

Υπολογιστικές Μέθοδοι Στη Φυσική

ΦΥΣ 145

Άνοιξη 2013

Διδάσκων:

Φώτης Πτωχός

e-mail: fotis@ucy.ac.cy

Τηλ: 22.89.2837

Γραφείο: B235 – ΘΕΕ02

Γενικές Πληροφορίες

▪ Ώρες/Αίθουσα διδασκαλίας:

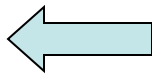
- Δευτέρα 10:30 – 12:00
- Αίθουσα: 229 ΦΥΣΙΚΟΥ

Διαλέξεις

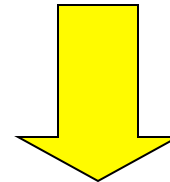


Εργαστήρια

- Δευτέρα 15:00 – 18:00
- Αίθουσα: B103



ΥΠΟΧΡΕΩΤΙΚΗ ΠΑΡΑΚΟΛΟΥΘΗΣΗ



Πάνω από μια απουσία στα εργαστήρια χωρίς σημαντικό λόγο ισοδυναμεί με αυτόματη αποτυχία στο μάθημα

Απορίες

- Διακόπτεται για απορίες κατά την διάρκεια των διαλέξεων.
- **Περάστε από το γραφείο μου:** Τρίτη είναι η καλύτερη μέρα αν και μπορείτε να έρθετε οποιαδήποτε ώρα...
- Πείτε μου αν κάτι στο μάθημα δεν δουλεύει για σας και πως μπορεί να αλλάξει
- Δύο μεταπτυχιακός συνάδελφοί σας θα είναι παρόντες κατά τη διάρκεια των εργαστηρίων και θα βοηθήσουν σε οποιαδήποτε προβλήματα

Βιβλιογραφία

Δεν θα ακολουθήσω κάποιο συγκεκριμένο βιβλίο.

Οι διαλέξεις/σημειώσεις θα είναι στο web.

Οι ασκήσεις/οδηγίες των εργαστηρίων όπως και οι **εργασίες** που θα δουλεύετε σπίτι θα τις βρίσκετε επίσης στο web.

(<http://www2.ucy.ac.cy/~phy145/phy145.htm>)

Χρήσιμη βιβλιογραφία/παραδείγματα μπορούν να βρεθούν στο web.

Προσπαθήστε και μόνοι σας τις πρώτες δύο βδομάδες να ασχοληθείτε και να εξοικειωθείτε με

Λειτουργικό σύστημα: **Linux**

Γλώσσα προγραμματισμού: **Fortran 77/90**

Λογισμικό Γραφικών αναπαραστάσεων: **gnuplot**

Όλοι έχετε accounts στο τμήμα της Φυσικής και μπορείτε να χρησιμοποιείται τους υπολογιστές του τμήματος για πρακτική εξάσκηση πέρα από τις ώρες εργαστηρίου.

Βιβλιογραφία

- Για Fortran θα μπορούσατε να χρησιμοποιήσετε:
 - **Fortran** από Σ. Περσίδη
 - **Fortran 90 explained** από M. Metcalf
 - [Μαθήματα Fortran](#)
- Για Linux θα μπορούσατε να χρησιμοποιήσετε τα ακόλουθα:
 - [Linux in a nutshell](#) από Ellen Siever, Stephen Spainhour, Stephen Figgins and Jessica P. Hekman , ed. O'Reilly
 - [Running Linux](#) από Matt Welsh, Matthias Kalle Dalheimer, and Lar Kaufman, ed. O'Reilly
- Για το λογισμικό παραστάσεων GNUPLOT:
 - Από την επίσημη ιστοσελίδα του [gnuplot](#)
- Πολλά μπορούν να βρεθούν επίσης χρησιμοποιώντας την μηχανή αναζήτησης στο web [google](#)

Βιβλιογραφία

- Για το κύριο μέρος του μαθήματος: **Υπολογιστική φυσική**
 - **Numerical Recipes** από William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery. Μπορείτε να το βρείτε επίσης **δωρεάν** με κώδικα σε Fortran77, Fortran 90 και C.
 - **Numerical Methods for Physics** (2nd edition) από A. Garcia (εκδ. Prentice Hall)
 - **A First Course in Computational Physics** από P. DeVries εκδ. Wiley & Sons
 - **An Introduction to Computer Simulation Methods** H. Gould and J. Tobochnik, Addison-Wesley, 2nd edition
 - **Computational Physics**, από N. Giordano, Prentice Hall

Βαθμολογία

- Η βαθμολογία θα βασιστεί στα ακόλουθα:
 - **10%** ασκήσεις εργαστηρίων
 - **10%** quiz στο εργαστήριο (1 κάθε 2 βδομάδες)
 - **15%** εργασίες στο σπίτι
 - **30%** 1 εξέταση προόδου (Σάββατο 9 Μάρτη, 10:00–16:00)
 - **35%** τελική εξέταση (Μάη)
- Αλλά:** Αν ο βαθμός της τελικής εξέτασης είναι καλύτερος από το βαθμό της προόδου, η τελική εξέταση μετρά 65%

Εργασίες στο σπίτι

- Εργασίες για το σπίτι θα περιέχουν 4–5 ασκήσεις
- Θα πρέπει να είναι δικιές σας και όχι αντιγραφή λύσεων άλλων συναδέλφων σας. Μπορείτε να συνεργάζεστε αλλά όχι ακριβώς ίδιες

Περιεχόμενο του Μαθήματος

- Βασικές αρχές προγραμματισμού σε Fortran, βασικές εντολές λειτουργικού συστήματος, εισαγωγή και βασικές λειτουργίες του [Editor/Emacs](#). Λογισμικό πακέτο γραφικών παραστάσεων δεδομένων.
- Επίλυση απλών διαφορικών εξισώσεων – Ραδιενεργή διάσπαση
- Μονοδιάστατη και δυσδιάστατη κίνηση σωμάτων σε ομογενές βαρυτικό πεδίο. Εκκρεμές. Ηλιακό σύστημα και πλανητικές τροχιές. Νόμοι του Kepler.
- Μή γραμμικά συστήματα
- Αριθμητική ολοκλήρωση
- Γεννήτορες τυχαίων αριθμών
- Ανάλυση πειραματικών δεδομένων
- Ολοκλήρωση Monte Carlo και προσομοίωση

Βασικές εντολές FORTRAN

❑ FORTRAN (**FOR**mula **TRAN**slator):

- Ξεκίνησε σαν γλώσσα προγραμματισμού στις αρχές της δεκαετίας του 50. Το 1966 απέκτησε την πρώτη τυπική της μορφή (Fortran IV). Αναθεωρήσεις της οδήγησαν στην Fortran 77 (πιο διαδεδομένη μορφή της). Αργότερα με την ανάπτυξη νέων γλωσσών προγραμματισμού (C, C++) η γλώσσα αναπτύχθηκε περισσότερο στην μορφή της Fortran 90 που ωστόσο δεν χρησιμοποιείται ευρέως.
- Η Fortran σχεδιάστηκε για επιστήμονες και μηχανικούς. Τα 30 τελευταία χρόνια χρησιμοποιήθηκε για διάφορα προγράμματα, από σχεδιασμό τμημάτων γεφυρών και αεροπλάνων, ανάλυση επιστημονικών δεδομένων, αυτοματοποίηση σε εργοστάσια κλπ
- Τα χρόνια της παρουσίας της διάφοροι χρήστες έγραψαν βιβλιοθήκες από χρήσιμα προγράμματα που χρησιμοποιούνται από χιλιάδες άλλους χρήστες κατά τρόπο παρόμοιο με αυτό του δανεισμού βιβλίων από βιβλιοθήκες.

Πρόγραμμα

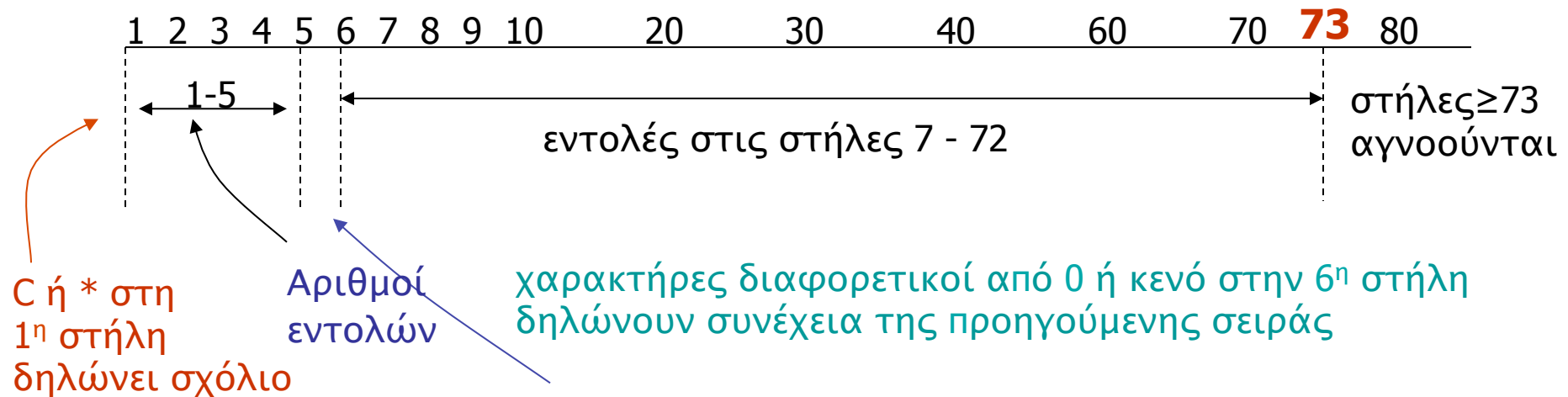
- ❑ Πρόγραμμα είναι μια σειρά από οδηγίες (εντολές προγράμματος) συνταγμένες με τέτοιο τρόπο ώστε να επιτρέψει έναν ηλεκτρονικό υπολογιστή να λύσει κάποιο πρόβλημα. Το πρόβλημα προς λύση είναι σπασμένο σε πολύ μικρότερα κομμάτια. Τα κομμάτια αυτά θα πρέπει να σχηματίζουν μια καλώς ορισμένη δομή, όπου το πιο περίπλοκο και μεγάλο πρόβλημα στην κορυφή και το πιο απλό και εύκολα επιλύσιμο στη βάση.
 - Από κει και ο ορισμός: top down programming: προγραμματισμός από πάνω προς τα κάτω
- ❑ Για να πετύχετε τη λύση ενός προβλήματος με την χρήση μιας γλώσσας προγραμματισμού καλό είναι να ακολουθήσετε τα παρακάτω βήματα:
 - ❑ Εκφράστε το πρόβλημα στα Αγγλικά
 - ❑ Εξετάστε το πρόβλημα και σπάστε το σε μικρότερα προβλήματα
 - ❑ Εξετάστε τα επιμέρους τμήματα και επεξεργαστείτε τα σε μικρότερα
 - ❑ Σχεδιάστε γραφικά το σχέδιο δράσης
 - ❑ Γράψτε το κύριο πρόγραμμα με αναφορές στα υποπρογράμματα
 - ❑ Δοκιμάστε το πρόγραμμα

Σχέση γλώσσας προγραμματισμού, προγράμματος, Η/Υ

Προγραμματιστής Chef	Γλώσσα Ελληνικά	Πρόγραμμα Κρέπες	Υπολογιστής Μάγειρας, σκεύη
Beethoven	Μουσική	Πιάνο Σονάτα	Πιανίστας, πιάνο
Εσείς	FORTRAN	ολοκλήρωμα	Ο υπολογιστής σας

- Δηλαδή προγραμματισμός είναι η διεργασία του γραψίματος κάποιων εντολών που μπορούν να εκτελεστούν πολλές φορές στο μέλλον
- Προσέξτε ότι και στα 3 παραδείγματα το πρόγραμμα είναι το μέσο για την μεταφορά εντολών ή οδηγιών μεταξύ του προγραμματιστή (αυτού που ανακαλύπτει τις οδηγίες) και του υπολογιστή (αυτού που εκτελεί τις οδηγίες)
- Η γλώσσα θέτει τους όρους κάτω από τους οποίους θα σχηματιστεί το πρόγραμμα. Για να γίνει κατανοητό ένα πρόγραμμα από τον υπολογιστή θα πρέπει να γραφτεί σύμφωνα με τους όρους αυτούς της γλώσσας.
- Το πρώτο στάδιο της διεργασίας μάθησης του προγραμματισμού είναι να μάθετε πρώτα τι μπορεί να κάνει ο υπολογιστής και πως μπορείτε να γράψετε τις οδηγίες ώστε να κάνει αυτά τα πράγματα. Το υπόλοιπο της διεργασίας είναι να μάθετε να γράφετε ωραία προγράμματα.

Σύνταξη του Προγράμματος



- Οι θέσεις σε μια γραμμή ονομάζονται στήλες και αριθμούνται 1,2,3...80.
- Όλες οι εντολές FORTRAN πρέπει να τοποθετούνται στις στήλες 7-72
- Οτιδήποτε εμφανίζεται πέρα από τη στήλη 73 αγνοείται ή δίνει λάθος.
- Αν μια εντολή χρειάζεται να αποκτήσει ετικέτα για προγενέστερη ή μεταγενέστερη αναφορά η ετικέτα πρέπει να δοθεί στις στήλες 1-5 και η ετικέτα πρέπει να ναι κάποιος ακέραιος απο 1 – 99999
- Μερικές φορές δεν μπορούμε να συμπληρώσουμε μια εντολή σε μια σειρά. Τότε για να συνεχίσουμε στην επόμενη θα πρέπει να βάλουμε ένα χαρακτήρα διαφορετικό από * ή κενό στη στήλη 6 και να συνεχίσουμε την εντολή
- Γραμμές που ξεκινούν με το γράμμα (C) ή (*) στην 1^η στήλη αποτελούν σχόλια και δεν εκτελούνται

Γενική μορφή ενός προγράμματος Fortran

επικεφαλίδα

Αυτή η εντολή δηλώνει την αρχή και το όνομα ενός προγράμματος

εισαγωγικά σχόλια

Δίνονται για να δηλώσουν το σκοπό του προγράμματος και να δώσουν πληροφορίες για εισαγωγή/εξαγωγή πληροφορίας

τμήμα ορισμών

Το τμήμα αυτό χρησιμοποιείται για τον ορισμό διαφόρων μεταβλητών που χρησιμοποιούνται για την αποθήκευση σταθερών ή ενδιάμεσων αποτελεσμάτων του προγράμματος

τμήμα εκτέλεσης

Το τμήμα αυτό περιέχει τις εντολές που εκτελούν τμήματα του αλγόριθμου του προγράμματος.

Παράδειγμα προγράμματος

Υπολογισμός ύψους και ταχύτητας ενός σώματος που εκτελεί κατακόρυφη βολή σε μια χρονική στιγμή t : $h = h_0 + v_0t + 0.5at^2$ $v = v_0 + at$

PROGRAM PROJEC

```
*****
* This program calculates the velocity and height of a projectile *
* given its initial height, initial velocity, and constant      *
* acceleration. Variables used are:                             *
*   HGHT0 : initial height                                       *
*   HGHT  : height at any time                                   *
*   VELOC0 : initial vertical velocity                           *
*   VELOC  : vertical velocity at any time                       *
*   ACCEL  : vertical acceleration                               *
*   TIME   : time elapsed since projectile was launched         *
*                                                                 *
* Input:  none                                                  *
* Output: VELOC, HGHT                                           *
*****
```

```
REAL HGHT0, HGHT, VELOC0, VELOC, ACCEL, TIME
```

```
ACCEL = -9.807
HGHT0 = 100.0
VELOC0 = 90.0
TIME = 4.3
```

```
HGHT = 0.5 * ACCEL * TIME ** 2 + VELOC0 * TIME + HGHT0
VELOC = ACCEL * TIME + VELOC0
PRINT *, 'AT TIME ', TIME, ' THE VERTICAL VELOCITY IS ', VELOC
PRINT *, 'AND THE HEIGHT IS ', HGHT
```

END

επικεφαλίδα - απαραίτητο

Σχόλια
ξεκινούν
με * ή C
στη 1^η στήλη

ορισμοί

τμήμα εκτέλεσης

τερματισμός - απαραίτητο

Προσοχή ότι
οι εντολές
ξεκινούν
από τη
στήλη 7

Κύρια συστατικά ενός προγράμματος

Σταθερές

Μεταβλητές

Αριθμητικές πράξεις

Ανάθεση τιμών – ισότητες

Έλεγχος αποτελέσματος – επικοινωνία με το πρόγραμμα

Σταθερές (**constants**)

- ❑ Είναι οι ποσότητες το μέγεθος των οποίων δεν αλλάζει κατά την εκτέλεση ενός προγράμματος.
- ❑ Μπορεί να είναι αριθμητικές ή να περιέχουν χαρακτήρες
- ❑ Οι συνηθέστερες μορφές σταθερών είναι:
 - **INTEGER** (ακέραια σταθερά)
 - **REAL** (πραγματική σταθερά)
 - **DOUBLE PRECISION** (σταθερά διπλής ακρίβειας)
 - **CHARACTER** (σταθερά χαρακτήρων)
 - **LOGICAL** (λογική σταθερά) (αργότερα)

Σταθερές

❑ INTEGER (ακέραια σταθερά)

- Αποτελείται από ψηφία 0 έως 9 και πιθανώς ένα + ή – που δηλώνει αν ο αριθμός είναι θετικός ή αρνητικός
- Αν ο αριθμός είναι θετικός τότε το + δεν χρειάζεται
- Αν ο αριθμός είναι αρνητικός τότε το – πρέπει υποχρεωτικά να γραφεί μπροστά από τα ψηφία

Παραδείγματα ακεραίων σταθερών είναι:

3 -7 1803 +68 0 1000 -0 +0 -999



Υπάρχουν όρια μεταξύ των οποίων πρέπει να βρίσκεται κάθε χρησιμοποιούμενος ακέραιος. Τα όρια εξαρτώνται από τον Η/Υ.

Ανάλογα με τον Η/Υ, ο μεγαλύτερος ακέραιος που μπορεί να αναπαρασταθεί είναι: **2147483648** (32 bits μηχανή)

Αν κάποιος ακέραιος πάρει μεγαλύτερη τιμή, τότε έχουμε **υπερχείλιση (overflow)** και το πρόγραμμα θα δώσει ανοησίες




Σταθερές

□ REAL (πραγματικές σταθερές)

- Ένας πραγματικός αποτελείται από ψηφία (0 έως και 9),
μια τελεία (υποχρεωτικά) και πιθανώς το πρόσημο + ή -
- Αν ο αριθμός είναι θετικός το + μπορεί να παραληφθεί
- Αν ο αριθμός είναι αρνητικός τότε πρέπει οπωσδήποτε να γράψουμε το - μπροστά από τα ψηφία του.

π.χ. 2.3 -5.75 .302 -7. 0. +23. 0.0 -0.9999

❖ Μια σταθερά REAL γράφεται και με τη μορφή: $r \times E i$  $r \times 10^i$

π.χ. 2.3 γράφεται και σαν 2.3E00 ή 23.E-1 ή 2300.E-3

-.00854 γράφεται και σαν -8.54E-3 ή -854.E-5

10^{-12} γράφεται και σαν 1.E-12 ή .1E-11



Οι πραγματικοί αριθμοί έχουν ένα συγκεκριμένο εύρος τιμών που μπορούν να πάρουν, ανάλογα με τον Η/Υ

Η απόλυτη τιμή του πραγματικού δεν μπορεί να είναι μικρότερη ενός αριθμού **αλλά μπορεί να είναι 0.**



Σταθερές

□ DOUBLE PRECISION (σταθερά διπλής ακρίβειας)

- Μοιάζει με μια REAL σταθερά **εκτός από το γεγονός** ότι γράφεται και διατηρείται στη μνήμη του Η/Υ με περισσότερα (συνήθως υπερδιπλάσια) ψηφία
- Η γραφή της είναι ίδια με τη γραφή μιας REAL σταθεράς σε εκθετική μορφή **αλλά** χρησιμοποιούμε το γράμμα D αντί για E

π.χ. 7.50D2 -0.06D-70 37.342423D65 1.54D178

7.5×10^2 -6×10^{-72} 37342423×10^{59} 154×10^{176}

- Δηλαδή αποτελείται από ψηφία, μια υποδιαστολή, ενώ ο αριθμός που βρίσκεται στα δεξιά του D αποτελεί τον εκθέτη
- Το εύρος τιμών μιας σταθεράς διπλής ακρίβειας εξαρτάται από τον Η/Υ
Είναι πολύ μεγαλύτερο από ότι για τις σταθερές τύπου REAL



Σταθερές

□ Character (σταθερές χαρακτήρων)

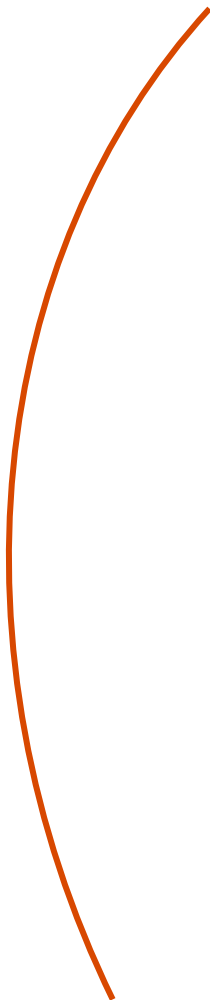
- Οι σταθερές αυτές αποτελούνται από μια γραμματοσειρά η οποία πρέπει πάντοτε να εσωκλείεται μεταξύ αποστρόφων

‘CYPRUS’, ‘’, ‘x*y’

Σωστός τρόπος γραφής

“CYPRUS”, “

Λάθος τρόπος γραφής



Άνοιγμα παρένθεσης.....

bits and bytes

- ❑ Ο υπολογιστής χρησιμοποιεί τη **κύρια μνήμη** για αποθήκευση δεδομένων
- ❑ Η μνήμη χωρίζεται σε **words** και κάθε word περιέχει τμήμα πληροφορίας
- ❑ Ο αριθμός των words σε μια κεντρική μνήμη εξαρτάται από τον Η/Υ (αρκετές χιλιάδες για ένα μικρο-processor σε εκατοντάδες εκατομμύρια για μεγάλους υπολογιστές)
- ❑ Κάθε word αποτελείται από μικρότερες μονάδες που ονομάζονται **bits**

Το bit μπορεί να έχει μόνο μια από δύο τιμές: **0 ή 1**

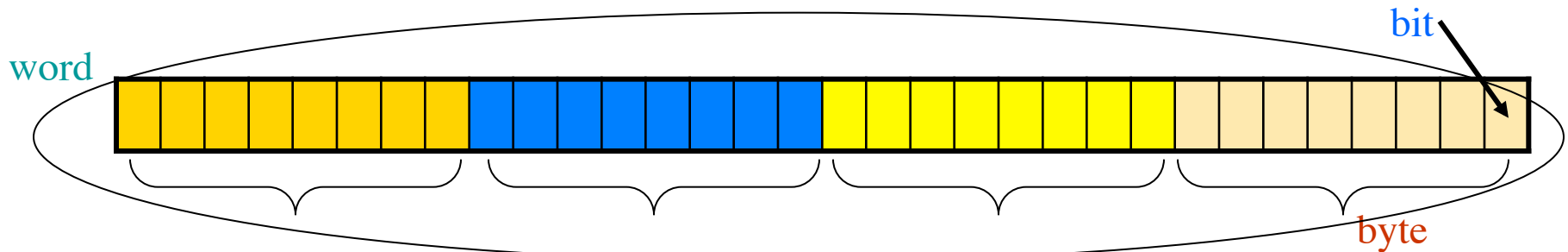
Αν το word περιέχει m bits τότε το word μπορεί να είναι σε οποιαδήποτε από 2^m διαφορετικές καταστάσεις

Η πραγματική σημασία που δίνεται σε κάποιο ιδιαίτερο συνδυασμό από 0 και 1 σε ένα word εξαρτάται από το μοντέλο του υπολογιστή

- ❑ Όλοι οι υπολογιστές (**σχεδόν**) αποθηκεύουν τους θετικούς INTEGERS με τέτοιο τρόπο ώστε κάθε bit που παίρνει τη τιμή 1 στο word αντιπροσωπεύει μια τιμή που είναι $2^{(n-1)}$ όπου n η σχετική θέση του bit στο word μετρώντας από δεξιά.

bits and bytes

- ❑ Οι περισσότεροι υπολογιστές έχουν μνήμη με 32 bits/word με ενδιάμεσο διαχωρισμό του word σε 4 ομάδες από 8 συνεχή bits
- ❑ Ένα τέτοιο γκρουπ αποτελούμενο από 8 bits ονομάζεται **byte**
- ❑ Ένα byte έχει 2^8 ή 256 διαφορετικές καταστάσεις.
Όλοι οι αριθμοί αντιπροσωπεύονται σε δυαδική μορφή



- ❑ Υπάρχει περιορισμένη ακρίβεια και επομένως προσεγγίσεις
- ❑ Το μήκος του word είναι συνήθως 32 bits ή 4 bytes όπου κάθε byte αποτελείται από 8 bits

Μονάδες μέτρησης για την αποθήκευση αριθμών είναι:

$$1 \text{ byte} = 1\text{B} = 8 \text{ bits} = 8\text{b}$$

$$1 \text{ kbyte} = 2^{10} \text{ bytes} = 1024 \text{ bytes}$$

$$1 \text{ Mbyte} = 2^{10} \text{ kbytes} = 1024 \times 1024 \text{ bytes}$$



Κλείσιμο παρένθεσης.....

Γιατί υπάρχει περιορισμός στους αριθμούς

- ❑ Σχετίζεται με την αναπαράσταση αριθμών στον υπολογιστή
- ❑ Τρία συστήματα αριθμών:
 - Οι αριθμοί μέτρησης 0,1,2,...,32767
Συνδέεται με τον δείκτη των καταλόγων της μηχανής και άρα το εύρος των αριθμών είναι προσδιορισμένο.
 - Οι αριθμοί σταθερής υποδιαστολής
Οι αριθμοί αυτοί έχουν σταθερό μήκος = μήκος του word του Η/Υ ή κάποιο πολλαπλάσιο
Η διαφορά μεταξύ διαδοχικών αριθμών είναι η ίδια
Τέτοιοι είναι οι αριθμοί των υπολογισμών του χεριού
(Όλοι οι πίνακες είναι συνήθως σταθερής υποδιαστολής)
 - Οι αριθμοί κινητής υποδιαστολής (floating point numbers)
Σχετίζονται με τη λεγόμενη επιστημονική σήμανση
Το σύστημα αυτό είναι σχεδιασμένο για να περιγράφει και μικρούς αλλά και μεγάλους αριθμούς


Αναπαράσταση αριθμών στον Η/Υ

- ❑ Οι αριθμοί κινητής υποδιαστολής (floating point numbers)

$$.31415927 \times 10^1$$

$$.12345678 \times 10^{-1}$$

$$-.12345678 \times 10^4$$

**Mantissa:** Το πρώτο τμήμα των 8 ψηφίων στους παραπάνω αριθμούς
Εκθέτης: Το τελευταίο ψηφίο (με εύρος τιμών [-38, +38])

Η mantissa και ο εκθέτης αποθηκεύονται στο ίδιο word της μηχανής

Σαν αποτέλεσμα η mantissa ενός αριθμού κινητής υποδιαστολής είναι μικρότερου μήκους του αντίστοιχου αριθμού σταθερής υποδιαστολής

Απεικόνιση αριθμών στους υπολογιστές

- Ένας ακέραιος αριθμός 2^N αντιπροσωπεύεται από N bits

Το 1^ο bit δίνει το πρόσημο

Τα υπόλοιπα $N-1$ bits χρησιμοποιούνται για τον αριθμό

Άρα 32 bits INTEGERS θα πέρνουν τιμές στο διάστημα:

$$-2147483648 < \text{Integer}_{32} < 2147483648 = 2^{31}$$

- Η αριθμητική με Integer αριθμούς είναι ακριβής εκτός από **υπερχείλιση** (overflow) και **υποχείλιση** (underflow)

- Οι αριθμοί **απλής ακρίβειας** (single precision ή floating - point F) χρειάζονται 4 bytes (32 bits) για την αναπαράστασή τους

Τα οποία χρησιμοποιούνται ως εξής:

{

23 bits για mantissa

8 bits για εκθέτη

1 bit για το πρόσημο

Η μικρότερη mantissa είναι $2^{-23} \sim 10^{-7}$

Εκθέτης:

$2^{\alpha - 127}$ όπου $\alpha \in [0, 255]$ (2^8 bits)

↖ "bias"

Απεικόνιση αριθμών

□ Για αριθμούς **διπλής ακρίβειας** χρειάζονται 8 bytes δηλαδή 64 bits

και χρησιμοποιούνται ως εξής:

- 52 bits για την mantissa
- 1 bit για το πρόσημο
- 11 bits για τον εκθέτη (**bias=1023**)

Η μικρότερη mantissa είναι $2^{-52} \sim 10^{-16}$

Ο εκθέτης είναι $2^{\alpha-1023}$ με $\alpha \in [0, 2048]$ (2^{11} bits)

Διάστημα

$$2.9 \times 10^{-39} \leq \text{float}_{32} \leq 3.4 \times 10^{38}$$

$$10^{-322} \leq \text{double}_{64} \leq 10^{308}$$

Ακρίβεια

float_{32} : 5-7 δεκαδικά ψηφία

double_{64} : 16 δεκαδικά ψηφία

Αριθμοί κινητής υποδιαστολής

□ Έχουν μια σειρά από ιδιαιτερότητες:

- Ας υποθέσουμε για απλότητα ότι έχουμε μια mantissa με 3 ψηφία και ο εκθέτης αποτελείται από 1 ψηφίο:

Το μηδέν, $0 = .000 \times 10^{-9}$ είναι απομονωμένο από τους υπόλοιπους αριθμούς

αφού οι αμέσως επόμενοι θετικοί αριθμοί είναι $.100 \times 10^{-9}$ και $.101 \times 10^{-9}$ βρίσκονται 10^{-12} μακριά!!

Δεν υπάρχει κανένας άλλος αριθμός με mantissa να ξεκινά από 0

- Κάθε δεκάδα έχει ακριβώς τον ίδιο πλήθος αριθμών: **συνολικά 900** πηγαίνοντας από $.100 \times 10^a$ μέχρι $.999 \times 10^a$ με $a = -9, -8, \dots, 0, \dots, 8, 9$

0.999x10 ⁻¹ 0.100x10 ⁰	1x10 ⁻⁴
0.999x10 ⁰ 0.100x10 ¹	1x10 ⁻³
0.999x10 ¹ 0.100x10 ²	1x10 ⁻²

- Σε κάθε δεκάδα οι 900 αριθμοί είναι χωρισμένοι σε ίσα διαστήματα

αλλά η διαφορά μεταξύ 2 δεκάδων είναι άνιση

Υπάρχει γεωμετρικό πήδημα

Αριθμοί κινητής υποδιαστολής

- ❑ Ο αριθμός 0 και -0 (δηλαδή -0.000×10^{-9}) είναι λογικά ίδιοι
Οι περισσότεροι υπολογιστές τους έχουν ίδιους αλλά μερικοί όχι
 - Συνήθως δεν υπάρχει άπειρο που να αντιστοιχεί στο μηδέν
- ❑ Στα μαθηματικά υπάρχει ένα και μοναδικό μηδέν
- ❑ Στο προγραμματισμό υπάρχουν 2 είδη:
 - Αυτό που εμφανίζεται σε εκφράσεις όπως $\alpha \cdot 0 = 0$ που συμπεριφέρεται σαν κανονικό 0
 - Αλλά και το 0 που εμφανίζεται σε σχέσεις της μορφής $\alpha - \alpha = 0$
Αυτό το 0 μπορεί να προέρχεται από $1 - (1 - \varepsilon)(1 + \varepsilon) = 0$ όταν το $\varepsilon < \frac{1}{2} 10^{-3}$
όπου το 3 είναι ο αριθμός των δεκαδικών ψηφίων της mantissa
 - Αυτό το τελευταίο 0 εμφανίζεται κατά την αφαίρεση 2 σχεδόν ίσου μεγέθους αριθμών
 - Αποτελεί την πηγή πολλών σφαλμάτων στους υπολογισμούς με ένα ηλεκτρονικό υπολογιστή