1ο. Η αριθμητική του υπολογιστή

1.1 Τί είναι Αριθμητική Ανάλυση

Υπάρχουν πολλά προβλήματα στη μαθηματική επιστήμη για τα οποία δεν υπάρχουν αναλυτικές εκφράσεις λύσεων. Στις περιπτώσεις αυτές έχουν αναπτυχθεί παράλληλοι μέθοδοι οι οποίες μας δίνουν προσεγγιστικές τιμές των λύσεων. Υπάρχουν διάφοροι τρόποι προσέγγισης των λύσεων. Για παράδειγμα, βρίσκουμε μια άλλη έκφραση του προβλήματος που μας δίνει μια λύση η οποία απλά προσεγγίζει την πραγματική λύση. Με την εξέλιξη των υπολογιστών και με τη δυνατότητα να γίνονται εκατομμύρια πράξεις το δευτερόλεπτο αναπτύχθηκε παράλληλα και η επιστήμη της αριθμητικής λύσης πολλών προβλημάτων. Η επιστήμη αυτή καλείται Αριθμητική Ανάλυση. Παρόλη πράγματι τη μεγάλη ταχύτητα που εκτελούνται οι αριθμητικές πράξεις ακόμα υπάρχουν προβλήματα σε πολλές επιστημονικές εφαρμογές που περιλαμβάνουν πολύπλοκες και χρονοβόρες αριθμητικές πράξεις. Αρκεί να ειπωθεί, για παράδειγμα, πως για τη λύση ενός γραμμικού συστήματος εκατό εξισώσεων με εκατό αγνώστους, με τη γνωστή μέθοδο των οριζουσών και για την εκτέλεση των απαραίτητων αριθμητικών πράξεων από έναν υπολογιστή που κάνει δύο εκατομμύρια πράξεις το δευτερόλεπτο, απαιτούνται $5x10^{144}$ αιώνες. Υπάρχουν όμως άλλες μέθοδοι οι οποίες για το ίδιο σύστημα με τον ίδιο υπολογιστή χρειάζονται σχεδόν μόνο εκατό περίπου δευτερόλεπτα.

Η Αριθμητική ανάλυση είναι το μέρος εκείνο της μαθηματικής επιστήμης που ασχολείται με την ανάπτυξη και εκτίμηση μεθόδων που υπολογίζουν αριθμητικά αποτελέσματα από γνωστά αριθμητικά δεδομένα.

Δύο είναι τα βασικά κριτήρια μιας αριθμητικής μεθόδου: (α) η ταχύτητα και (β) η ακρίβεια. Με τον όρο ταχύτητα εννοούμε τον υπολογιστικό χρόνο που χρειάζεται η Κεντρική Μονάδα Επεξεργασίας για να μας δώσει τα αριθμητικά αποτελέσματα. Η ακρίβεια συνδέεται με τα σφάλματα που παράγονται είτε με τον τρόπο που αποθηκεύονται τα δεδομένα στον υπολογιστή είτε με τα σφάλματα που οφείλονται στην ίδια τη μέθοδο.

Η διαδικασία της λύσης ενός προβλήματος που περιλαμβάνει πολύπλοκες αριθμητικές πράξεις συνίσταται από τέσσερα κύρια στάδια:

1. ***Διαμόρφωση του προβλήματος***. Κατ' αρχήν το πρόβλημα διαμορφώνεται ως προς τη μαθηματική του έκφραση, δηλαδή σαν ένα σύνολο μαθηματικών σχέσεων.
2. ***Επιλογή της μεθόδου***. Επιλέγεται μία ή συνδυασμός αριθμητικών μεθόδων, που είναι κατάλληλες για να δώσουν την καλύτερη προσεγγιστική λύση στο συγκεκριμένο πρόβλημα.
3. ***Προγραμματισμός και κωδικοποίηση***. Κατ' αρχάς καταστρώνουμε βήμα προς βήμα όλη την υπολογιστική διαδικασία σύμφωνα με την αριθμητική μέθοδο. Η διαδικασία αυτή είναι γνωστή και ως αλγόριθμος. Κατόπιν μεταφράζουμε την όλη υπολογιστική διαδικασία σε ένα σύνολο εντολών ορισμένης "γλώσσας", όπως π.χ. C, FORTRAN, BASIC, PASCAL, C++, PHP, JAVA κ.λ.π. Το σύνολο αυτό των εντολών καλείται πρόγραμμα.
4. ***Εκτέλεση του προγράμματος***. Το πρόγραμμα αφού "φορτωθεί" στη μνήμη του υπολογιστή, διέρχεται από την Κεντρική Μονάδα Επεξεργασίας (μερικές φορές συνηθίζεται να λέγεται ότι "τρέχουμε" το πρόγραμμα) και μας δίνει τα αντίστοιχα αποτελέσματα.

Στην εποχή μας με τη μεγάλη εξάπλωση των υπολογιστών είναι πολύ εύκολο να κατανοεί κανείς έννοιες της Αριθμητικής Ανάλυσης μόνος του, χωρίς να εννοούμε με αυτό πως καταργείται η αλληλεπίδραση με τον δάσκαλο. Το βιβλίο αυτό στοχεύει ακριβώς σε ένα νέο τρόπο διδασκαλίας που προσδιορίζεται από τη νέα διάσταση των δυνατοτήτων που έχει ένας υπολογιστής. Σε κάθε κεφάλαιο θα περιέχονται πολλά μικρά προγράμματα για να κατανοούνται καλύτερα οι έννοιες και οι αριθμητικές μέθοδοι που θα αναπτύσσονται. Συνήθως τα περισσότερα βιβλία Αριθμητικής Ανάλυσης αναγράφουν μόνο τους αλγορίθμους των αριθμητικών μεθόδων. Η μετάφραση όμως ενός αλγορίθμου σε πρόγραμμα δεν είναι μια εύκολη διαδικασία και επειδή το βιβλίο αυτό στοχεύει στην καλύτερη κατανόηση των αριθμητικών μεθόδων και όχι σε ασκήσεις προγραμματισμού θα παραθέτει προγράμματα γραμμένα στη γλώσσα BASIC. Εύκολα ένας προγραμματιστής θα μπορούσε να μεταφράσει τον κώδικα της BASIC σε μια άλλη γλώσσα. Εξάλλου η δομή όλων των προγραμματιστικών γλωσσών είναι ίδια.

1.2 Συστήματα αρίθμησης

Βασικό εργαλείο της Αριθμητικής Ανάλυσης είναι ο υπολογιστής. Είναι δε προφανές πως όποιος ασχολείται με εφαρμογές αριθμητικών μεθόδων σε διάφορα προβλήματα θα πρέπει να γνωρίζει α) τον τρόπο με τον οποίο αποθηκεύονται οι αριθμοί στη μνήμη του, β) ποια είναι τα σφάλματα της αποθήκευσης, γ) πώς εκτελούνται οι αριθμητικές πράξεις και δ) πώς μεταδίδονται τα σφάλματα κατά την εκτέλεση των αριθμητικών πράξεων και γενικά πώς "σκέπτεται" αριθμητικά ο υπολογιστής. Η γνώση των παραπάνω είναι η στοιχειώδης προϋπόθεση για να κάνει κάποιος τη σωστή επιλογή της αριθμητικής μεθόδου που πρόκειται να χρησιμοποιήσει στο πρόβλημά του. Τα δύο επόμενα κεφάλαια αναφέρονται στην αριθμητική του υπολογιστή και τα διάφορα σφάλματα που παράγονται.

Δεκαδικό σύστημα

Οι αριθμοί που χρησιμοποιούμε στην καθημερινή μας ζωή βασίζονται στο δεκαδικό σύστημα αρίθμησης το οποίο περιέχει δέκα ψηφία που παρίστανται με τα σύμβολα

0,1,2,3,4,5,6,7,8,9

και καλούνται δεκαδικά ψηφία.

Κάθε ακέραιος αριθμός Α στο δεκαδικό σύστημα αρίθμησης ερμηνεύεται ως ένα άθροισμα πολλαπλασίων ακεραίων δυνάμεων του 10, το οποίο καλείται βάση του συστήματος. Για παράδειγμα ο ακέραιος αριθμός Α=7682 γράφεται

$$7682=7x10^3 + 6x10^2 + 8x10^1 + 2x10^0$$

Κάθε πραγματικός αριθμός x μπορεί να γραφεί με τη μορφή

$$x=x_A+x_K$$

όπου x_A είναι ο μεγαλύτερος ακέραιος αριθμός από τους μικρότερους ή ίσους του x και x_K το κλασματικό μέρος. Το κλασματικό μέρος ερμηνεύεται ως ένα άθροισμα πολλαπλασίων αρνητικών ακεραίων δυνάμεων του 10. Για παράδειγμα το κλασματικό μέρος του αριθμού x=234.6235 γράφεται

$$0.6235=6x10^{-1}+2x10^{-2}+3x10^{-3}+5x10^{-4}$$

Σύμφωνα με τα παραπάνω ο αριθμός x μπορεί να γραφεί με τη μορφή.

$234.6235=2x10^{2}+3x10^{1}+4x10^{0}+6x10^{-1}+2x10^{-2}+3x10^{-3}+5x10^{-4}$

ή ισοδύναμα

$(0.2346235)x10^{3}$

που καλείται εκθετική παράσταση. Ο αριθμός που περιέχεται μέσα στην παρένθεση λέγεται ***κλασματικό μέρος***, το 10 βάση και το 3 ***εκθέτης***. Το πρώτο ψηφίο μετά την υποδιαστολή δεν πρέπει ποτέ να είναι μηδέν. Όλα τα ψηφία του κλασματικού μέρους καλούνται ***σημαντικά ψηφία***. Θα πρέπει να τονισθεί ιδιαίτερα ότι το 0 ως πρώτο ψηφίο δεν είναι σημαντικό. Για παράδειγμα η εκθετική παράσταση του αριθμού 0.00035643 είναι $(0.35643)x10^{-3}$ Γενικά θα μπορούσαμε να πούμε ότι κάθε αριθμός x με βάση οποιονδήποτε αριθμό β γράφεται

$$x = (0.d1\ d2\ d3 \ldots dk \ldots)\ \beta^{e}$$

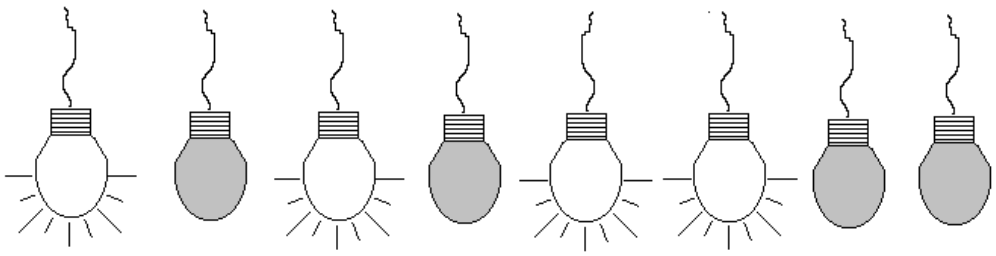
Όπου $0.d_1\ d_2\ d_3 \ldots d_k \ldots$ είναι το κλασματικό μέρος με d_1 διάφορο του μηδενός, β είναι η ***βάση*** του συστήματος, e ο εκθέτης και τα $d_1, d_2, d_3, \ldots, d_k$, ψηφία του συστήματος.

Αν $d_1=0$, τότε και $d_2=d_3= \ldots =d_k= \ldots =0$

Δυαδικό σύστημα

Ο υπολογιστής για καθαρά πρακτικούς λόγους εργάζεται με το σύστημα αρίθμησης με βάση το δύο, δηλαδή αυτό που καλούμε δυαδικό σύστημα. Κι αυτό γιατί είναι δυνατόν να αναπαρίσταται γεγονότα εκφρασμένα με δυαδική έκφραση.

Για παράδειγμα,



Στο σχήμα αυτό παρουσιάζονται σε σειρά οκτώ ηλεκτρικοί λαμπτήρες εκ των οποίων ορισμένοι είναι σε λειτουργία και ορισμένοι όχι. Αν παραστήσουμε με το ψηφίο **1** αυτούς που βρίσκονται σε λειτουργία και με **0** αυτούς που δε βρίσκονται σε λειτουργία, τότε έχουμε το δυαδικό αριθμό

1 0 1 0 1 1 0 0

του οποίου αντίστοιχος αριθμός στο δεκαδικό είναι

$1x2^7+0x2^6+1x2^5+0x2^4+1x2^3+1x2^2+0x2^1+0x2^0$

$=128+0+32+0+8+4+0+0=172$

Βέβαια οι υπολογιστές δε χρησιμοποιούν ηλεκτρικούς λαμπτήρες αλλά καταστάσεις διαφοράς δυναμικού. Για παράδειγμα, μπορεί να χρησιμοποιείται το 1 για διαφορά δυναμικού 5 Volt και 0 για 0 Volt. Είναι προφανές πως η μεγάλη ταχύτητα επεξεργασίας πληροφοριών εξαρτάται από τη ταχύτητα με την οποία εναλλάσσεται η δυαδική έκφραση.

Μετατροπή δεκαδικού αριθμού σε δυαδικό

Η μετατροπή ενός αριθμού του δυαδικού συστήματος σε αριθμό του δεκαδικού είναι πολύ εύκολη διαδικασία, ενώ το αντίστροφο είναι πιο πολύπλοκο. Αν για παράδειγμα έχουμε τον αριθμό 183.703125 ακολουθούμε διαφορετική διαδικασία για το ακέραιο και για το κλασματικό μέρος.

Σε ότι αφορά το ακέραιο μέρος, διαιρούμε τον αριθμό δια του 2 και κρατάμε το υπόλοιπο που θα είναι προφανώς ή 0 ή 1. Συνεχίζουμε την ίδια διαδικασία με το προκύπτον πηλίκο. Η διαδικασία τερματίζεται μέχρι που το πηλίκο να γίνει 0. Τα υπόλοιπα των διαιρέσεων με διάταξη από το τέλος στην αρχή είναι η παράσταση του αριθμού στο δυαδικό σύστημα.

Για το κλασματικό μέρος ακολουθείται άλλη διαδικασία. Πολλαπλασιάζουμε το κλασματικό μέρος με το 2. Κρατάμε το ακέραιο μέρος -που προφανώς θα είναι 1 ή 0 και πολλαπλασιάζουμε το νέο κλασματικό μέρος. Η ίδια διαδικασία συνεχίζεται μέχρις ότου το κλασματικό μέρος γίνει 0. Αν το κλασματικό μέρος δεν γίνεται ποτέ 0 η διαδικασία τερματίζεται μετά από συγκεκριμένο πλήθος ψηφίων.

Παράδειγμα

Έστω ότι θέλουμε να μετατρέψουμε τον αριθμό 183.703125 σε δυαδικό. Σύμφωνα με όσα αναφέραμε παραπάνω θα ακολουθηθεί διαφορετική διαδικασία για το ακέραιο μέρος 183 και διαφορετική για το κλασματικό 0.703125. Επειδή η διαδικασία είναι πολύπλοκη παραθέτουμε ένα πρόγραμμα που μετατρέπει δεκαδικούς αριθμούς σε αντίστοιχους δυαδικούς. Στο πρόγραμμα δίνεται πρώτα το ακέραιο μέρος και μετά το κλασματικό. Στην περίπτωση που το πλήθος των δυαδικών ψηφίων του κλασματικού μέρους μαζί με το πλήθος των ακεραίων υπερβαίνει τον αριθμό 24, η διαδικασία σταματάει και έχουμε σφάλμα αποκοπής. Ο αριθμός 24 δεν είναι τυχαίος, αλλά η επιλογή του εξηγηθεί στην επόμενη παράγραφο. Για το συγκεκριμένο παράδειγμα έχουμε Ακέραιο μέρος 183 Κλασματικό μέρος 0.703125

Δηλαδή ο αριθμός είναι 10110111.101101

Το αντίστοιχο πρόγραμμα που μετατρέπει δεκαδικούς αριθμούς στο δυαδικό σύστημα είναι:

Ακέραιο μέρος		Δεκαδικό μέρος	
2)183	1	0.703125	
2)91	1	2	
2)45	1	1.406250	1
2)22	0	2	
2)11	1	0.8125	0
2)5	1	2	
2)2	0	1.625	1
2)1	1	2	
		1.25	1
		2	
		0.5	0
		2	
		1.0	1

Συμπλήρωμα δυαδικών αριθμών

Πριν περιγράψουμε τον τρόπο με τον οποίο αποθηκεύονται οι αριθμοί στη μνήμη του υπολογιστή, θα πρέπει πρώτα να γίνουν κατανοητές ορισμένες έννοιες που είναι γνωστές στη συνήθη αριθμητική.
Καλούμε συμπλήρωμα ενός δυαδικού αριθμού Α τον δυαδικό αριθμό που προκύπτει αν αφαιρέσουμε κάθε ψηφίο του Α από το 1. Για παράδειγμα

Δυαδικός αριθμός Α	110011001100
	111111111111
	110011001100
Συμπλήρωμα του Α	001100110011

Αν τώρα έχουμε να αφαιρέσουμε δυο δυαδικούς αριθμούς Α και Β, μπορεί η πράξη της αφαίρεσης να μετατραπεί σε πρόσθεση. Π.χ. αν έχουμε Α=111010 και Β=101101 και θέλουμε να υπολογίσουμε τη διαφορά Υ=Α-Β με τον κλασικό τρόπο, έχουμε

Α	111010
Β	101101
Υ	001101

Αν βρούμε το συμπλήρωμα του Β που είναι 111111101101=010010 και τον προσθέσουμε στον Α, θα έχουμε

Α	1 1 1 0 1 0
Συμπλ. Β	0 1 0 0 1 0
	1 0 0 1 1 0 0

και προσθέτοντας το 1 που υπάρχει στην έβδομη θέση σαν μονάδα θα έχουμε το αποτέλεσμα που είναι 1101. Στην περίπτωση που δεν υπάρχει 1 στην έβδομη θέση, τότε σημαίνει ότι ο Β είναι μεγαλύτερος από τον Α και το αποτέλεσμα είναι αρνητικό. Το δε αποτέλεσμα του Α με το συμπλήρωμα του Β δεν είναι η διαφορά Α-Β. Για να βρούμε τη διαφορά Α-Β -που είναι αρνητικός αριθμός- βρίσκουμε το συμπλ.(Α+συμπλ.Β). Για παράδειγμα αν Α=101010 και Β=110011 τότε

Α	101010
Συμπλ. Β	001100

A-B 110110

Παρατηρούμε ότι δεν υπάρχει ψηφίο στην έβδομη θέση και κατά συνέπεια η διαφορά A-B είναι αρνητικός αριθμός και ισούται με το συμπλήρωμα του 110110 δηλαδή -001001.

1.3 Αριθμοί κινητής υποδιαστολής

Όπως είπαμε στην προηγούμενη παράγραφο κάθε κλασματικός πραγματικός αριθμός x γράφεται με μοναδικό τρόπο στην εκθετική του μορφή

$x = \pm (0.d_1\ d_2\ d_3 \ldots d_n\ d_{n+1} \ldots) x \beta^e$

όπου β η βάση, e ο εκθέτης και $a=0.\ d_1\ d_2\ d_3 \ldots d_n\ d_{n+1}.$ το κλασματικό μέρος με d_1 πάντοτε διάφορο του μηδενός.

Το πλήθος των ψηφίων του κλασματικού μέρους θα μπορεί να είναι άπειρο. Ο αριθμός x=2/3 του οποίου το κλασματικό μέρος είναι

a=0.666666666666 . . . . . .

όταν αυτός πρόκειται να αποθηκευθεί στη μνήμη ενός υπολογιστή είναι προφανές ότι το πλήθος των ψηφίων του κλασματικού μέρους θα πρέπει να είναι ένας πεπερασμένος αριθμός. Έτσι ο a δεν αποθηκεύεται ακριβώς αλλά κατά προσέγγιση. Αν n είναι το πλήθος των ψηφίων που μπορεί να αποθηκευθούν θα πρέπει τα ψηφία d_{n+1} και πάνω να αποκοπούν.

Υπάρχουν δύο είδη αποκοπής α) η κοπή και β) η στρογγύλευση.

1. κατά την κοπή το ψηφίο d_n παραμένει όπως έχει ανεξάρτητα από το μέγεθος του d_{n+1} ενώ
2. κατά τη στρογγύλευση και συγκεκριμένα στο δεκαδικό σύστημα, ο d_n αυξάνεται κατά μία μονάδα αν ο d_{n+1} είναι 5,6,7,8,9 και παραμένει όπως έχει αν ο d_{n+1} είναι 0,1,2,3,4. Στην περίπτωση του δυαδικού συστήματος ο d_n αυξάνεται κατά 1 όταν ο d_{n+1} είναι 1 και παραμένει όπως έχει αν ο d_{n+1} είναι 0.

Θα καλούμε αριθμό κινητής υποδιαστολής μήκους n την εκθετική παράσταση ενός αριθμού x αλλά με πλήθος ψηφίων του κλασματικού μέρους n. Στη γενική του μορφή ένας αριθμός κινητής υποδιαστολής μπορεί να γραφεί ως

$$x = \sigma a \beta^e$$

όπου σ=+1 ή -1, α=0. d_1 d_2 d_3 . . .d_n d_{n+1}. . το κλασματικό μέρος ή mantissa, n μήκος, β η βάση και e ένας ακέραιος αριθμός που καλείται εκθέτης. Θα πρέπει $d_1 \neq 0$. Αν $d_{1,}=0$, τότε και $d_2=d_3= . . . =d_k= . . . =0$

Για τη mantissa α ισχύει $0.1 \leq \alpha \prec 1$ στο δεκαδικό σύστημα και $(0.1)_2 \leq \alpha \prec 1$ (ή ισοδύναμα σε δεκαδική έκφραση $0.5 \leq \alpha \prec 1$) στο δυαδικό σύστημα.

Παράδειγμα

Να μετατραπούν οι αριθμοί x=4/3, y=5/9 και z=1/33 σε αριθμούς κινητής υποδιαστολής μήκους n=7. Η αποκοπή των στοιχείων να γίνει α) με κοπή και β) με στρογγύλευση

Απάντηση:

α) Κοπή

x*=0.6666666101, y*=0.5555555 100 και z*=0.3030303 10-1

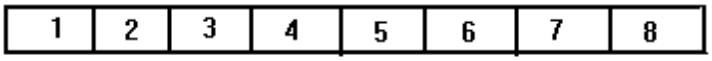
β) Στρογγύλευση

x*=0.6666667101, y*=0.5555556 100 και z*=0.3030303 10-1**1.4**

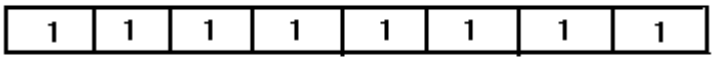
Κώδικες

Το σύνολο των κανόνων που διέπουν τον τρόπο διάταξης των δυαδικών ψηφίων ώστε, να παριστάνουν χαρακτήρες γραμμάτων, ψηφία και σύμβολα καλείται κώδικας. Κάθε τύπος υπολογιστή χρησιμοποιεί έναν ορισμένο τύπο κώδικα όπως για παράδειγμα οι πιο σπουδαίοι σήμερα κώδικες είναι οι EBCDIC (Extended Binary-Coded Decimal Intercgange Code) και ASCII-8 (American Stand- and Code for Information Interchange). Και οι δυο παραπάνω κώδικες χρησιμοποιούν bytes των 8 bits.

Ας υποθέσουμε ότι έχουμε μια σειρά από οκτώ δυαδικές θέσεις (bits) όπως φαίνεται στο παρακάτω σχήμα.



Κάθε δυαδική θέση (bit) μπορεί να εναλάσσεται με τα ψηφία του δυαδικού συστήματος 0 και 1. Είναι προφανές ότι ο μεγαλύτερος φυσικός αριθμός που μπορεί να παρασταθεί είναι αυτός που όλες οι δυαδικές θέσεις θα έχουν το ψηφίο 1. Δηλαδή,



που δεν είναι άλλος από τον αριθμό

$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 = 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 255$

μπορεί να δεχθεί ακέραιους αριθμούς από το 0 έως το 255 δηλαδή πλήθος αριθμών 256. Αν κάθε ακέραιος αντιστοιχεί σε ένα σύμβολο - σύμβολο μπορεί να είναι γράμμα του Λατινικού ή ελληνικού αλφαβήτου ή επίσης ψηφίο του δεκαδικού συστήματος ή οποιοδήποτε άλλο στοιχείο - τότε μπορεί να συμβολισθούν συνολικά 256 σύμβολα.

Το σύνολο των οκτώ δυαδικών θέσεων καλείται ***χαρακτήρας*** (***byte***) και αποτελεί τη βασική μονάδα μνήμης στους μικροϋπολογιστές.

Στον κώδικα ASCII ένας χαρακτήρας (byte) χωρίζεται σε δυο μέρη των 4 bits το καθένα. Το πρώτο μέρος καλείται ζώνη και το δεύτερο αριθμητικό 8-4-2-1 όπως φαίνεται και στο παρακάτω σχήμα.

ζώνη				αριθμητικό			
z	z	z	z	8	4	2	1

Η ζώνη και το αριθμητικό μέρος διαθέτουν το καθένα 4 δυαδικές θέσεις (bits) και κατά συνέπεια μπορεί να πάρει αριθμούς από 0-15 δηλαδή όσα είναι τα ψηφία του δεκαεξαδικού συστήματος. Μαζί η ζώνη και το δεκαδικό μέρος μπορεί να συμβολίζονται με διψήφια ψηφία του δεκαεξαδικού συστήματος με το πρώτο να συμβολίζει τη ζώνη και το δεύτερο το αριθμητικό.

Προκειμένου να συμβολίζονται ψηφία του δεκαδικού συστήματος και επειδή είναι δέκα σε αριθμό (λιγότερα από 16) χρησιμοποιείται ο ίδιος αριθμός για τη ζώνη και διαφορετικός στο αριθμητικό. Τα γράμματα του Λατινικού αλφαβήτου επειδή υπερβαίνουν τον αριθμό 16 οι πρώτοι 16 χαρακτήρες θα έχουν την ίδια ζώνη και αριθμητικό που θα

αυξάνεται κατά μονάδα και οι υπόλοιποι χαρακτήρες θα έχουν διαφορετική ζώνη. Το ίδιο θα συμβαίνει και για τα σύμβολα.

Στους παρακάτω πίνακες φαίνεται πως συμβολίζεται κάθε χαρακτήρας είτε είναι ψηφίο είτε Λατινικό γράμμα. Με παρόμοιο τρόπο συμβολίζεται και τα υπόλοιπα σύμβολα.

Στους παρακάτω πίνακες ενδεικτικά αναφέρονται ορισμένοι από τους χαρακτήρες του κώδικα ASCII 8 που αντιστοιχούν στους φυσικούς αριθμούς από το 0 έως το 255. Είναι εύκολο να αντιληφθεί κανείς με τον παραπάνω τρόπο μπορούμε να αποθηκεύσουμε μόνο αλφαριθμητικά στοιχεία και όχι αριθμούς που θα μας επιτρέπουν να εκτελούμε αριθμητικές πράξεις. Πριν όμως προχωρήσουμε στον τρόπο με τον οποίο μπορούμε να αποθηκεύσουμε αριθμούς θα μιλήσουμε πρώτα για τους όρους δεδομένα (data) και πληροφορίες (information).

Σύμβολο	Ζώνη	8-4-2-1	16/δικό
A	1010	0001	A1
B	1010	0010	A2
C	1010	0011	A3
D	1010	0100	A4
E	1010	0101	A5
F	1010	0110	A6
G	1010	0111	A7
H	1010	1000	A8
I	1010	1001	A9
J	1010	1010	AA
K	1010	1011	AB
L	1010	1100	AC
M	1010	1101	AD
N	1010	1110	AE
O	1010	1111	AF
P	1011	0000	B0

Q	1011	0001	B1
R	1011	0010	B2
S	1011	0011	B3
T	1011	0100	B4
U	1011	0101	B5
V	1011	0110	B6
W	1011	0111	B7
X	1011	1000	B8
Y	1011	1001	B9
Z	1011	1010	BA

Σύμβολο	Ζώνη	8-4-2-1	16/δικό
0	0101	0000	50
1	0101	0001	51
2	0101	0010	52
3	0101	0011	53
4	0101	0100	54
5	0101	0101	55
6	0101	0110	56
7	0101	0111	57
8	0101	1000	58
9	0101	1001	59

1.4 Τύποι δεδομένων

Με πολύ απλά λόγια θα μπορούσαμε να πούμε πως υπολογιστής είναι ένας **επεξεργαστής πληροφοριών**. Τί ενοούμε όμως με τον όρο **πληροφορία** στην επιστήμη του υπολογιστή; Ο συνήθης ορισμός της πληροφορίας είναι:

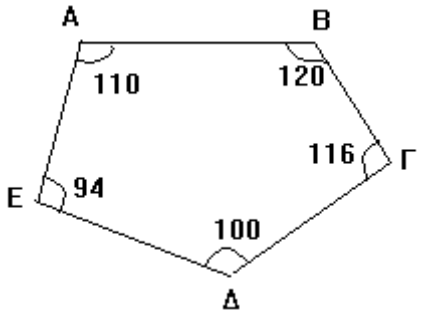
Πληροφορία (information) = Δεδομένα (data) + Δομή (structure)

Για να γίνει πιο αντιληπτή αυτή η σχέση θα αναφέρουμε το εξής παράδειγμα,

Θεωρούμε τους παρακάτω αριθμούς

110, 120, 116, 100, 94

Οι οποίοι αριθμοί είναι ορισμένα **δεδομένα** (data) χωρίς συγκεκριμένη σημασία. Αν όμως ορίσουμε μια **δομή** (structure) στα δεδομένα, για παράδειγμα έστω ότι οι παραπάνω αριθμοί παριστάνουν γωνίες σε μοίρες ενός πενταγώνου



τότε οι αριθμοί έχουν δομή και αποκτούν σημασία, δηλαδή έγιναν **πληροφορίες** (information). Ο υπολογιστής μάλιστα βασισμένος στις πληροφορίες αυτές μπορεί να σχεδιάσει ένα τέτοιο πεντάγωνο.

Πολλές φορές αντί του όρου “επιστήμη του υπολογιστή” χρησιμοποιούμε τον όρο “επιστήμη της πληροφορικής” και ο οποίος όρος προέρχεται από τη λέξη **πληροφορία** με τον τρόπο που η έννοια αυτή ορίσθηκε παραπάνω..

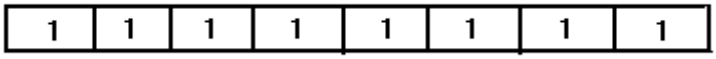
Τύποι δεδομένων μπορεί να είναι αριθμοί, γράμματα, σύμβολα, ενέργειες, γεγονότα κτλ. Σε κάθε γλώσσα υπάρχουν διάφοροι τύποι δεδομένων. Η Quick Basic για παράδειγμα χρησιμοποιεί τρία κύρια είδη τύπων δεδομένων. 1) Συρμοί ή αλφαριθμητικά δεδομένα που μπορεί να είναι μεταβλητού ή σταθερού μήκους, 2) ακέραιοι που μπορεί να είναι μικροί ή μεγάλοι ακέραιοι αριθμοί, 3) πραγματικοί αριθμοί (αριθμοί κινητής υποδιαστολής) που μπορεί να είναι απλής

και διπλής ακρίβειας. Η Quick Basic εκτός από τους παραπάνω τύπους δεδομένων διαθέτει και τις Δομές Δεδομένων η δυνατότητα δηλαδή, με ένα όνομα μεταβλητής να αναπαριστάνονται πολλές τιμές δεδομένων.

Στο βιβλίο αυτό μας ενδιαφέρει να αναπτύξουμε μόνο τα δεδομένα των ακεραίων αριθμών και των πραγματικών. Η διάκριση σε ακέραιους και πραγματικούς οφείλεται στο διαφορετικό τρόπο με τον οποίο αποθηκεύονται στον υπολογιστή.

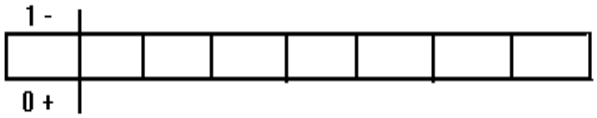
1.5 Αποθήκευση ακεραίων αριθμών

"Έστω ότι εργαζόμαστε σε έναν μικροϋπολογιστή με χαρακτήρες των 8-bits. Το μεγαλύτερο αριθμό που μπορεί να παραστήσει ένας χαρακτήρας είναι αυτός που σε κάθε δυαδική θέση έχει για ψηφίο το 1 δηλαδή

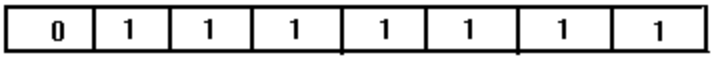


$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 = 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 255$

δηλαδή ένας χαρακτήρας μπορεί να παραστήσει θετικούς ακεραίους από το 0 έως το 255. Αν θέλουμε όμως με το ένα χαρακτήρα να παριστάνονται θετικοί και αρνητικοί ακέραιοι αριθμοί, τότε μια δυαδική θέση πρέπει να διατεθεί για το πρόσημο. Αν, για παράδειγμα, η πρώτη δυαδική θέση διατεθεί για το πρόσημο (με το 0 συμβολίζεται το (+) και με το 1 το (-)).

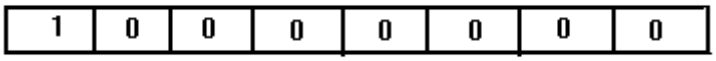


τότε ο μεγαλύτερος θετικός ακέραιος αριθμός που μπορεί να αποθηκευθεί θα είναι



δηλαδή $2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 127$

Ο μεγαλύτερος ακέραιος αριθμός είναι ο 127. Η παράσταση του μικρότερου ακέραιου θα είναι το συμπλήρωμα του μεγαλύτερου θετικού ακέραιου δηλαδή του 0111111 που δεν είναι άλλος από τον



2^8 = -128

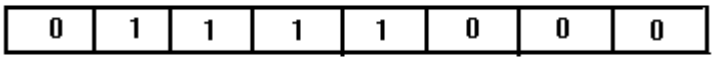
Συνήθως στους περισσότερους μικροϋπολογιστές το 1 της πρώτης στήλης εκτός του ότι συμβολίζει το αρνητικό πρόσημο προστίθεται και σαν μονάδα. Έτσι ο μικρότερος αρνητικός ακέραιος είναι -128.

Παράδειγμα

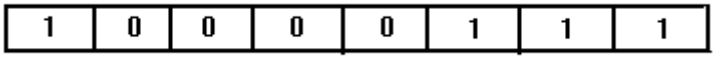
Θέλουμε να αποθηκεύσουμε το -120 σε έναν χαρακτήρα των 8 bits.

Απάντηση

Καταρχάς ο αριθμός αυτός μπορεί να αποθηκευθεί γιατί βρίσκεται στο διάστημα [-128, 127]. Σύμφωνα με αυτά που είπαμε παραπάνω, επειδή ο αριθμός είναι αρνητικός, θα αποθηκευθεί το συμπλήρωμα του 120. Ο αριθμός 120 σε χαρακτήρα 8- bits γράφεται

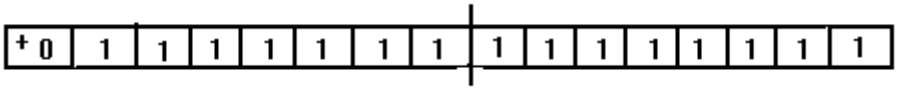


Το συμπλήρωμά του είναι

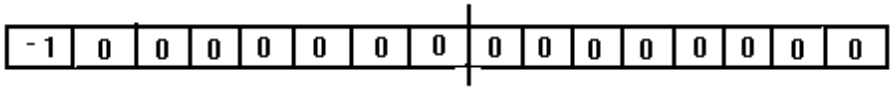


Από τα παραπάνω φαίνεται ότι το συμπλήρωμα κάθε θετικού ακεραίου αριθμού θα έχει στη πρώτη δυαδική θέση πάντοτε το 1 που θα δηλώνει και το αρνητικό πρόσημο.

Συνήθως για την αποθήκευση των ακεραίων αριθμών χρησιμοποιούνται δύο χαρακτήρες μαζί. Η πρώτη δυαδική θέση διατίθεται για το πρόσημο. Συνεπώς ο μεγαλύτερος θετικός ακέραιος αριθμός που μπορεί να παρασταθεί θα είναι



δηλαδή $2^{14}+2^{13}+2^{12}+ \ldots +2^{1}+2^{0}=2^{15}-1=32767$ Ο μικρότερος αρνητικός ακέραιος θα είναι το συμπλήρωμα του παραπάνω αριθμού δηλ.



ο οποίος ισούται με $-2^{15}=-32768$.

Ακέραιοι αριθμοί που βρίσκονται εκτός του διαστήματος

[-32768, 32767]

δεν είναι δυνατόν να αποθηκευθούν και ο υπολογιστής δίνει μήνυμα "overflow error".

1.6 Αποθήκευση κλασματικών αριθμών

Αν x είναι ένας πραγματικός κλασματικός αριθμός, τότε αντιστοιχεί στο x ένας αριθμός x* κινητής υποδιαστολής μήκους n ο οποίος και προσεγγίζει το x στα n πρώτα σημαντικά ψηφία.. Δηλαδή

$x^*=\sigma\alpha^*2^e$.

Έστω ότι για την αποθήκευση ενός αριθμού κινητής υποδιαστολής χρησιμοποιούνται συνολικά τέσσερις χαρακτήρες - όπως συνήθως συμβαίνει στις περισσότερες γλώσσες προγραμματισμού-.

Από τους τέσσερις χαρακτήρες ο ένας χρησιμοποιείται για την αποθήκευση του εκθέτη e και οι υπόλοιποι τρεις για τη mantissa α*. Αφού ο e είναι ακέραιος, τότε σύμφωνα με αυτά που είπαμε στην προηγούμενη παράγραφο ο e θα μπορεί να παίρνει τιμές

$-128 \le e \le 127$

Επειδή επίσης ισχύει $0.1 \le \alpha^* < 1$ ο μικροϋπολογιστής μπορεί να χειρίζεται αριθμούς που οι τιμές τους βρίσκονται στο διάστημα

$2^{-128} \le x \le 2^{127}$ ή ισοδύναμα $2.9\ 10^{-39} \le x \le 1.7\ 10^{38}$

Αριθμοί εκτός των παραπάνω ορίων δεν μπορούν να αποθηκευθούν και ο υπολογιστής δίνει το μήνυμα overflow error.

Για την αποθήκευση της mantissa α* χρησιμοποιούνται τρεις χαρακτήρες ή ισοδύναμα 24 δυαδικές θέσης μνήμης ή 24 bits. Επειδή το πρώτο ψηφίο της mantissa είναι πάντοτε 1 (σαν σημαντικό ψηφίο) αυτό θα εννοείται και τη θέση του θα καταλαμβάνει το πρόσημο. Δηλαδή 0 για (+) και 1 για (-).

Παράδειγμα

Δίνεται ο αριθμός 328.625 και ζητούμε να παρασταθεί η αποθήκευσή του σε έναν υπολογιστή που χρησιμοποιεί 4 χαρακτήρες των 8 bits.

Απάντηση

Χρησιμοποιούμε κατ' αρχάς το πρόγραμμα για να μετατρέψουμε τον αριθμό 328.625 σε δυαδική έκφραση. Ο αριθμός αυτός είναι 101001000.101 ο οποίος σε έκφραση αριθμού κινητής υποδιαστολής είναι

$x=(0.101001000101)\ 10^{1001}$

Στον πρώτο χαρακτήρα θα αποθηκευθεί ο εκθέτης 1001 και στους υπόλοιπους τρεις η mantissa 101001000. Δηλαδή

Εκθέτης	mantissa
00001001	00100100000000000000000

Το πρώτο στοιχείο της mantissa ενώ είναι 1 βάζουμε το 0 γιατί ο αριθμός είναι θετικός.

Όπως είπαμε για τη mantissa χρησιμοποιούμε τρεις χαρακτήρες. Κάθε χαρακτήρας μπορεί να αποθηκεύσει έναν ακέραιο αριθμό από 0 έως 255. Αν θεωρήσουμε ότι κάθε χαρακτήρας είναι ένα ψηφίο του συστήματος αρίθμησης με βάση το 256, τότε η mantissa μπορεί να γραφεί σαν αριθμός του συστήματος με βάση το 256. Αν α_1, α_2 και α_3 είναι τα ψηφία που δηλώνουν οι τρεις χαρακτήρες, τότε η mantissa α* γράφεται

$$\alpha^*=0.\alpha_1\alpha_2\alpha_3=\alpha_1\,256^{-1}+\alpha_2\,256^{-2}+\alpha_3\,256^{-3}$$

Αν τώρα χρησιμοποιούμε τη στρογγύλευση σαν είδος αποκοπής, τότε το σφάλμα της στρογγύλευσης θα είναι το πολύ $0.5\ 256^{-3}$ δηλαδή $1.7\ 10^{-8}$ και ο υπολογιστής μας θα γράφει 7 σημαντικά δεκαδικά ψηφία προκειμένου να παραστήσει κάποιο αριθμό.

Με πιο απλά λόγια θα μπορούσαμε να πούμε πως το πλήθος των δυαδικών θέσεων της mantissa είναι 24. Δηλαδή οι θέσεις 25 και πάνω αγνοούνται. Συνεπώς το λάθος στρογγύλευσης θα είναι $0.5\ 2^{-25} = 1.7\ 10^{-8}$

1.7 Ασκήσεις

1. Για την αποθήκευση ακεραίων αριθμών έστω ότι διατίθενται 4 χαρακτήρες. Να βρεθεί το διάστημα των ακεραίων αριθμών που μπορεί να αποθηκευθούν χωρίς να έχουμε σφάλμα overflow.
2. Έστω ότι για την αποθήκευση ενός αριθμού κινητής υποδιαστολής διατίθενται 5 χαρακτήρες εκ των οποίων ο ένας χρησιμοποιείται για τον εκθέτη. Να βρεθεί ποιό είναι το σφάλμα στρογγύλευσης και πόσο είναι το μήκος του αριθμού κινητής υποδιαστολής στο δεκαδικό σύστημα.
3. Για το πρόσημο του αριθμού κινητής υποδιαστολής διατίθεται μια δυαδική θέση και μάλιστα η πρώτη από αυτές που καταλαμβάνει η mantissa. Να δοθεί μια σχηματική παράσταση για την αποθήκευση των αριθμών 245.6125 και -32.3456. Για την περίπτωση του αρνητικού αριθμού να αποθηκευθεί το συμπλήρωμά του.
4. Να γραφεί ένα πρόγραμμα που να μετατρέπει πραγματικούς αριθμούς σε αριθμούς κινητής υποδιαστολής μήκους n=7 στο δεκαδικό σύστημα. Για την αποκοπή των ψηφίων να εφαρμοσθεί η στρογγύλευση.
5. Να γραφεί ένα πρόγραμμα το οποίο να μετατρέπει δυαδικούς αριθμούς σε αντίστοιχους δεκαδικούς.
6. Να μετατραπεί ο ατέρμων δυαδικός κλασματικός αριθμός $x=(0.101010101010101\ldots)_2$ σε αντίστοιχο δεκαδικό. (Υπόδειξη: Μετατρέποντας τον x σε δυνάμεις του 2 θα έχουμε x=2-1+2-3+2-5 +2-7+….. που είναι το άθροισμα απείρων όρων φθίνουσας γεωμετρικής προόδου)