Κεφάλαιο 2ο: Λίστες - Πολυώνυμα

2.1 Λίστες

Οι λίστες παίζουν σημαντικό ρόλο στη χρήση συναρτήσεων του *Mathematica*. Συχνά, οι απαντήσεις που δίνει το πρόγραμμα κατά την εκτέλεση συναρτήσεων έχουν τη μορφή λίστας. Ως προς τον συμβολισμό, η λίστα μοίαζει με ένα πεπερασμένο σύνολο, δηλαδή έχει τη μορφή {1, 2, 3, a, b, 4, 5}. Όπως βλέπουμε τα μέλη της λίστα μπορεί να είναι είτε αριθμοί είτε γράμματα. Στην πραγματικότητα μέλη της λίστας μπορεί να είναι οποιαδήποτε αντικείμενα, ακόμη και συναρτήσεις του *Mathematica* ή και άλλες λίστες. Για παράδειγμα, η λίστα {1, 4, Plus[x,y], {a,b}} είναι αποδεκτή, και τα μέλη της είναι οι αριθμοί 1 και 4, η συνάρτηση Plus[x, y] και η λίστα {a, b}.

```
81, 4, Plus@x, yD, 8a, b<<

81, 4, x + y, 8a, b<<
```

Μια λίστα, ανεξάρτητα από τη μορφή με την οποία εμφανίζεται στο παράθυρο εργασίας, είναι και αυτή μία συνάρτηση. Αυτό σημαίνει ότι, ενώ ο χρήστης βλέπει τη λίστα με τον τρόπο που περιγράψαμε, το *Mathematica* αναγνωρίζει τη λίστα με διαφορετική μορφή. Πράγματι αν χρησιμοποιήσουμε τη συνάρτηση **FullForm** έχουμε:

```
FullForm@%D

List@1, 4, Plus@x, yD, List@a, bDD
```

Παρατηρούμε ότι η πλήρη μορφή μιας λίστας δεν διαφέρει πολύ από την πλήρη μορφή ενός αθροίσματος ή γινομένου. Η μόνη διαφορά που υπάρχει είναι η διαφορετική επικεφαλίδα, δηλαδή το πρόθεμα List, Plus και Times που έχει η λίστα, το άθροισμα και το γινόμενο αντίστοιχα. Αυτό σημαίνει ότι αν μπορέσουμε να αλλάξουμε την επικεφαλίδα List σε Plus, τότε αντί της λίστας θα πάρουμε άθροισμα.

Το *Mathematica* διαθέτει τη συνάρτηση **Head** με την οποία βρίσκει την επικεφαλίδα μιας παράστασης:

Head[expr]: επιστρέφει την επικεφαλίδα της παράστασης expr

```
Head@%D

List
```

```
Head@x + yD

Plus
```

Η συνάρτηση **Head** μας επιστρέφει την επικεφαλίδα, δηλαδή το πρόθεμα της παράστασης, ακόμα και για πολύπλοκες παραστάσεις:

```
FullForm@Hx + yL^zD

Power@Plus@x, yD, zD
```

```
Head@%D

Power
```

Το *Mathematica* διαθέτει τη συνάρτηση **Apply** με την οποία αντικαθιστά την επικεφαλίδα μιας παράστασης:

Apply[f, expr]: αντικαθιστά την επικεφαλίδα της παράστασης expr με την επικεφαλίδα f

Έστω μία λίστα με αριθμούς:

```
810, 20, 30<

810, 20, 30<
```

Η πλήρης μορφή της λίστας είναι:

```
FullForm@%D

List@10, 20, 30D
```

Χρησιμοποιώντας την συνάρτηση **Apply**, αλλάζουμε την επικεφαλίδα της λίστας από List σε Plus και Times:

```
Apply@Plus, %D

60
```

```
Apply@Times, %8D

6000
```

Παρατηρούμε ότι τα αποτελέσματα είναι το άθροισμα και το γινόμενο των μελών της λίστας.

Το *Mathematica* είναι εφοδιασμένο με διάφορες συναρτήσεις, με τις οποίες μπορούμε εύκολα να χειριστούμε τις λίστες. Η συνάρτηση **Part** είναι χρήσιμη όταν θέλουμε να αναφερθούμε σε κάποιο μέλος μιας λίστας. Ο συμβολισμός [[ ]] αποτελεί συντομογραφία της συνάρτησης **Part**.

Part[expr, n]:	επιστρέφει το n-οστό στη σειρά μέλος της παράστασης expr
expr[[n]]:	είναι ισοδύναμη με την παράσταση Part[expr, n]
Part[expr, -n]:	επιστρέφει το n-οστό στη σειρά μέλος της παράστασης expr μετρώντας από το τέλος
expr[[-n]]:	είναι ισοδύναμη με την παράσταση Part[expr, -n]
Part[expr, 0]:	επιστρέφει την επικεφαλίδα της παράστασης expr
expr[[0]]:	είναι ισοδύναμη με την παράσταση Part[0]
Part[expr, $\{i_1, i_2, ..., i_k\}$]:	επιστρέφει μια λίστα της οποίας τα στοιχεία είναι τα $i_1, i_2, ..., i_k$ μέλη της παράστασης expr
expr[[$\{i_1, i_2, ..., i_k\}$]]:	είναι ισοδύναμη με την παράσταση Part[expr, $\{i_1, i_2, ..., i_k\}$]

Έστω η λίστα με όνομα a:

```
a = 81, 2, 10, x, y, 84, 5, 6<, 810, 20, 30<<

81, 2, 10, x, y, 84, 5, 6<, 810, 20, 30<<
```

Παρατηρούμε ότι μέλη της λίστας είναι αριθμοί, γράμματα αλλά και άλλες λίστες.

Στη συνέχεια βλέπουμε μερικές εφαρμογές της συνάρτησης **Part** και του ισοδύναμου συμβολισμού [[ ]]:

```
Part@a, 3D

10
```

```
a@@3DD

10
```

Παρατηρούμε ότι επιστρέφεται ως αποτέλεσμα το τρίτο μέλος της λίστας που είναι ο αριθμός 3.

```
Part@a, −2D

84, 5, 6<
```

```
a@@−2DD

84, 5, 6<
```

Παρατηρούμε ότι επιστρέφεται ως αποτέλεσμα το δεύτερο μέλος της λίστας, μετρώντας από το τέλος, που είναι η λίστα {4, 5, 6}.

```
Part@a, 0D

List
```

```
a@@0DD

List
```

Παρατηρούμε ότι επιστρέφεται ως αποτέλεσμα η επικεφαλίδα της παράσταση a, η οποία είναι List αφού η παράσταση a είναι μια λίστα.

Μπορούμε ακόμη, χρησιμοποιώντας την τελευταία επιλογή της συνάρτησης **Part**, να σχηματίσουμε μια άλλη λίστα επιλέγοντας συγκεκριμένα στοιχεία της λίστας a:

```
Part@a, 81, 3, 5<D

81, 10, y<
```

```
a@@81, 3, 5<DD

81, 10, y<
```

Παρατηρούμε ότι επιστρέφεται ως αποτέλεσμα μια λίστα με το πρώτο, τρίτο και πέμπτο μέλος της λίστα a.

Επίσης μπορούμε να σχηματίσουμε μια άλλη λίστα επιλέγοντας συγκεκριμένα στοιχεία της λίστας a με την εξής εντολή:

```
8Part@a, 3D, Part@a, 5D, Part@a, 6D<
```

```
810, y, 84, 5, 6<<
```

```
8a@@3DD, a@@5DD, a@@6DD<
```

```
810, y, 84, 5, 6<<
```

Παρατηρούμε ότι επιστρέφεται ως αποτέλεσμα μια λίστα με το τρίτο, πέμπτο και έκτο μέλος της λίστα a.

Η λίστα {4, 5, 6} θεωρείται μέλος της λίστας a, μάλιστα είναι το έκτο μέλος της λίστας α, και για αυτό μπορούμε να αναφερθούμε σε αυτή. Ο αριθμός 5, όμως, δεν είναι μέλος της λίστας a, επομένως δεν μπορούμε να αναφερθούμε σε αυτόν με τον τρόπο που περιγράψαμε πιο πάνω. Παρατηροούμε, ότι ο αριθμός 5 είναι μέλος μιας επιμέρους λίστας, η οποία είναι το έκτο μέλος της λίστας a, ενώ ο αριθμός 5 είναι το δεύτερο μέλος της επιμέρους λίστας. Έτσι, χρησιμοποιώντας, δύο δείκτες μπορούμε να αναφερθούμε στον αριθμό 5. Με τις επόμενες εντολές ζητάμε από το *Mathematica* να μας παρουσιάσει από το έκτο μέλος της λίστας a, το δεύτερο μέλος.

```
Part@a, 6, 2D
```

```
5
```

```
a@@6, 2DD
```

```
5
```

2.2 Πίνακες και Λίστες

Όπως είδαμε παραπάνω, μπορούμε να σχηματίσουμε μια άλλη λίστα επιλέγοντας συγκεκριμένα στοιχεία της λίστας a:

```
Part@a, 86, 7<D
```

```
884, 5, 6<, 810, 20, 30<<
```

```
a@@86, 7<DD

884, 5, 6<, 810, 20, 30<<
```

Παρατηρούμε ότι μπορούμε να έχουμε λίστες, των οποίων τα μέλη είναι αποκλειστικά λίστες. Στην πραγματικότητα ένας mxn πίνακας ορίζεται σαν μια λίστα, η οποία αποτελείται από m επιμέρους λίστες, τις γραμμές του πίνακα, καθεμία από τις οποίες περιέχει n στοιχεία. Το *Mathematica* διαθέτει την συνάρτηση **MatrixForm** με την οποία δείχνει τον πίνακα στη συνήθη μορφή του:

MatrixForm[list]: εμφανίζει τα στοιχεία της λίστας list ως πίνακα με τη συνήθη μορφή.

```
MatrixForm@%D

J 4   5   6 N
  10  20  30
```

Ένα από τα μειονεκτήματα που έχουν οι λίστες είναι ότι για να εισαχθούν στο *Mathematica* πρέπει να αναγραφούν τα στοιχεία τους ένα προς ένα. Για το λόγο αυτό το πρόγραμμα είναι εφοδιασμένο με κάποιες συναρτήσεις, οι οποίες βοηθούν στην κατασκευή λιστών και κατ' επέκταση πινάκων. Μια από αυτές τις συναρτήσεις είναι η συνάρτη **Range**:

Range[imax]: επιστρέφει τη λίστα {1, 2, 3, ..., imax}.

Range[imin,imax]: επιστρέφει τη λίστα {imin, imin+1, imin+2, ..., imaz-1, imax}.

Range[imin,imax,step]: επιστρέφει τη λίστα {imin, imin+step, imin+2*step, ..., **§** imax}.

Είναι προφανές ότι στις δύο πρώτες περιπτώσεις η συνάρτηση **Range** δημιουγεί μια λίστα με διαδοχικούς ακεραίους,

```
Range@10D

81, 2, 3, 4, 5, 6, 7, 8, 9, 10<
```

```
Range@3, 9D

83, 4, 5, 6, 7, 8, 9<
```

ενώ στην τρίτη περίπτωση ελέγχουμε το βήμα αύξησης step των στοιχείων της λίστας. Στην τελευταία περίπτωση είναι δυνατόν το άνω όριο imax να μην υπάρχει στην λίστα, διότι δεν το επιτρέπει το βήμα που καθορίσαμε. Για παράδειγμα, στη λίστα:

```
Range@1, 20, 4D

81, 5, 9, 13, 17<
```

δεν περιέχεται το άνω όριο, αφού το άθροισμα 17 + 4 ξεπερνά το όριο αυτό. Επομένως πρέπει να είμαστε προσεκτικοί, όταν καθορίζουμε το βήμα αύξησης, και θέλουμε οπωσδήποτε το άνω όριο να περιέχεται στη λίστα που θα δημιουργηθεί.

Η συνάρτηση **Range** μπορεί να χρησιμοποιηθεί και για την κατασκευή πινάκων:

```
8Range@4D, Range@7, 10D, Range@10, 20, 3D<

881, 2, 3, 4<, 87, 8, 9, 10<, 810, 13, 16, 19<<
```

Παρατηρούμε

```
MatrixForm@%D

 1   2   3   4
 7   8   9  10
10  13  16  19
```

Μια άλλη συνάρτηση σχηματισμού λιστών και κατ' επέκταση πινάκων είναι η **Table**:

Table[expr, {imax}]: σχηματίζει μία λίστα με imax αντίγραφα της παράστασης expr.

Table[expr, {i, imax}]: σχηματίζει μία λίστα από τις τιμές της παράστασης expr, όταν το i παίρνει τιμές από 1 μέχρι imax.

Table[expr, {i, imin, imax}]: σχηματίζει μία λίστα από τις τιμές της παράστασης expr, όταν το i παίρνει τιμές από imin μέχρι imax.

Table[expr, {i, imin, imax, step}]: σχηματίζει μία λίστα από τις τιμές της παράστασης expr, όταν το i παίρνει τιμές imin, imin+step, imin+2*step, ..., **§** imax.

Table[expr, {i, imin, imax}, {j, jmin, jmax}]: σχηματίζει μία λίστα της οποίας τα στοιχεία είναι επιμέρους λίστες. Το πλήθος των επιμέρους λιστών προσδιορίζεται από το πλήθος των τιμών του δείκτη i. Ο δείκτης j καθορίζει το πλήθος των στοιχείων των επιμέρους λιστών.

Ανεξάρτητα από την πολυπλοκότητα της σύνταξης της συνάρτησης **Table**, η χρήση της είναι απλή:

```
Table@xyz, 85<D

8xyz, xyz, xyz, xyz, xyz<
```

```
Table@2^i, 8i, 8<D

82, 4, 8, 16, 32, 64, 128, 256<
```

```
Table@10 - i, 8i, 0, 10<D

810, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0<
```

```
Table@x^i, 8i, 2, 10, 2<D

8x^2, x^4, x^6, x^8, x^10<
```

Η συνάρτηση **Table** είναι ιδανική για την κατασκευή πινάκων. Η παρακάτω εντολή σχηματίζει έναν 3x4 πίνακα:

```
Table@i^j, 8i, 3<, 8j, 4<D

881, 1, 1, 1<, 82, 4, 8, 16<, 83, 9, 27, 81<<
```

Ας δούμε με ποιο τρόπο εκτελείται η παραπάνω εντολή. Η συνάρτηση **Table** αρχίζει με τον πρώτο δείκτη, δηλαδή τον δείκτη i, στον οποίο δίνει την πρώτη τιμή, δηλαδή i = 1. Αμέσως μετά πηγαίνει στον δεύτερο δείκτη, δηλαδή τον j, στον οποίο δίνει με τη σειρά όλες τις δυνατές τιμές που αντιστοιχούν στο δείκτη αυτό, δηλαδή j = 1, j = 2, j = 3 και j = 4. Αυτό σημαίνει ότι τα ζεύγη (i, j) που θα πάρουμε είναι (1, 1), (1, 2), (1, 3) και (1, 4). Τα ζεύγη αυτά τα αντικαθιστά στην παράσταση i ^ j με αποτέλεσμα τις τιμές 1^1 **=** 1, 1^2 **=** 1, 1^3 **=** 1και 1^4 **=** 1 οι οποίες αποτελούν τα στοιχεία της πρώτης επιμέρους λίστας. Στη συνέχεια, ο δείκτης i θαπάρει την επόμενη τιμή, δηλαδή i = 2, ενώ ο δείκτης j θα ξαναπάρει όλες τις δυνατές τιμές. Έτσι, σχηματίζονται τα ζεύγη (2, 1), (2, 2), (2, 3) και (2, 4). Τα ζεύγη αυτά αντιστοιχούν στις τιμές 2^1 **=** 2, 2^2 **=** 4, 2^3 **=** 8και 2^4 **=** 16 οι οποίες αποτελούν τα στοιχεία της δεύτερης επιμέρους λίστας. Η διαδικασία αυτή συνεχίζεται μέχρι να εξαντληθούν όλες οι δυνατές τιμές του πρώτου δείκτη i.

Η συνάρτηση **MatrixForm** μπορεί να μας δώσει, τώρα, τον πίνακα στη συνήθη του μορφή:

```
MatrixForm@%D

⎛ 1  1   1   1 ⎞
⎜ 2  4   8  16 ⎟
⎝ 3  9  27  81 ⎠
```

Στο επόμενο παράδειγμα φαίνεται καθαρά ο τρόπος με τον οποίο χειρίζεται τους δείκτες η συνάρτηση **Table**:

```
Table@a@i, jD, 8i, 3<, 8j, 4<D êê MatrixForm

⎛ a@1, 1D  a@1, 2D  a@1, 3D  a@1, 4D ⎞
⎜ a@2, 1D  a@2, 2D  a@2, 3D  a@2, 4D ⎟
⎝ a@3, 1D  a@3, 2D  a@3, 3D  a@3, 4D ⎠
```

Η οδηγία "// MatrixForm" στο τέλος της προηγούμενης εντολής ζητά από το μαθημάτικα να εφαρμόσει τη συνάρτηση **MatrixForm** στο αποτέλεσμα που βρεθεί. Γενικότερα, η εντολή "x // f" θα εφαρμόσει τη συνάρτηση f στο x:

```
x êê f

f@xD
```

Είναι προφανές ότι ορισμένες μορφές της συνάρτησης **Table** είναι ισοδύναμες με τη συνάρτηση **Range**.

Δημιουργία λίστας με τη συνάρτηση **Table**[expr, {i, imax}]:

```
Table@i, 8i, 10<D

81, 2, 3, 4, 5, 6, 7, 8, 9, 10<
```

Δημιουργία λίστας με τη συνάρτηση **Range**[imax]:

```
Range@10D

81, 2, 3, 4, 5, 6, 7, 8, 9, 10<
```

Έλεγχος αν οι δύο προηγούμενες εντολές είναι ισοδύναμες:

```
Table@i, 8i, 10<D « Range@10D

True
```

Ο συμβολισμός == είναι συντομογραφία της συνάρτησης **Equal**, και ταυτόχρονα αποτελεί το σύμβολο της μαθηματικής ισότητας. Η παραπάνω εντολή είναι ουσιαστικά μία ερώτηση προς το *Mathematica*, αν το αριστερό μέλος είναι ίσο με το δεξιό μέλος της ισότητας. Όπως βλέπουμε η απάντηση είναι καταφατική (True).

Ο μαθηματικός συμβολισμός ≠, ο οποίος είναι συντομογραφία της συνάρτησης **Unequal**, εισάγεται στο *Mathematica* με τον συνδυασμό !=. Όπως και στην περίπτωση της ισότητας, η εντολή x != y είναι μια ερώτηση προς το *Mathematica* αν ισχύει η μαθηματική σχέση x ≠ y.

```
Table@i, 8i, 10<D != Range@10D

False
```

Όταν συγκρίνουμε λίστες πρέπει να είμαστε προσεκτικοί, διότι το *Mathematica* δεν αντιμετωπίζει τις λίστες ως σύνολα. Για παράδειγμα, ίσως εμείς να θεωρούμε ότι οι παρακάτω λίστες, οι οποίες κατασκευάζονται με τη χρήση των συναρτήσεων **Table** και **Range**, ταυτίζονται

```
Table@10 - i, 8i, 0, 5<D

810, 9, 8, 7, 6, 5<
```

```
Range@5, 10D

85, 6, 7, 8, 9, 10<
```

όμως το *Mathematica* βλέπει τα πράγματα διαφορετικά.

```
Table@10 - i, 8i, 0, 5<D ≠ Range@5, 10D

True
```

Το παραπάνω παράδειγμα δείχνει ότι το *Mathematica* αντιμετωπίζει τις λίστες σαν διατεταγμένα σύνολα, οπότε η διάταξη των στοιχείων μιας λίστας παίζει σημαντικό ρόλο.

Οι συναρτήσεις **Range** και **Table** είναι χρήσιμες όταν θέλουμε να κατασκευάσουμε λίστες και κατ' επέκταση πίνακες, των οποίων τα στοιχεία ικανοποιούν κάποιους κανόνες. Όταν, όμως, θέλουμε λίστες με συγκεκριμένα στοιχεία, τα οποία δεν ακολουθούν κάποιο κανόνα, τότε ασφαλώς πρέπει να εισάγουμε τα

στοιχεία ένα προς ένα.

Μερικές φορές, όμως, χρειαζόμαστε μια λίστα με τυχαία στοιχεία. Δηλαδή, μία λίστα της οποίας τα μέλη δεν είναι κάποια συγκεκριμένα στοιχεία αλλά ούτε ικανοποιούν κάποια σχέση. Ο ευκολότερος τρόπος για να πάρουμε μια τέτοια λίστα είναι να χρησιμοποιήσουμε τη συνάρτηση **Random**:

Random[ ]:	επιστρέφει ένα τυχαίο πραγματικό αριθμό μεταξύ 0 και 1.
Random[type]:	επιστρέφει ένα τυχαίο αριθμό τύπου type μεταξύ 0 και 1. Ο τύπος (type) του αριθμού μπορεί να είναι: Integer, Real, Complex.
Random[type, range]:	επιστρέφει ένα τυχαίο αριθμό του συγκεκριμένου τύπου type ο οποίος βρίσκεται στο πεδίο range. Ο τύπος (type) του αριθμού μπορεί να είναι: Integer, Real, Complex. Το πεδίο (range) μπορεί να είναι της μορφής {imin, imax} ή απλά imax που είναι ισοδύναμο με το πεδίο {0, imax}.

Η συνάρτηση **Random** είναι ένα παράδειγμα, το οποίο δείχνει ότι μια συνάρτηση μπορεί να έχει από μηδέν έως περισσότερα ορίσματα. Επίσης, παρατηρούμε ότι και στην περίπτωση που δεν υπάρχει κανένα όρισμα, οι αγκύλες [ ] συνοδεύουν το όνομα της συνάρτησης.

Η εντολή που ακολουθεί είναι μία λίστα τεσσάρων εντολών. Η πρώτη θα δώσει ένα τυχαίο πραγματικό αριθμό μεταξύ 0 και 1, η δεύτερη ένα τυχαίο ακέραιο μεταξύ 0 και 1, η τρίτη ένα τυχαίο πραγματικό μεταξύ 0 και 5 και η τελευταία ένα τυχαίο ακέραιο μεταξύ 12 και 34.

```
8Random@D, Random@IntegerD,
 Random@Real, 5D, Random@Integer, 812, 34<D<

80.269114, 0, 3.44615, 27<
```

Επειδή η συνάρτηση **Random** επιστρέφει ένα τυχαίο αριθμό, είναι προφανές ότι κάθε φορά που εκτελούμε την προηγούμενη εντολή, θα παίρνουμε μια διαφορετική λίστα τεσσάρων αριθμών.

Αν η συνάρτηση **Random** συνδυαστεί με τη συνάρτηση **Table**, μπορεί να μας δώσει ένα τυχαίο πίνακα με το μέγεθος που θέλουμε. Ας υποθέσουμε, για παράδειγμα, ότι θέλουμε ένα πίνακα 3x3 με στοιχεία ακεραίους μεταξύ 5 και 15:

```
b = Table@Random@Integer, 85, 15<D, 8i, 1, 3<, 8j, 1, 3<D

889, 13, 9<, 813, 9, 5<, 88, 11, 7<<
```

```
MatrixForm@%D

 7  8  5
15  8  6
 5  7  5
```

Όπως αναφέραμε και προηγουμένως, η συνάρτηση **Random** επιστρέφει ένα διαφορετικό τυχαίο αριθμό κάθε φορά που εκτελείται. Αυτό σημαίνει ότι αν εκτελέσουμε την ίδια εντολή που καθόρισε τον πίνακα b, τότε ο πίνακας αυτός θα αλλάξει:

```
c = Table@Random@Integer, 85, 15<D, 8i, 1, 3<, 8j, 1, 3<D

8813, 13, 10<, 88, 10, 15<, 87, 13, 8<<
```

```
MatrixForm@%D

13  13  10
 8  10  15
 7  13   8
```

Έτσι, έχουμε ουσιαστικά μια μηχανή κατασκευής τυχαίων πινάκων, οι οποίοι βέβαια έχουν κάποια κοινά χαρακτηριστικά. Τα στοιχεία τους είναι ακέραιοι αριθμοί, και μάλιστα μεταξύ 5 και 15.

Τώρα, πλέον το *Mathematica* γνωρίζει δύο 3x3 πίνακες, και συγκεκριμένα τους πίνακες b και c.

Το *Mathematica* διαθέτει αρκετές συναρτήσεις με τις οποίες διαχειρίζεται τους πίνακες. Μερικές από τις οποίες είναι:

Dot[A, B] ή A.B:	επιστρέφει το γινόμενο των πινάκων A και B.
Det[A]:	επιστρέφει την ορίζουσα ενός τετραγωνικού πίνακα A.
Inverse[A]:	επιστρέφει τον αντίστροφο πίνακα ενός αντιστρέψιμου πίνακα A.
Transpose[A]:	επιστρέφει τον ανάστροφο του πίνακα A.

Χρησιμοποιώντας τις συναρτήσεις αυτές, μπορεί κάποιος εύκολα να διαχειριστεί του πίνακες b και c. Συγκεκριμένα μπορούμε να τους πολλαπλασιάσουμε:

```
b.c

88190, 236, 230<, 8301, 353, 318<, 8156, 200, 195<<
```

```
MatrixForm@%D
```

$$\begin{pmatrix} 190 & 236 & 230 \\ 301 & 353 & 318 \\ 156 & 200 & 195 \end{pmatrix}$$

Ο πολλαπλασιασμός έγινε με την τοποθέτηση μιας τελείας μεταξύ των πινάκων. Η τελεία αυτή είναι ουσιαστικά συντομογραφία της συνάρτησης **Dot**, γεγονός που εύκολα μπορεί να επαληθευθεί:

```
Alias@"."D

Dot
```

```
Dot@b, cD

88190, 236, 230<, 8301, 353, 318<, 8156, 200, 195<<
```

```
MatrixForm@%D
```

$$\begin{pmatrix} 190 & 236 & 230 \\ 301 & 353 & 318 \\ 156 & 200 & 195 \end{pmatrix}$$

Να βρούμε την ορίζουσα τους:

```
Det@bD

−49
```

```
Det@cD

−622
```

Να βρούμε τον αντίστροφό τους, εφόσον διαπιστώσουμε ότι υπάρχει:

```
Inverse@bD
```

$$\left\{\left\{\frac{2}{49}, \frac{5}{49}, -\frac{8}{49}\right\}, \left\{\frac{45}{49}, -\frac{10}{49}, -\frac{33}{49}\right\}, \left\{-\frac{65}{49}, \frac{9}{49}, \frac{64}{49}\right\}\right\}$$

```
MatrixForm@%D
```

$$\begin{pmatrix} \frac{2}{49} & \frac{5}{49} & -\frac{8}{49} \\ \frac{45}{49} & -\frac{10}{49} & -\frac{33}{49} \\ -\frac{65}{49} & \frac{9}{49} & \frac{64}{49} \end{pmatrix}$$

```
b.%
```

```
881, 0, 0<, 80, 1, 0<, 80, 0, 1<<
```

```
MatrixForm@%D
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```
Inverse@cD
```

$$\left\{\left\{\frac{115}{622}, -\frac{13}{311}, -\frac{95}{622}\right\}, \left\{-\frac{41}{622}, -\frac{17}{311}, \frac{115}{622}\right\}, \left\{-\frac{17}{311}, \frac{39}{311}, -\frac{13}{311}\right\}\right\}$$

```
MatrixForm@%D
```

$$\begin{pmatrix} \frac{115}{622} & -\frac{13}{311} & -\frac{95}{622} \\ -\frac{41}{622} & -\frac{17}{311} & \frac{115}{622} \\ -\frac{17}{311} & \frac{39}{311} & -\frac{13}{311} \end{pmatrix}$$

```
c.%
```

```
881, 0, 0<, 80, 1, 0<, 80, 0, 1<<
```

```
MatrixForm@%D
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Τέλος μπορούμε να βρούμε τον ανάστροφό τους:

```
Transpose@bD

887, 15, 5<, 88, 8, 7<, 85, 6, 5<<
```

```
MatrixForm@%D

 7  15  5
 8   8  7
 5   6  5
```

```
Transpose@cD

8813, 8, 7<, 813, 10, 13<, 810, 15, 8<<
```

```
MatrixForm@%D

 13   8   7
 13  10  13
 10  15   8
```

Επίσης, μπορούμε στη διαχείριση των πινάκων να χρησιμοποιήσουμε και ήδη γνωστές συναρτήσεις του *Mathematica*, όπως τη συνάρτηση **Plus** ή την αντίστοιχη συντομογραφία της "+":

```
Plus@b, cD

8820, 21, 15<, 823, 18, 21<, 812, 20, 13<<
```

```
MatrixForm@%D

 20  21  15
 23  18  21
 12  20  13
```

```
b + c

8820, 21, 15<, 823, 18, 21<, 812, 20, 13<<
```

```
MatrixForm@%D
```

$$\begin{pmatrix} 20 & 21 & 15 \\ 23 & 18 & 21 \\ 12 & 20 & 13 \end{pmatrix}$$

Παρατήρηση: Το *Mathematica* δεν κάνει διάκριση μεταξύ διανυσμάτων γραμμών και διανυσμάτων στηλών. Η συνάρτηση **Dot** είναι κατασκευασμένη έτσι ώστε να δίνει κάθε δυνατό αποτέλεσμα.

Πράγματι, έστω:

```
p = 81, 2, 3<

81, 2, 3<
```

Η λίστα p που ορίστηκε μπορεί να θεωρηθεί και ως πίνακας αφού το *Mathematica* δεν κάνει διάκριση ανάμεσα στις λίστες και στους πίνακες. Επίσης, μπορεί να θεωρηθεί και ως διάνυσμα με συντεταγμένες 1, 2 και 3. Όμως, ο πίνακας p ή το διάνυσμα p μπορεί να θεωρηθεί είτε σαν πίνακας (διάνυσμα) γραμμή είτε σαν πίνακας (διάνυσμα) στήλη. Αν ο p θεωρηθεί σαν πίνακας (διάνυσμα) γραμμή, τότε το γινόμενο b·p δεν ορίζεται, ενώ αν θεωρηθεί σαν πίνακας (διάνυσμα) στήλη, τότε το γινόμενο p·b δεν μπορεί να οριστεί. Επομένως, σε κάθε περίπτωση ένα από τα γινόμενα b·p ή p·b δεν μπορεί ναοριστεί. Το *Mathematica*, όμως, θα δώσει απάντηση και στις δύο περιπτώσεις.

```
8p.b, b.p<

8859, 64, 40<, 862, 46, 51<<
```

Αυτό οφείλεται στο γεγονός ότι το *Mathematica* δεν κάνει διάκριση μεταξύ διανυσμάτων γραμμών και διανυσμάτων στηλών. Η συνάρτηση **Dot** είναι κατασκευασμένη έτσι ώστε να δίνει κάθε δυνατό αποτέλεσμα. Όταν, λοιπόν, πολλαπλασιάζουμε δύο διανύσματα {a, b, c} και {x, y, z} χρησιμοποιώντας την συνάρτηση **Dot**, το *Mathematica* αντιμετωπίζει το πρώτο σαν διάνυσμα γραμμή και το δεύτερο σαν διάνυσμα στήλη. Έτσι, ουσιαστικά επιστρέφει το εσωτερικό γινόμενο των δύο διανυσμάτων:

```
82, 4, 6<.8x, y, z<

2 x + 4 y + 6 z
```

Στο γινόμενο p·b που είδαμε παραπάνω, η συνάρτηση **Dot** αντιμετώπισε το διάνυσμα p ως γραμμή, οπότε το αποτέλεσμα ήταν ένας 1x3 πίνακας, ενώ στο γινόμενο b·p το διάνυσμα p αντιμετωπίστηκε ως στήλη, οπότε το αποτέλεσμα ήταν ένας πίνακας 3x1. Έτσι και στις δύο περιπτώσεις το *Mathematica* θα επιστέψει μια λίστα με τρία μέλη.

Η αντιμετώπιση αυτή είναι ικανοποιητική στις περισσότερες περιπτώσεις. Υπάρχουν, όμως, περιπτώσεις

που μπορούμε να οδηγηθούμε σε λάθος. Για παράδειγμα, ας υποθέσουμε ότι θέλουμε να πολλαπλασιάσουμε ένα 3x1 πίνακα X=$\{x_1,x_2,x_3\}$ με ένα 1x3 πίνακα Y=$\{y_1,y_2,y_3\}$. Ασφαλώς το αποτέλεσμα θα πρέπει να είναι ένας 3x3 πίνακας.

```
X = 8x1, x2, x3<

8x1, x2, x3<
```

```
Y = 8y1, y2, y3<

8y1, y2, y3<
```

Το *Mathematica* όμως, όπως είδαμε παραπάνω, θα αντιμετωπίσει το X σαν διάνυσμα γραμμή και το Y σαν διάνυσμα στήλη, οπότε το αποτέλεσμα δεν θα είναι ένας πίνακας 3x3 αλλά το εσωτερικό γινόμενο το διανυσμάτων X και Y:

```
X.Y

x1 y1 + x2 y2 + x3 y3
```

Μπορούμε, όμως, να αναγκάσουμε το πρόγραμμα να αντιμετωπίσει τους πίνακες X και Y όπως έμεις θέλουμε, δίνοντας τους πίνακες X και Y με μορφή πολλαπλής λίστας και όχι απλής. Συγκεκριμένα:

Δίνουμε τον πίνακα X με μορφή πολλαπλής λίστας ως εξής:

```
X = 88x1<, 8x2<, 8x3<<

88x1<, 8x2<, 8x3<<
```

Παρατηρούμε, ότι έχουμε τρεις επιμέρους λίστες οι οποίες αντιστοιχούν στις τρεις γραμμές ενός πίνακα. Ο πίνακας αυτός θα έχει ως πρώτη γραμμή την πρώτη επιμέρους λίστα, δηλαδή το στοιχείο x_1, ως δεύτερη γραμμή τη δεύτερη επιμέρους λίστα, δηλαδή το στοιχείο x_2 και ως τρίτη γραμμή την τρίτη επιμέρους λίστα, δηλαδή το στοιχείο x_3. Δηλαδή θα έχουμε έναν πίνακα 3x1, ακριβώς όπως θέλαμε.

Πράγματι, ο πίνακας X στην συνήθη του μορφή είναι ένας 3x1 πίνακας:

```
MatrixForm@%D

i x1 y
j x2 z
k x3 {
```

Δίνουμε τον πίνακα X με μορφή πολλαπλής λίστας ως εξής:

```
Y = 88y1, y2, y3<<

88y1, y2, y3<<
```

Παρατηρούμε, ότι έχουμε μία επιμέρους λίστα η οποία αντιστοιχεί σε μια γραμμή ενός πίνακα. Ο πίνακας αυτός θα έχει ως πρώτη και μοναδική γραμμή την επιμέρους λίστα, δηλαδή τα στοιχεία $8x_1, x_2, x_3<$. Επομένως θα έχουμε έναν πίνακα 1x3, ακριβώς όπως θέλαμε.

Πράγματι, ο πίνακας Y στην συνήθη του μορφή είναι ένας 1x3 πίνακας:

```
MatrixForm@%D

H y1  y2  y3 L
```

Στην περίπτωση που δώσουμε τους πίνακες X και Y με την παραπάνω μορφή τους, το Mathemtica επιστρέφει ως αποτέλεσμα του γινομένου των πινάκων X και Y έναν 3x3 πίνακα:

```
X.Y

88x1 y1, x1 y2, x1 y3<, 8x2 y1, x2 y2, x2 y3<, 8x3 y1, x3 y2, x3 y3<<
```

```
MatrixForm@%D

i x1 y1  x1 y2  x1 y3 y
j x2 y1  x2 y2  x2 y3 z
k x3 y1  x3 y2  x3 y3 {
```

2.3 Συναρτήσεις για τη Διαχείριση Λιστών

Επειδή οι λίστες παίζουν σημαντικό ρόλο στο *Mathematica*, το πρόγραμμα διαθέτει αρκετές συναρτήσεις με τις οποίες κανείς μπορεί να διαχειριστεί τις λίστες.

Έστω η λίστα με όνομα a:

```
a = 81, 2, 3, 83, 4<, 85, 6, 7<, 8, 9, 10<

81, 2, 3, 83, 4<, 85, 6, 7<, 8, 9, 10<
```

Στη συνέχεια δίνονται κάποιες συναρτήσεις με τις οποίες μπορούμε να πάρουμε έναν ή περισσότερους όρους ή ακόμη και συγκεκριμένα μέρης της λίστας a:

First[expr]: επιστρέφει το πρώτο στοιχείο της παράστασης expr.

```
First@aD

1
```

Last[expr]: επιστρέφει το τελευταίο στοιχείο της παράστασης expr.

```
Last@aD

10
```

Extract[expr, list]: επιστρέφει τα στοιχεία της expr, τα οποία βρίσκονται στις θέσεις που καθορίζονται στη list.

```
Extract@a, 4D

83, 4<
```

```
Extract@a, 84, 1<D

3
```

```
Extract@a, 884, 2<, 85, 2<<D

84, 6<
```

Take[list, n]: επιστρέφει μία λίστα με τα πρώτα n στοιχεία της λίστας list.

```
Take@a, 5D

81, 2, 3, 83, 4<, 85, 6, 7<<
```

Take[list, -n]: επιστρέφει μία λίστα με τα τελευταία n στοιχεία της λίστας list.

```
Take@a, −5D

883, 4<, 85, 6, 7<, 8, 9, 10<
```

Take[list, {m, n}]: επιστρέφει μία λίστα με τα στοιχεία της λίστας list, που βρίσκονται από τη θέση m έως τη θέση n.

```
Take@a, 84, 5<D

883, 4<, 85, 6, 7<<
```

Drop[list, n]: διαγράφει τα πρώτα n στοιχεία της λίστας list.

```
b = Drop@a, 3D

883, 4<, 85, 6, 7<, 8, 9, 10<
```

Drop[list, -n]: διαγράφει τα τελευταία n στοιχεία της λίστας list.

```
Drop@b, −2D

883, 4<, 85, 6, 7<, 8<
```

Drop[list, {m, n}]: διαγράφει τα στοιχεία της λίστας list, που βρίσκονται από τη θέση m έως τη θέση n.

```
Drop@a, 84, 5<D

81, 2, 3, 8, 9, 10<
```

Drop[list, {n}]: διαγράφει το στοιχείο της λίστας list που βρίσκεται στη θέση n.

```
Drop@a, 88<D

81, 2, 3, 83, 4<, 85, 6, 7<, 8, 9<
```

Rest[expr]: επιστρέφει τη παράσταση expr, διαγράφοντας το πρώτο στοιχείο.

```
Rest@aD

82, 3, 83, 4<, 85, 6, 7<, 8, 9, 10<
```

```
Rest@bD

885, 6, 7<, 8, 9, 10<
```

2.4 Λίστες και Κατασκευή Συναρτήσεων.

Όπως είπαμε και στην αρχή του κεφαλαίου οι λίστες παίζουν σημαντικό ρόλο στη χρήση συναρτήσεων του *Mathematica*, αφού συχνά, οι απαντήσεις που δίνει το πρόγραμμα κατά την εκτέλεση συναρτήσεων έχουν τη μορφή λίστας. Επομένως οι λίστες παίζουν σημαντικό ρόλο και στη κατασκευή συνάρτησης με συνδυασμό υπαρχώντων συναρτήσεων του *Mathematica*.

Για παράδειγμα, η παρακάτω συνάρτηση primeDivisors, η οποία βρίσκει τους πρώτους διαιρέτες ενός ακεραίου, κατασκευάστηκε με συνδυασμό των συναρτήσεων **FactorInteger**, **Transpose** και **First** του *Mathematica*, δηλαδή συναρτήσεων που δίνουν αποτέλεσμα με μορφή λίστας ή διαχειρίζονται λίστες.

```
primeDivisors@x_IntegerD := First@Transpose@FactorInteger@xDDD
```

Παρατηρούμε ότι το όνομα της συνάρτησης είναι primeDivisors και ότι ακολουθείται η περιγραφική λογική του *Mathematica* με τη διαφορά ότι το πρώτο γράμμα της πρώτης λέξης δεν είναι κεφαλαίο. Με τον τρόπο αυτό διακρίνονται εύκολα οι συναρτήσεις που κατασκευάζει ο χρήστης, από αυτές που είναι ενσωματωμένες στο *Mathematica*.

Είδαμε στο προηγούμενο κεφάλαιο, ότι ο συμβολισμός x_ σημαίνει ότι το x πρέπει να αντιμετωπιστεί σαν μεταβλητή, η οποία θα αντικατασταθεί, κατά την εκτέλεση της συνάρτησης, από το όρισμα που θα δώσει ο χρήστης. Ο συμβολισμός x_Integer σημαίνει το ίδιο με την επιπλέον συνθήκη ότι το όρισμα, που θα δώσει ο χρήστης, πρέπει να έχει επικεφαλίδα Integer. Δηλαδή, γίνεται έλεγχος στο όρισμα που δίνει ο χρήστης, και η συνάρτηση εκτελείται μόνο αν το όρισμα είναι ακέραιος.

Για παράδειγμα, αν το όρισμα της συνάρτησης είναι ο ακέραιος 808500, η συνάρτηση εκτελείται και μας δίνει ως αποτέλεσμα (με μορφή λίστας) τους πρώτους διαιρέτες του ακεραίου 808500,

```
primeDivisors@808500D

82, 3, 5, 7, 11<
```

ενώ αν το όρισμα της συνάρτησης είναι ο πραγματικός αριθμός 808500.0, η συνάρτηση δεν εκτελείται

```
primeDivisors@808500.0D

primeDivisors@808500.D
```

αφού ο αριθμός 808500.0 έχει επικεφαλίδα Real.

```
Head@808500.0D

Real
```

Ας δούμε, τώρα, πως δουλεύει η συνάρτηση primeDivisors που κατασκευάσαμε:

Είναι γνωστό, από το πρώτο κεφάλαιο, ότι η συνάρτηση **FactorInteger** επιστρέφει μια λίστα λιστών της μορφής:

$$\{\{p_1, n_1\},\{p_2, n_2\},...,\{p_m, n_m\}\}$$

όπου p_i, i=1, ..., m, είναι οι πρώτοι παράγοντες του ακεραίου (που δίνεται ως όρισμα) και n_i οι αντίστοιχοι εκθέτες:

```
FactorInteger@808500D

882, 2<, 83, 1<, 85, 3<, 87, 2<, 811, 1<<
```

Επειδή η κάθε επιμέρους λίστα περιέχει δύο στοιχεία, η απάντηση που δίνει η συνάρτηση **FactorInteger** είναι ένας πίνακας με δύο στήλες. Η πρώτη στήλη περιέχει τους πρώτους παράγοντες και η δεύτερη στήλη τους αντίστοιχους εκθέτες. Επομένως αν πάρουμε τον ανάστροφο πίνακά του, με τη χρήση της συνάρτησης **Transpose**, θα βρούμε έναν πίνακα με δύο γραμμές. Η πρώτη γραμμή θα αποτελείται από τους πρώτους παράγοντες του ακεραίου και η δεύτερη γραμμή από τους αντίστοιχους εκθέτες:

```
Transpose@%D

882, 3, 5, 7, 11<, 82, 1, 3, 2, 1<<
```

Παρατηρούμε ότι ο πίνακας δίνεται με τη μορφή λίστας που περιέχει δύο επιμέρους λίστες. Η πρώτη επιμέρους λίστα αντιστοιχεί στη πρώτη γραμμή του πίνακα και η δεύτερη επιμέρους λίστα στη δεύτερη γραμμή του πίνακα. Επομένως αν στη λίστα που προέκυψε εφαρμόσουμε την συνάρτηση **First** θα έχουμε αποτέλεσμα μία λίστα με τους πρώτους διαιρέτες του ακεραίου που έχουμε δώσει ως όρισμα:

```
First@%D

82, 3, 5, 7, 11<
```

Η συνάρτηση primeDivisors κατασκευάστηκε για να δίνει τους πρώτους διαιρέτες ενός ακεραίου. Δεν είναι όμως ο μοναδικός τρόπος με τον οποίο μπορούμε να πάρουμε το αποτέλεσμα αυτό. Ας δούμε και μία δεύτερη εκδοχή της ίδιας συνάρτησης.

Το *Mathematica* διαθέτει τη συνάρτηση **Divisors** η οποία μας επιστρέφει όλους τους θετικού διαιρέτες ενός ακεραίου:

Divisors[n]: επιστρέφει μια λίστα με όλους τους θετικούς ακεραιούς οι οποίοι διαιρούν τον ακέραιο n.

```
Divisors@808500D

81, 2, 3, 4, 5, 6, 7, 10, 11, 12, 14, 15, 20, 21, 22, 25, 28, 30,
 33, 35, 42, 44, 49, 50, 55, 60, 66, 70, 75, 77, 84, 98, 100,
 105, 110, 125, 132, 140, 147, 150, 154, 165, 175, 196, 210,
 220, 231, 245, 250, 275, 294, 300, 308, 330, 350, 375, 385,
 420, 462, 490, 500, 525, 539, 550, 588, 660, 700, 735, 750,
 770, 825, 875, 924, 980, 1050, 1078, 1100, 1155, 1225, 1375,
 1470, 1500, 1540, 1617, 1650, 1750, 1925, 2100, 2156, 2310,
 2450, 2625, 2695, 2750, 2940, 3234, 3300, 3500, 3675, 3850,
 4125, 4620, 4900, 5250, 5390, 5500, 5775, 6125, 6468, 7350,
 7700, 8085, 8250, 9625, 10500, 10780, 11550, 12250, 13475,
 14700, 16170, 16500, 18375, 19250, 23100, 24500, 26950, 28875,
 32340, 36750, 38500, 40425, 53900, 57750, 67375, 73500, 80850,
 115500, 134750, 161700, 202125, 269500, 404250, 808500<
```

Ασφαλώς ανάμεσα σε αυτούς υπάρχουν και οι πρώτοι διαιρέτες του ακεραίου. Επομένως, αν μπορούσαμε να καθοδηγήσουμε το πρόγραμμα να επιλέξει από την προηγούμενη λίστα των διαιρετών του ακεραίου μόνο τους πρώτους αριθμούς, θα είχαμε το ίδιο αποτέλεσμα με τη συνάρτηση primeDivisors που έχουμε κατασκευάσει.

Το *Mathematica* διαθέτει τις συναρτήσεις **Select** και **PrimeQ**:

Select[list, crit]: επιλέγει εκείνα τα στοιχεία της λίστας list, για τα οποία η συνθήκη crit είναι αληθής.
PrimeQ[expr]: επιστρέφει True αν η expr είναι πρώτος αριθμός και False σε κάθε άλλη περίπτωση.

Ο συνδυασμός των συναρτήσεων **Select** και **PrimeQ** μπορεί να βοήθησει προς την κατεύθυνση αυτή. Συγκεκριμένα, η συνάρτηση PrimeQ μπορεί να αποτελέσει τη συνθήκη με το οποίο η συνάρτηση Select θα επιλέξει τα στοιχεία από μια λίστα:

```
Select@%, PrimeQD

82, 3, 5, 7, 11<
```

Επομένως, η συνάρτηση primeDivisors μπορεί να κατασκευαστεί και με τον παρακάτω τρόπο:

```
primeDiv@x_IntegerD := Select@Divisors@xD, PrimeQD
```

```
primeDiv@808500D

82, 3, 5, 7, 11<
```

Όπως βλέπουμε οι συναρτήσεις primeDiv και primeDivisors έχουν ακριβώς το ίδιο αποτέλεσμα. Ωστόσο η μία από τις δύο είναι γρηγορότερη από τη άλλη. Παρακολουθώντας το παρακάτω παράδειγμα, με τη συνάρτηση **Timing** να μετρά το χρόνο εκτέλεσης της κάθε συνάρτησης

```
8Timing@primeDiv@192139662355662352351765136123DD,
 Timing@primeDivisors@192139662355662352351765136123DD<

880.406 Second, 837, 383, 401, 49297, 8136893, 84293300053<<,
 80.391 Second, 837, 383, 401, 49297, 8136893, 84293300053<<<
```

παρατηρούμε ότι η συνάρτηση primeDiv χρειάζεται περισσότερο χρόνο. Αυτό είναι φυσιολογικό, διότι η συνάρτηση primeDiv χρησιμοποιεί τη συνάρτηση **Divisors** η οποία μας δίνει όλους τους διαιρέτες του ακεραίου, οι οποίοι στη συνέχεια ελέγχονται ένας προς ένας από τη συνάρτηση **PrimeQ**, για να μας επιστρέψει η συνάρτηση **Select** μόνο αυτούς που είναι πρώτοι αριθμοί. Είναι προφανές ότι δεν χρειαζόμαστε όλους τους διαιρέτες, αφού πολλοί από αυτούς δεν θα είναι πρώτοι. Αντίθετα, η συνάρτηση primeDivisors χρησιμοποιεί την συνάρτηση **FactorInteger**, η οποία μας δίνει μόνον τους πρώτους διαιρέτες του ακεραίου, οπότε δεν χρειάζεται να γίνει οποιαδήποτε επιλογή.

Το παράδειγμα αυτό δείχνει ότι έχει σημασία ο τρόπος με τον οποίο κατασκευάζεται μια συνάρτηση, διότι βλέπουμε ότι η κατασκευή προσδιορίζει και το χρόνο εκτέλεσης της συνάρτησης.

Ας δούμε ένα ακόμα παράδειγμα κατασκευής συνάρτησης.

Το *Mathematica* διαθέτει τη συνάρτηση **GCD** η οποία βρίσκει τον μέγιστο κοινό διαιρέτη ακεραίων:

GCD$[n_1, n_2, ..., n_k]$: επιστρέφει το μέγιστο κοινό διαιρέτη των ακεραίων n_i.

```
GCD@124, 568, 10^3D

4
```

Ασφαλώς δεν είναι ανάγκη να βρούμε κάποια άλλη συνάρτηση υπολογισμού του μέγιστου κοινού διαιρέτη ακεραίων. Μπορούμε, όμως, χρησιμοποιώντας γνωστές συναρτήσεις του *Mathematica* να επαληθεύσουμε τη συνάρτηση **GCD**.

Όπως είναι γνωστό, ο Μ.Κ.Δ. των ακεραίων n_1, n_2, ..., n_k είναι ο μεγαλύτερος από τους κοινούς διαιρέτες των αριθμών αυτών.

Η συνάρτηση **Divisors** μπορεί να μας δώσει τους διαιρέτες των ακεραίων που μας ενδιαφέρουν. Π.χ.

```
lst1 = Divisors@124D

81, 2, 4, 31, 62, 124<
```

```
lst2 = Divisors@568D

81, 2, 4, 8, 71, 142, 284, 568<
```

```
lst3 = Divisors@10^3D

81, 2, 4, 5, 8, 10, 20, 25, 40, 50, 100, 125, 200, 250, 500, 1000<
```

Το *Mathematica* διαθέτει τις συναρτήσεις **Intersection** και **Union** με τις οποίες βρίσκει την τομή και την ένωση λιστών:

Intersection[$list_1$, $list_2$, ..., $list_k$]: επιστρέφει μια ταξινομημένη λίστα των κοινών στοιχείων των λιστών $list_i$.

Union[$list_1$, $list_2$, ..., $list_k$]: επιστρέφει μια ταξινομημένη λίστα όλων των στοιχείων που περιέχονται σε μία τουλάχιστον από τις λίστες $list_i$.

Εφαρμόζοντας τη συνάρτηση **Intersection** στις λίστες lst1 (οι διαιρέτες του 124), lst2 (οι διαιρέτες του 568) και lst3 (οι διαιρέτες του 10^3) έχουμε ως αποτέλεσμα τη λίστα με τους κοινούς διαιρέτες των αριθμών 124, 568 και 10^3.

```
Intersection@lst1, lst2, lst3D

81, 2, 4<
```

Παίρνοντας το τελευταίο στοιχείο της προηγούμενης λίστας με τη συνάρτηση **Last**, βρίσκουμε τον Μ.Κ.Δ. των αριθμών 124, 568 και 10^3.

```
Last@%D

4
```

Αν συνδυάσουμε όλες αυτές τις συναρτήσεις, μπορούμε να έχουμε την επαλήθευση που ζητούμε:

```
GCD@124, 568, 10^3D « Last@
  Intersection@Divisors@124D, Divisors@568D, Divisors@10^3DDD

True
```

2.4 Πολυώνυμα

Είναι γνωστό ότι ένα πολυώνυμο p(x) μιας μεταβλητής x, με πραγματικούς συντελεστές, είναι μια παράσταση της μορφής:

$$p(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + ... + \alpha_n x^n$$

όπου οι συντελεστές α_0, α_1, α_2, ..., α_n είναι πραγματικοί αριθμοί. Είναι προφανές ότι θα μπορούσαμε να έχουμε πολυώνυμα με ακεραίους ή ρητούς συντελεστές.

Ο βαθμός του πολυωνύμου δεν μπορεί να προσδιορίσει το πολυώνυμο, διότι υπάρχουν πολλά πολυώνυμα με το ίδιο βαθμό. Εκείνο που προσδιορίζει πλήρως ένα πολυώνυμο είναι η ακολουθία των συντελεστών του, δηλαδή η ακολουθία α_0, α_1, α_2, ..., α_n. Πράγματι, αν υποθέσουμε ότι έχουμε την ακολουθία 2, -2, 1, 2, -3, 4, τότε το πολυώνυμο που αντιστοιχεί σε αυτή είναι:

$$2 - 2 x + x^2 + 2 x^3 - 3 x^4 + 4 x^5$$

με την προϋπόθεση ότι θα γράφουμε το πολυώνυμο αρχίζοντας πάντοτε με το σταθερό του όρο. Ενώ για το πολυώνυμο:

$$2 x^2 + 2 x^4 - 3 x^5$$

η ακολουθία των συντελεστών που αντιστοιχεί σε αυτό είναι: 0, 0, 2, 0, 2, -3.

Έστω το πολυώνυμο p1:

```
p1 = 6 + 5 x - 9 x^2 - 3 x^3 + 10 x^4 - 4 x^6

6 + 5 x - 9 x^2 - 3 x^3 + 10 x^4 - 4 x^6
```

Το *Mathematica* διαθέτει αρκετές συναρτήσεις με τις οποίες μπορούμε να διαχειριστούμε πολυώνυμα. Μερικές από αυτές είναι:

Coefficient[expr, form]: επιστρέφει τον συντελεστή του όρου form στην παράσταση expr.

Coefficient[expr, form,n]: επιστρέφει τον συντελεστή του όρου form^n στην παράσταση expr.

```
Coefficient@p1, xD

5
```

```
Coefficient@p1, x^4D

10
```

```
Coefficient@p1, x, 4D

10
```

```
Coefficient@p1, x^2, 2D

10
```

Variables[poly]: επιστρέφει μια λίστα με όλες τις μεταβλητές του πολυωνύμου poly.

```
Variables@p1D

8x<
```

```
p2 = x + 2 y^2 + x^2 y^3

x + 2 y^2 + x^2 y^3
```

```
Variables@p2D

8x, y<
```

CoefficientList[poly, var]: επιστρέφει μια λίστα με τους συντελεστές των δυνάμεων της μεταβλητής var στο πουλώνυμο poly, αρχίζοντας από το σταθερό όρο.

```
CoefficientList@p1, xD

86, 5, −9, −3, 10, 0, −4<
```

```
CoefficientList@p2, xD

82 y^2, 1, y^3<
```

```
CoefficientList@p2, yD

8x, 0, 2, x^2<
```

Όταν θέλουμε να εισάγουμε ένα συγκεκριμένο πολυώνυμο στο *Mathematica* πρέπει να πληκτρολογήσουμε ένα προς ένα όλους τους όρους του. Υπάρχουν, όμως, περιπτώσεις που θέλουμε ένα τυχαίο πολυώνυμο, π,χ, με ακέραιους συντελεστές και κάποιου βαθμού. Στις περιπτώσεις αυτές θα μπορούσαμε να αποφύγουμε τη πληκτρολόγηση όλων των όρων, αν είχαμε μία "μηχανή" παραγωγής πολυωνύμων. Μια τέτοια μηχανή μπορεί να κατασκευαστεί συνδυάζοντας γνωστές συναρτήσεις του *Mathematica*.

Έστω ότι θέλουμε ένα τυχαίο πολυώνυμο με ακέραιους συντελεστές μεταξύ -5 και 5 και 6ου βαθμού. Μια πρώτη εκτίμηση για ένα τέτοιο πολυώνυμο θα είναι:

```
Table@Random@Integer, 8−5, 5<D x^k, 8k, 0, 6<D

83, −x, −5 x^2, −5 x^3, −4 x^4, 4 x^5, −2 x^6<
```

Παρατηρούμε ότι το αποτέλεσμα που μας επέστρεψε το *Mathematica* δεν είναι ένα πολυώνυμο, αλλά μια λίστα, της οποίας τα μέλη είναι οι όροι κάποιου πολυωνύμου. Ξέρουμε, όμως, ότι η διαφορά ανάμεσα στο άθροισμα και στη λίστα είναι η διαφορετική επικεφαλίδα, επομένως αν αλλάξουμε την επικεφαλίδα αυτή μπορούμε να πάρουμε το άθροισμα που θέλουμε:

```
Plus @@ %
```

$$3 - x - 5\,x^2 - 5\,x^3 - 4\,x^4 + 4\,x^5 - 2\,x^6$$

Ο συμβολισμός @@ είναι συντομογραφία της συνάρτησης **Apply**, η οποία όπως έχουμε δει αλλάζει την επικεφαλίδα παραστάσεων:

```
Alias@"@@"D
```

```
Apply
```

Επομένως, βρήκαμε μια μηχανή παραγωγής πολυωνύμων. Μάλιστα θα μπορούσαμε να τη γράψουμε με μορφή συνάρτησης για να μπορούμε να τη χρησιμοποιούμε:

```
randomPoly@n_D :=
 Plus @@ Table@Random@Integer, 8-5, 5<D x^k, 8k, 0, n<D
```

Είναι προφανές, ότι η συνάρτηση αυτή δίνει ένα πολυώνυμο n-ου βαθμού με ακέραιους συντελεστές μεταξύ -5 και 5:

```
p2 = randomPoly@4D
```

$$5\,x^2 - 3\,x^3 - 5\,x^4$$

Επειδή, ο βαθμός ενός πολυωνύμου είναι πάντοτε φυσικός αριθμός, η συνάρτηση αυτή θα μπορούσε να βελτιωθεί, αν επιβάλλουμε το n να είναι ένας φυσικός αριθμός:

```
randomPolynomial@n_Integer?PositiveD :=
 Plus @@ Table@Random@Integer, 8-5, 5<D x^k, 8k, 0, n<D
```

Η επικεφαλίδα Integer αναγκάζει το n να είναι ακέραιος, ενώ ο έλεγχος ?Positive αναγκάζει το n να είναι και θετικός, αν θέλουμε η συνάρτηση randomPolynomial να δώσει απάντησει:

```
p3 = randomPolynomial@5D
```

$$2 - 4\,x - 5\,x^2 + 2\,x^3 - x^4 - 4\,x^5$$

```
randomPolynomial@-5D
```

```
randomPolynomial@-5D
```

```
randomPolynomial@0D

randomPolynomial@0D
```

Τώρα, το *Mathematica* γνωρίζει τρία διαφορετικά πολυώνυμα p1, p2 και p3, των οποίων μπορούμε να υπολογίσουμε το γινόμενό τους:

```
p1  p2  p3

H5 x^2 − 3 x^3 − 5 x^4L H2 − 4 x − 5 x^2 + 2 x^3 − x^4 − 4 x^5L
 H6 + 5 x − 9 x^2 − 3 x^3 + 10 x^4 − 4 x^6L
```

Παρατηρούμε ότι το αποτέλεσμα που επιστρέφει το *Mathematica* δεν είναι ικανοποιητικό, αφού παρουσιάζει το γινόμενο τοποθετώντας το κάθε πολυώνυμο μέσα σε παρενθέσεις.

Το *Mathematica,* όμως, διαθέτει τη συνάρτηση **Expand** με την οποία μπορούμε να αναπτύσουμε ένα γινόμενο.

Expand[expr] αναπτύσσει όλα τα γινόμενα και τις ακέραιες δυνάμεις που υπάρχουν στην παράσταση expr.

```
Expand@%D

60 x^2 − 106 x^3 − 358 x^4 + 359 x^5 + 694 x^6 − 688 x^7 − 564 x^8 +
 960 x^9 + 260 x^10 − 681 x^11 + 54 x^12 + 308 x^13 − 68 x^14 − 80 x^15
```

Επίσης, το *Mathematica* διαθέτει συναρτήσεις με τις οποίες αναλύει πολυώνυμα σε γινόμενο πολυωνύμων:

Factor[poly] μετατρέπει το πολυώνυμο poly σε γινόμενο πολυωνύμων με ακέραιους συντελεστές εφόσον αυτό είναι δυνατό.

Simplify[expr] εκτελεί μια σειρά μετασχηματισμών στην παράσταση expr και επιστρέφει την απλούστερη μορφή που βρίσκει.

Έστω το πολυώνυμο poly:

```
poly = H2 + xL^4 H1 + xL^4 H3 + xL^3;
```

Αναπτύσουμε το γινόμενο και πέρνουμε το πολυώνυμο poly1:

```
poly1 = Expand@polyD

432 + 3024 x + 9432 x^2 + 17296 x^3 + 20715 x^4 +
 17015 x^5 + 9783 x^6 + 3939 x^7 + 1089 x^8 + 197 x^9 + 21 x^10 + x^11
```

Μετρατρέπουμε το πολυώνυμο poly1 σε απλούστερη μορφή:

```
Simplify@poly1D

H3 + xL^3 H2 + 3 x + x^2L^4
```

Αναλύουμε το πολυώνυμο poly1 σε γινόμενο πολυωνύμων:

```
Factor@poly1D

H1 + xL^4 H2 + xL^4 H3 + xL^3
```

Δύο συνήθη προβλήματα που σχετίζονται με τα πολυώνυμα είναι η εύρεση της τιμής ενός πολυωνύμου, όταν η μεταβλητή του αντικαθίσταται με ένα συγκεκριμένο αριθμό και η εύρεση ριζών, εφόσον υπάρχουν. Το *Mathematica* διαθέτει συναρτήσεις, οι οποίες δίνουν απάντηση και στα δύο αυτά προβλήματα, άλλοτε με ακρίβεια και άλλοτε με προσέγγιση.

ReplaceAll[expr, rules] ή expr /. rules	χρησιμοποιεί του κανόνες rules που δίνονται, για να μετασχηματίσει την παράσταση expr.
Solve[eqn, var] var.	επιχειρεί να βρει τις ακριβείς ρίζες της εξίσωσης eqn ως προς τη μεταβλητή

Οι κανόνες (rules) δίνονται με τη μορφή x **Ø** a, όπου το σύμβολο **Ø** είναι ο συνδυασμός των χαρακτήρων - και >.

```
ReplaceAll@poly, x → 1D

82944
```

```
Alias@"ê."D

ReplaceAll
```

Τα πολυώνυμα poly και poly1, τα οποία ορίσαμε προηγουμένως, παριστάνουν ουσιαστικά το ίδιο πολυώνυμο. Επομένως, το Mathematica θα πρέπει να απαντάει καταφατικά στο επόμενο ερώτημα:

```
ReplaceAll@poly, x → 1D « poly1 ê. x → 1

True
```

Η συνάρτηση **Solve** είναι αποτελεσματική, όταν η πολυωνιμική εξίσωση είναι βαθμού το πολύ 4. Ας δούμε μερικά παραδείγματα:

```
Solve@-2 x^2 + 28 x - 39 « 0, xD

           1           è!!!!!!!              1           è!!!!!!!
99x →  ¡¡¡ I14 -  118 M=, 9x →  ¡¡¡ I14 +  118 M==
           2                                 2
```

Παρατηρούμε ότι το αποτέλεσμα της συνάρτησης **Solve** δεν είναι της μορφής x = x_1, αλλά δίνεται σαν κανόνας αντικατάστασης x **Ø** x_1. Αυτό μας επιτρέπει, όπως θα δούμε παρακάτω, να κάνουμε την επαλήθευση.

```
Solve@x^3 - 12 x^2 + 31 x - 27 « 0, xD

                                 è!!!!!!!!!!!      1ê3              è!!!!!!!!!!!    1ê3
                1   i 837      3   18885   y         I 1 I279 +  18885 MM
99x → 4 +  ¡¡¡ j ¡¡¡¡¡¡ - ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ z    + ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ =,
                3   k  2            2        {              32ê3
                                 è!!!   i 837      3 è!!!!!!!!!!!     1ê3
  9x → 4 -  1  I1 + ą   3 M j ¡¡¡¡¡¡ - ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ z     -
            6                   k  2              2      {
                  è!!!        1            è!!!!!!!!!!!     1ê3
      I1 - ą  3 M I ¡ I279 +  18885 MM
      ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ =,
                        2 32ê3
                                 è!!!   i 837      3 è!!!!!!!!!!!     1ê3
  9x → 4 -  1  I1 - ą   3 M j ¡¡¡¡¡¡ - ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ z     -
            6                   k  2              2      {
                  è!!!        1            è!!!!!!!!!!!     1ê3
      I1 + ą  3 M I ¡ I279 +  18885 MM
      ¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡¡ ==
                        2 32ê3
```

Επίσης, παρατηρούμε ότι μερικές φορές η απάντηση που δίνει η συνάρτηση **Solve** δεν είναι ικανοποιητική. Στις περιπτώσεις αυτές μπορούμε να βρούμε την αριθμητική τιμή των ριζών με δύο τρόπους:

Ο ένας είναι να χρησιμοποιήσουμε τη συνάρτηση **N**:

```
N@%D

88x → 8.83812<, 8x → 1.58094 + 0.745374 ą<,
 8x → 1.58094 - 0.745374 ą<<
```

Ο άλλος είναι να δώσουμε έναν τουλάχιστον από τους συντελεστές του πολυωνύμου με μορφή δεκαδικού αριθμού:

```
Solve@x^3 - 12 x^2 + 31 x - 27.0 « 0, xD

88x → 1.58094 - 0.745374 ą<,
 8x → 1.58094 + 0.745374 ą<, 8x → 8.83812<<
```

Ας δούμε στη συνέχεια πως μπορούμε να κάνουμε την επαλήθευση των ριζών μιας εξίσωσης.

Έστω το πολυώνυμο 3ου βαθμού:

```
a = 6 - 5 x - 2 x^2 + x^3

6 - 5 x - 2 x^2 + x^3
```

Υπολογισμός των ριζών με χρήση της συνάρτησης **Solve**:

```
t = Solve@a « 0, xD

88x → -2<, 8x → 1<, 8x → 3<<
```

Επαλήθευση των ριζών της εξίσωσης με χρήση του τελεστή αντικατάσταση /.:

```
a ê. t

80, 0, 0<
```

Επαλήθευση των ριζών της εξίσωσης με χρήση της συνάρτησης **ReplaceAll**:

```
ReplaceAll@a, tD

80, 0, 0<
```

Όταν έχουμε μια εξίσωση 3ου ή 4ου βαθμού, οι ρίζες δίνονται συνήθως με πολύπλοκες παραστάσεις, οι οποίες τις περισσότερες φορές περιέχουν διπλά ριζικά. Ρίζες με τέτοια μορφή δεν είναι συνήθως χρήσιμες, οπότε σε αυτές τις περιπτώσεις βρίσκουμε τις αριθμητικές τιμές των ριζών είτε με τη χρήση της συνάρτησης **N** είτε με το να δίνουμε έναν συντελεστή της εξίσωσης με μορφή δεκαδικού αριθμού. Όμως, και στις δύο περιπτώσεις, οι ρίζες που θα πάρουμε είναι προσεγγίσεις και όχι ακριβείς ρίζες.

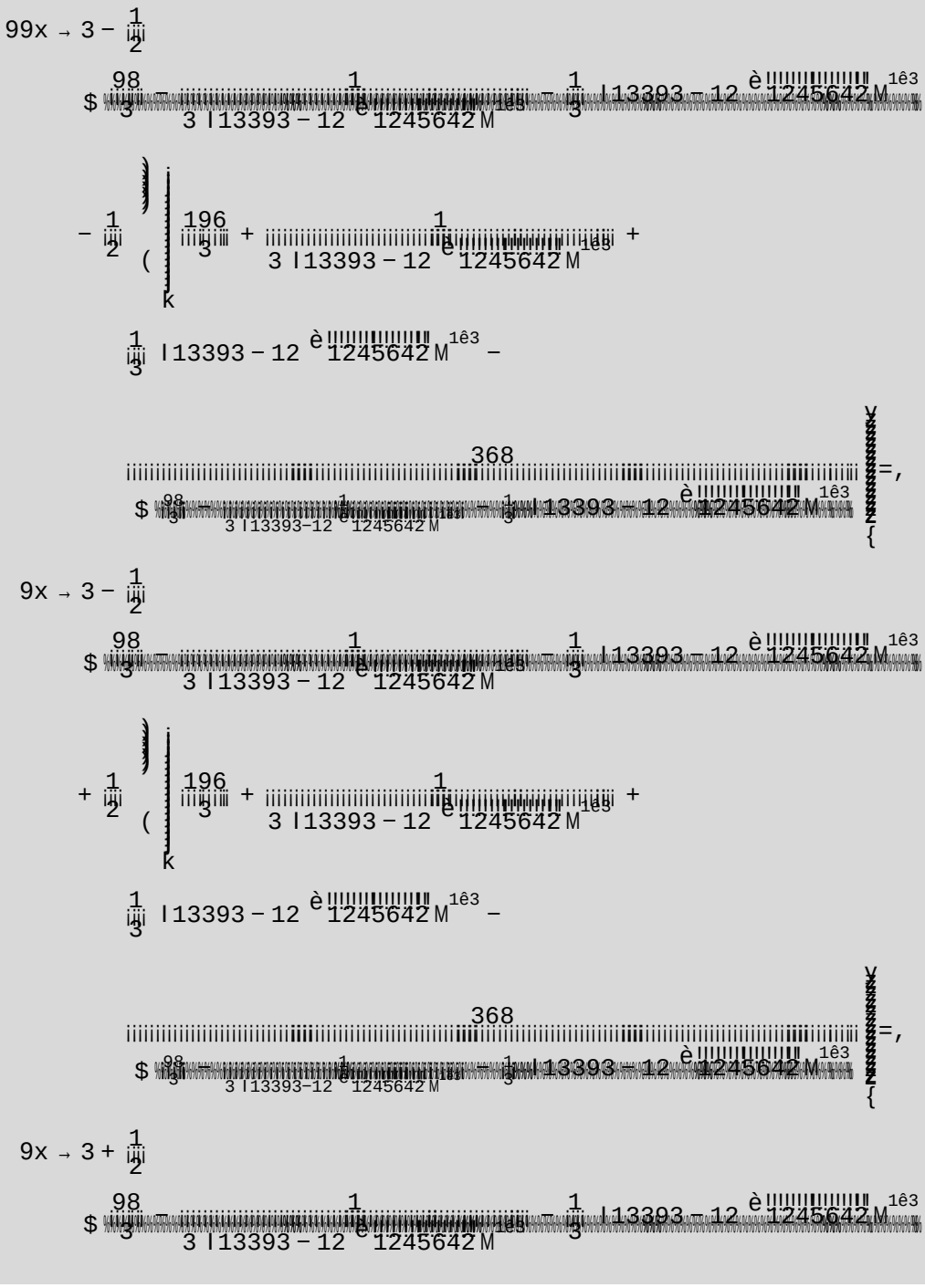
Έστω το πολυώνυμο 4ου βαθμού:

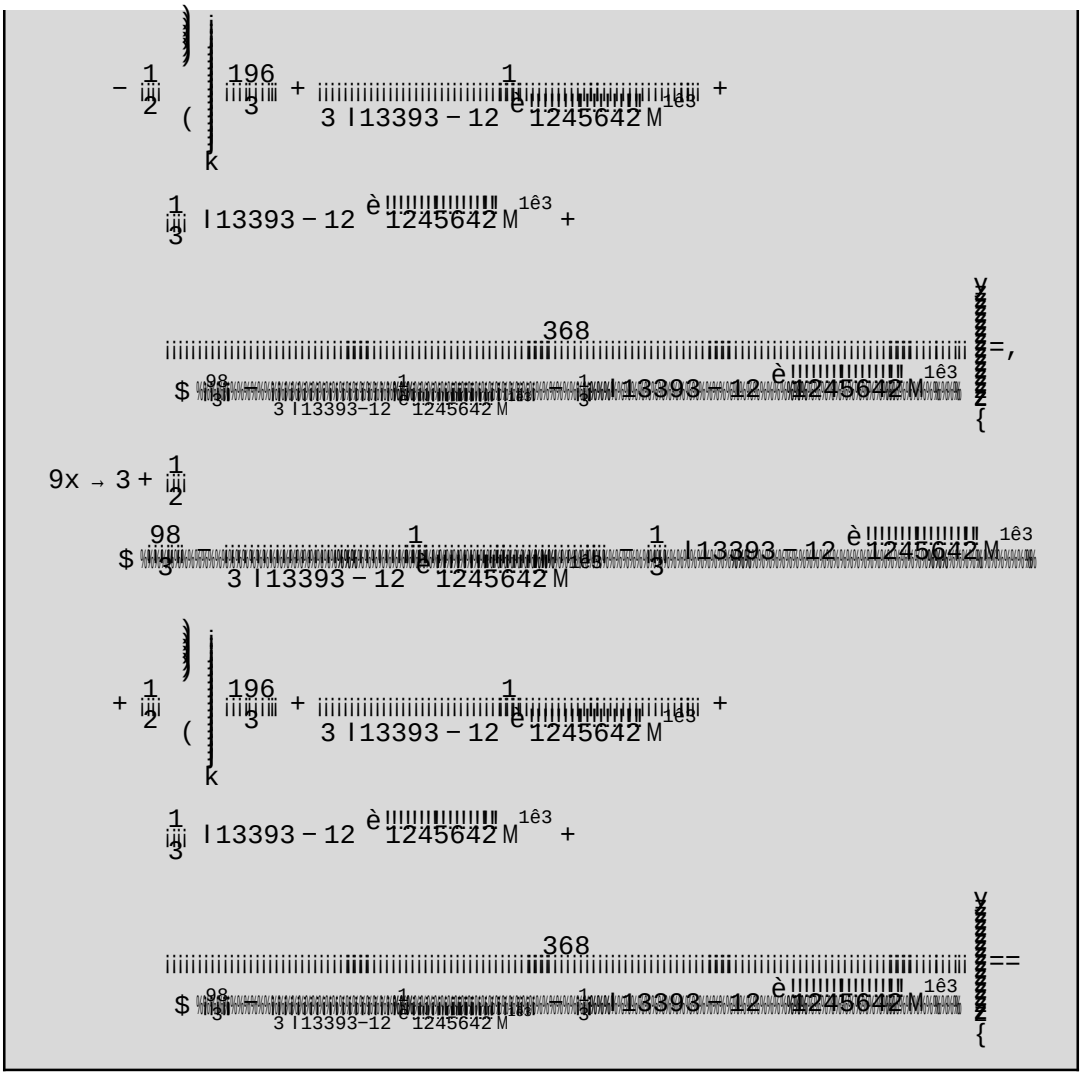
```
b = -8 + 2 x + 5 x^2 - 12 x^3 + x^4
```

$-8 + 2\,x + 5\,x^2 - 12\,x^3 + x^4$

Υπολογισμός των ριζών με χρήση της συνάρτησης **Solve**:

```
Solve@b « 0, xD
```





Υπολογισμός των αριθμητικών τιμών των ριζών με τη χρήση της συνάρτησης **N**:

```
N@%D

88x → 0.618718 - 0.698448 ą<,
 8x → 0.618718 + 0.698448 ą<, 8x → -0.795029<, 8x → 11.5576<<
```

Το *Mathematica* διαθέτει τη συνάρτηση NSolve, με την οποία βρίσκει τις αριθμητικές (προσεγγιστικές) λύσεις μιας εξίσωσεις:

NSolve[eqn, x]	επιστρέφει αριθμητικές λύσεις της εξίσωσης eqn.

Έτσι στην περίπτωση που δεν μας ενδιαφέρουν οι ακριβείς λύσεις, τότε μπορούμε να εφαμόσουμε απ' ευθείας την συνάρτηση **NSolve** για να πάρουμε τις αριθμητικές (προσεγγιστικές λύσεις) λύσεις της εξίσωσης:

```
s = NSolve@b « 0, xD

88x → −0.795029<, 8x → 0.618718 − 0.698448 ą<,
 8x → 0.618718 + 0.698448 ą<, 8x → 11.5576<<
```

Επαλήθευση των ριζών της εξίσωσης με χρήση του τελεστή αντικατάσταση /.:

```
b ê. s

8−4.996 × 10^−16, 3.33067 × 10^−16 − 1.30451 × 10^−15 ą,
 3.33067 × 10^−16 + 1.30451 × 10^−15 ą, 0.<
```

Παρατηρούμε ότι οι τιμές του πολυωνύμου b στις ρίζες s είναι πολύ μικρές, γεγονός που δείχνει την ποιότητα της προσέγγισης, όμως δεν είναι μηδέν, δηλαδή δεν έχουμε ακριβείς ρίζες.

Όταν έχουμε μια εξίσωση 5ου βαθμού και άνω, συνήθως η συνάρτηση **Solve** δεν δίνει ικανοποιητική απάντηση.

Έστω το πολυώνυμο q:

```
q = 1 − 5 x + 7 x^2 + 10 x^3 − 2 x^4 − x^5

1 − 5 x + 7 x^2 + 10 x^3 − 2 x^4 − x^5
```

Υπολογισμός των ριζών με χρήση της συνάρτησης **Solve**:

```
Solve@q « 0, xD

88x → Root@−1 + 5 #1 − 7 #1^2 − 10 #1^3 + 2 #1^4 + #1^5 &, 1D<,
 8x → Root@−1 + 5 #1 − 7 #1^2 − 10 #1^3 + 2 #1^4 + #1^5 &, 2D<,
 8x → Root@−1 + 5 #1 − 7 #1^2 − 10 #1^3 + 2 #1^4 + #1^5 &, 3D<,
 8x → Root@−1 + 5 #1 − 7 #1^2 − 10 #1^3 + 2 #1^4 + #1^5 &, 4D<,
 8x → Root@−1 + 5 #1 − 7 #1^2 − 10 #1^3 + 2 #1^4 + #1^5 &, 5D<<
```

Παρατηρούμε ότι η συνάρτηση **Solve** μας δίνει 5 ρίζες, όμως η μορφή τους δεν είναι αυτή που θέλαμε. Σε αυτές τις περιπτώσεις μπορούμε να χρησιμοποιήσουμε τη συνάρτηση **NSolve** για να πάρουμε τις αριθμητικές (προσεγγιστικές) τιμές των ριζών:

```
NSolve@q « 0, xD
```

```
88x → -3.98547<, 8x → -1.11737<, 8x → 0.248221 - 0.156657 ą<,
 8x → 0.248221 + 0.156657 ą<, 8x → 2.6064<<
```

```
Clear@a, bD
```

Εκτός από τα πολυώνυμα, το *Mathematica* μπορεί να διαχειριστεί και ρητές συναρτήσεις. Άλλωστε οι ρητές συναρτήσεις είναι, όπως ξέρουμε, πηλίκο δύο πολυωνύμων, επομένως συνδέονται αρκετά στενα με τα πολυώνυμα. Συνήθη προβλήματα που αφορούν τις ρητές συναρτήσεις είναι η απλοποίηση κοινών παραγόντων από αριθμητή και παρανομαστή, ή η ανάλυση σε απλούστερα κλάσματα.

Το *Mathematica* διαθέτει αρκετές συναρτήσεις οι οποίες αντιμετωπίζουν αυτά τα συνήθη προβλήματα. Μερικές από αυτές είναι:

Apart[expr]	αναλύει τη ρητή παράσταση expr σε απλούστερα κατά το δυνατόν κλάσματα.

```
Apart@1 ê H3 + 10 x + 12 x^2 + 6 x^3 + x^4LD
```

$$\frac{1}{2\,(1+x)^3}-\frac{1}{4\,(1+x)^2}+\frac{1}{8\,(1+x)}-\frac{1}{8\,(3+x)}$$

```
Apart@H2 - 3 x^2L ê H2 - 3 x + 2 x^2 - 2 x^3 + x^5LD
```

$$-\frac{1}{6\,(-1+x)^2}-\frac{7}{9\,(-1+x)}-\frac{2}{9\,(2+x)}+\frac{1+2\,x}{2\,(1+x^2)}$$

Apart[expr, var]	το ίδιο με την συνάρτηση Apart[expr], με τη διαφορά ότι χρησιμοποιούνται μόνον οι μεταβλητές var, ενώ οι υπόλοιπες θεωρούνται ως σταθερές.

```
Apart@Hx - yL ê H6 + 5 x - 2 x^2 - x^3LD
```

$$-\frac{x}{-6-5\,x+2\,x^2+x^3}+\frac{y}{-6-5\,x+2\,x^2+x^3}$$

```
Apart[x - y] ê [6 + 5 x - 2 x^2 - x^3, x]
```

$$\frac{-1-y}{6\,(1+x)} + \frac{-2+y}{15\,(-2+x)} + \frac{3+y}{10\,(3+x)}$$

Together[expr] προσθέτει τους όρους της παράστασης expr και στο αποτέλεσμα επιχειρεί να απλοποιήσει κοινούς παράγοντες από αριθμητή και παρανομαστή.

```
Together[a ê b + x ê y]
```

$$\frac{b\,x + a\,y}{b\,y}$$

```
Together[(1 + Sqrt[2]) ê (a - Sqrt[2]) - (1 - Sqrt[2]) ê (a + Sqrt[2])]
```

$$-\frac{2\,\left(\sqrt{2} + \sqrt{2}\right)}{\left(\sqrt{2} - a\right)\left(\sqrt{2} + a\right)}$$

Collect[expr, x] από τους όρους της παράστασης expr βγάζει κοινό παράγοντα τις δυνάμεις του x.

```
Collect[x^4 + (a + y) (x + y^3) x^2 + (y^3 + 3) x^2, x]
```

$$x^4 + x^3\,(a + y) + x^2\,(3 + y^3 + y^3\,(a + y))$$

```
Collect[x^4 + (a + y) (x + y^3) x^2 + (y^3 + 3) x^2, y]
```

$$3\,x^2 + a\,x^3 + x^4 + x^3\,y + (x^2 + a\,x^2)\,y^3 + x^2\,y^4$$

```
Collect[x^4 + (a + y) (x + y^3) x^2 + (y^3 + 3) x^2, {x, y}]
```

$$x^4 + x^3\,(a + y) + x^2\,(3 + (1 + a)\,y^3 + y^4)$$

Cancel[expr] απλοποιεί κοινούς παράγοντες από αριθμητή και παρανομαστή στην παράσταση expr.

```
Cancel@Hx^3 - 4 xL ê Hx^3 - 2 x^2L + Hx^2 - 3 x + 2L ê Hx^2 - 1LD
```

$$\frac{-2+x}{1+x}+\frac{2+x}{x}$$

```
Together@Hx^3 - 4 xL ê Hx^3 - 2 x^2L + Hx^2 - 3 x + 2L ê Hx^2 - 1LD
```

$$\frac{2+x+2x^2}{x(1+x)}$$