πίνακα block

- Μέσω της συνάρτησης `setblock!(block_array,v, i...)`, όπου `v` είναι ο πίνακας και `i` ο δείκτης του block.
- Άλλοι τρόποι είναι ο εξής: `block_array[Block(i...)] = v` ή `block_array[Block.(i)...]`

In [101…
```
using BlockArrays

X = BlockArray{Float64}(undef_blocks, [1,2], [3,2])

@show setblock!(X, ones(2,2), 2, 2)
@show X[Block(2,2)] = ones(2,2)

X
```

```
setblock!(X, ones(2, 2), 2, 2) = [1.0 1.0; 1.0 1.0]
X[Block(2, 2)] = ones(2, 2) = [1.0 1.0; 1.0 1.0]
```

Out[101…
```
2×2-blocked 3×5 BlockArray{Float64,2}:
 #undef  #undef  #undef  │  #undef  #undef
 ─────────────────────────┼─────────────────
 #undef  #undef  #undef  │    1.0     1.0
 #undef  #undef  #undef  │    1.0     1.0
```

Λήψη πληροφορίας από τον πίνακα block

- Μέσω της συνάρτησης `getblock(block_array, i...)`, όπου `i` δείκτης του block.
- Άλλος τρόπος είναι ο εξής: `block_array[Block(i...)]`

In [102…
```
#Πλήρες παράδειγμα δημιουργίας πίνακα block.
using BlockArrays

X = BlockArray{Float64}(undef_blocks, [1,2], [3,2])
@show X[Block(1,1)] = [23.4 45.3 819.3211]
@show X[Block(1,2)] = zeros(1,2)
@show X[Block(2,1)] = round.(10*rand(2,3)) .+ 3
@show X[Block(2,2)] = ones(2,2)

X
```

```
X[Block(1, 1)] = [23.4 45.3 819.3211] = [23.4 45.3 819.3211]
X[Block(1, 2)] = zeros(1, 2) = [0.0 0.0]
X[Block(2, 1)] = round.(10 * rand(2, 3)) .+ 3 = [13.0 5.0 5.0; 7.0 4.0 8.0]
X[Block(2, 2)] = ones(2, 2) = [1.0 1.0; 1.0 1.0]
```

Out[102…
```
2×2-blocked 3×5 BlockArray{Float64,2}:
 23.4  45.3  819.321  │  0.0  0.0
──────────────────────┼──────────
 13.0   5.0    5.0    │  1.0  1.0
  7.0   4.0    8.0    │  1.0  1.0
```

In [103…
```
#Λήψη του block X₂₁

@show getblock(X,2,1)
@show X[Block(2,1)];
```

```
getblock(X, 2, 1) = [13.0 5.0 5.0; 7.0 4.0 8.0]
X[Block(2, 1)] = [13.0 5.0 5.0; 7.0 4.0 8.0]
```

Παραδείγματα με Block Πινάκων

- Υπολογισμός αντιστρόφου

In [130…
```
n = 1231
m = 3213
A11 = rand(n,n)
A12 = rand(n,m)
A21 = zeros(m,n)
A22 = rand(m,m)

A = [A11 A12; A21 A22]

@time A⁻¹ = inv(A)

A11⁻¹ = inv(A11)
A22⁻¹ = inv(A22)
@time A⁻¹ = [A11⁻¹ -A11⁻¹*A12*A22⁻¹; A21 A22⁻¹];
```

```
  2.263950 seconds (6 allocations: 152.878 MiB, 1.24% gc time)
  0.511404 seconds (69 allocations: 222.589 MiB, 18.97% gc time)
```

In [105…
```
#Εφαρμογή 1
using LinearAlgebra
n = 1231
m = 3213
A11 = rand(n,n)
A12 = rand(n,m)
A21 = rand(m,n)
A22 = rand(m,m)

A = [A11 A12; A21 A22]

@time A⁻¹ = inv(A)

A11⁻¹ = inv(A11)
O1 = zeros(m,n)
O2 = zeros(n,m)
D = inv(A22-A21*A11⁻¹*A12)
@time A⁻¹ = [I -A11⁻¹*A12; O1 I]*[A11⁻¹ O2; O1 D]*[I O2; -A21*A11⁻¹ I];
```

```
  2.355667 seconds (6 allocations: 152.878 MiB, 3.16% gc time)
  3.797038 seconds (437.92 k allocations: 901.661 MiB, 5.73% gc time)
```

In [106…

```
#Εφαρμογή 2
using LinearAlgebra
n = 1231
m = 3213
A11 = rand(n,n)
A12 = rand(n,m)
A21 = rand(m,n)
A22 = rand(m,m)

A = [A11 A12; A21 A22]

@time A⁻¹ = inv(A)

A22⁻¹ = inv(A22)
O1 = zeros(m,n)
O2 = zeros(n,m)
D = inv(A11-A12*A22⁻¹*A21)
@time A⁻¹ = [I O2; -A22⁻¹*A21 I]*[D O2; O1 A22⁻¹]*[I -A12*A22⁻¹; O1 I];
```

```
  2.476351 seconds (6 allocations: 152.878 MiB, 5.00% gc time)
  3.891535 seconds (257 allocations: 945.245 MiB, 4.59% gc time)
```

- Υπολογισμός ορίζουσας

In [129…

```
#Εφαρμογή 3
using LinearAlgebra
n = 1231
m = 3213
A11 = rand(n,n)
A12 = rand(n,m)
A21 = rand(m,n)
A22 = rand(m,m)

A = [A11 A12; A21 A22]

@time det(A)

A11⁻¹ = inv(A11)
@time det(A11)*det(A22 - A21*A11⁻¹*A12);
```

```
  0.695820 seconds (5 allocations: 150.708 MiB, 4.13% gc time)
  0.914006 seconds (16 allocations: 278.055 MiB, 22.31% gc time)
```

Σημείωση: Παρατηρούμε ότι ένας κομψός μαθηματικός τύπος υλοποιούμενος υπολογιστικά, μπορεί να μην επιφέρει την πιο αποτελεσματική εκτέλεση.

Ορθογώνιοι Πίνακες

- Ένας απλός τρόπος δημιουργίας ενός τυχαίου ορθογώνιου πίνακα είναι μέσω της QR παραγοντοποίησης πίνακα που υλοποιείται με την συνάρτηση `qr`.
  Ουσιαστικά η `qr` δέχεται έναν πίνακα $\mathrm{A} \in \mathbb{R}^{m \times n}$ και επιστρέφει έναν ορθογώνιο πίνακα `Q` και έναν άνω τριγωνικό πίνακα `R`.

In [108…
```
using LinearAlgebra

A = 10*(rand(5,4)) .+ 3
Q,R = qr(A)

#Ο ορθογώνιος πίνακας
Q
```

Out[108…
```
5×5 LinearAlgebra.QRCompactWYQ{Float64,Array{Float64,2}}:
 -0.530115   0.494278    0.0137522  -0.170945  -0.667275
 -0.197853  -0.684416   -0.136678    0.494093  -0.479187
 -0.575317   0.0918422  -0.641932    0.172763   0.467602
 -0.320767  -0.527642    0.0490958  -0.782313   0.0654141
 -0.495934  -0.0205652   0.752757    0.291187   0.319677
```

In [109…
```
#Ο άνω τριγωνικός πίνακας
R
```

Out[109…
```
4×4 Array{Float64,2}:
 -19.6942  -22.3725  -14.7998   -14.7324
   0.0     -10.6204   -8.66471   -5.08377
   0.0       0.0       5.71564   -0.107809
   0.0       0.0       0.0        5.22817
```

Σημείωση (1): Λόγω των σφαλμάτων στρογγύλευσης το γινόμενο QR δεν ισούται με τον A.

In [110…
```
@show A
println()
@show Q*R

A == Q*R
```

```
A = [10.440211861269354 6.61058553871734 3.6414141107646887 4.401887427574513; 3.896557469693497
5 11.695203904060758 8.077244256087962 8.99220909435301; 11.330420181531604 11.895866894129796
4.049721162271851 8.981356182280507; 6.317257424003632 12.780099346829779 9.599758422454457 3.31
27376389018233; 9.76704337798803 11.313691261906905 11.820394647988795 8.852092814018935]

Q * R = [10.440211861269358 6.610585538717338 3.6414141107646856 4.4018874275745095; 3.896557469
693498 11.695203904060762 8.077244256087962 8.99220909435301; 11.330420181531604 11.895866894129
796 4.04972116227185 8.981356182280505; 6.317257424003633 12.780099346829779 9.599758422454457
3.3127376389018233; 9.767043377988031 11.313691261906905 11.820394647988794 8.852092814018935]
```

Out[110…
```
false
```

Σημείωση (2): Τα γινόμενα Q^TQ και QQ^T έχουν απώλεια ορθογωνιότητας και για αυτό διαφέρουν από τον I.

In [111…
```
using LinearAlgebra

A = 10*(rand(3,3)) .+ 3
Q,R = qr(A)

@show Q'*Q
@show Q*Q'
println()
@show Q'*Q == Q*Q' == I

#Με rounding ωστόσο ισχύει.
@show round.(Q'*Q) == round.(Q*Q') == I;
```

```
Q' * Q = [1.0000000000000002 1.1102230246251565e-16 1.3877787807814457e-17; 1.1102230246251565e-16 0.9999999999999998 -8.326672684688674e-17; 1.3877787807814457e-17 -8.326672684688674e-17 0.9999999999999998]
Q * Q' = [0.9999999999999998 1.6653345369377348e-16 1.8041124150158794e-16; 1.6653345369377348e-16 1.0 1.1102230246251565e-16; 1.8041124150158794e-16 1.1102230246251565e-16 1.0]

Q' * Q == Q * Q' == I = false
round.(Q' * Q) == round.(Q * Q') == I = true
```

In [112…
```
#Παράδειγμα "τέλειων" ορθογώνιων πινάκων.
using LinearAlgebra

U = 1/2 * [1+im 1-im; 1-im 1+im]
U' * U == U * U' == I

#ή U = [3//5 -4//5; im*4//5 im*3//5]
```

Out[112… true

Ιδιότητες ορθογώνιου πίνακα

1. $U^{-1} = U^T$

In [113…
```
U = [3//5 -4//5; im*4//5 im*3//5]

inv(U) == U'
```

Out[113… true

In [114…
```
U = 1/2 * [1+im 1-im; 1-im 1+im]
inv(U) == U'
```

Out[114… true

2. Γινόμενο ορθογωνίων = ορθογώνιος

In [115…
```
using LinearAlgebra

A = 10*(rand(12,34)) .+ 3
Q1,R = qr(A)

A = 10*(rand(12,34)) .+ 3
Q2,R = qr(A)

round.((Q1*Q2)' * (Q1*Q2)) == round.((Q1*Q2) * (Q1*Q2)') == I
```

Out[115… true

3. $det(U^TU) = det(I) = 1 \Rightarrow (detU)^2 = 1 \Rightarrow detU = \pm 1$

In [116…
```
using LinearAlgebra

A = 10*(rand(12,34)) .+ 3
Q,R = qr(A)

@show round(det(Q*Q')) == det(I)
@show det(Q)^2 == 1
@show det(Q);
```

```
round(det(Q * Q')) == det(I) = true
det(Q) ^ 2 == 1 = true
det(Q) = -1.0
```

Trace (Ίχνος) Πίνακα

- Μπορούμε να υπολογίσουμε το ίχνος ενός πίνακα $\mathrm{A} \in \mathbb{R}^{n \times n}$ μέσω της συνάρτησης `tr`.

In [117…
```
using LinearAlgebra

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

tr(A)
```

Out[117… 15

Ιδιότητα ίχνους πίνακα:

- $tr(AA^T) = tr(A^TA)$

In [118…
```
using LinearAlgebra

A = round.(12*rand(34,34) .+ 56)

tr(A*A') == tr(A'*A)
```

Out[118… true

Rank (Τάξη) Πίνακα

- Μπορούμε να υπολογίσουμε την τάξη ενός πίνακα $\mathrm{A} \in \mathbb{R}^{n \times n}$ μέσω της συνάρτησης `rank`.

In [119…
```
#Full rank
using LinearAlgebra

A = [1 2; 3 4; 5 6]

@show rank(A);
```

```
rank(A) = 2
```

In [120…
```
#Rank deficient
using LinearAlgebra

A = [1 2; 2 4; 0 0]

@show rank(A);
```

```
rank(A) = 1
```

Ιδιότητες τάξεως πίνακα:

Έστω πίνακας $\mathrm{A} \in \mathbb{R}^{m \times n}$:

1. $rank(A) = rank(A^T)$

In [121…
```
using LinearAlgebra
A = 123*rand(12,34) .+ 345

rank(A) == rank(A')
```

Out[121…
```
true
```

2. $rank(A) + nullspace(A) = n$ $\Rightarrow$ full rank πίνακα $\Rightarrow$ δεν υπάρχει(undefined) nullspace

In [122…
```
#undefined nullspace. (nullspace = υπόχωρος Rⁿ των λύσεων της εξίσωσης Ax = 0)
using LinearAlgebra
A = [1 2; 3 4; 5 6]

@show nullspace(A)
@show length(nullspace(A))

rank(A) + length(nullspace(A)) == size(A,2)
```

```
nullspace(A) = Array{Float64}(undef,2,0)
length(nullspace(A)) = 0
```

Out[122…
```
true
```

In [123…

```
#defined nullspace
using LinearAlgebra
A = [1 2; 2 4; 0 0]

@show nullspace(A)
@show length(nullspace(A))

rank(A) + length(nullspace(A)) == size(A,2)
```

```
nullspace(A) = [-0.894427190999916; 0.447213595499958]
length(nullspace(A)) = 2
```

Out[123… false

3. $rank(AB) \geq rank(A) + rank(B) - n,$ με $B \in \mathbb{R}^{n\times p}$.

In [124…

```
using LinearAlgebra

A = 123*rand(12,34) .+ 345
B = 123*rand(34,23) .+ 345

rank(A*B) ≥ rank(A) + rank(B) - size(A,2)
```

Out[124… true

4. $rank(AB) = rank(A) = rank(AC),$ με $A, B, C \in \mathbb{R}^{n\times n}$, B, C μη ιδιάζοντες.

In [125…

```
using LinearAlgebra

A = 123*rand(45,45) .+ 345
B = 123*rand(45,45) .+ 345
C = 123*rand(45,45) .+ 345

rank(A*B) == rank(A) == rank(A*C)
```

Out[125… true

In [126…

```
using LinearAlgebra

A = [1 2; 3 4]
B = [1 2; 1 2]
C = [2 3; 0 0]

@show A*B
@show rank(A*B)
@show A*C
@show rank(A*C)

rank(A*B) == rank(A) == rank(A*C)
```

```
A * B = [3 6; 7 14]
rank(A * B) = 1
A * C = [2 3; 6 9]
rank(A * C) = 1
```

Out[126… false

5. $rank(AB) \leq min\{rank(A), rank(B)\}$, με $B \in \mathbb{R}^{n \times p}$.

In [127…
```
using LinearAlgebra

A = 123*rand(123,45) .+ 345
B = 123*rand(45,4533) .+ 345

rank(A*B) ≤ min(rank(A), rank(B))
```

Out[127… true

6. $rank(A + B) \leq rank(A) + rank(B)$, με $B \in \mathbb{R}^{m \times n}$

In [128…
```
using LinearAlgebra

A = 123*rand(123,45) .+ 345
B = 123*rand(123,45) .+ 345

rank(A + B) ≤ rank(A) + rank(B)
```

Out[128… true