ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στον Προγραμματισμό Η/Υ για Χημικούς Μηχανικούς

Παρουσίαση Διαλέξεων: 1. Αλγόριθμοι

Καθηγητής Δημήτρης Ματαράς

ΑΛΓΟΡΙΘΜΟΙ

να
για
στην
καταναλοί
Πραγματικά
διδάξει
είναι
δεν
είτε
κάτι
κάποιος
Που
το
Μόνο
αφού
γλωσσολογία
κατανόηση
διανοητικό
κατασκευάζει
αυτή γνώση
σίγουρα
Μουσική άλλο
έχει Περισσότερα
υΠολογιστών ότι κώδικα
Πολύ
αναλύει
κανείς
εκΠαιδεύεται
ένα των
άλλων
γενικής
χημεία
χρησιμεύει
Πρόκειται
αντικειμένων
στον
ανθρωΠος
αΠό
γνωρίζει εΠιστήμη
κάΠοιον χρησιμοΠοιεί
γράφει
ΠραγΜατικότητα
υΠολογιστή

Αλγόριθμοι

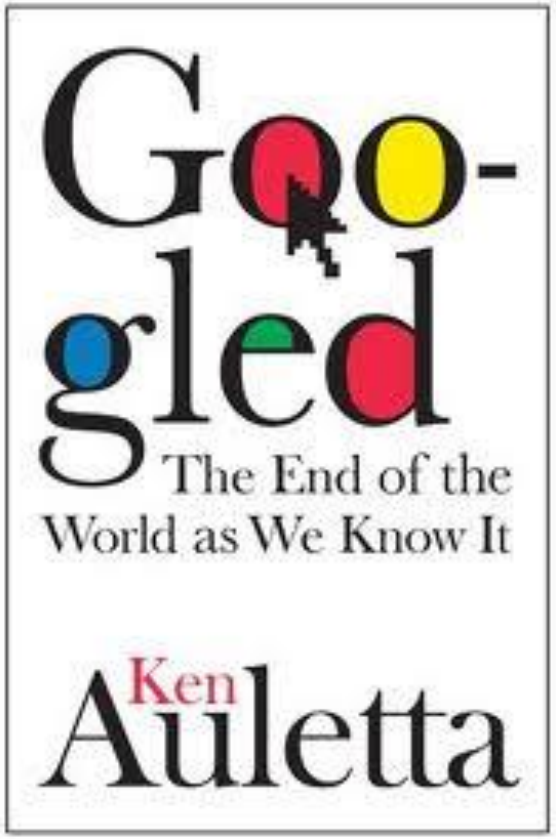
- «Η αλγοριθμική δεν είναι ένας κλάδος της επιστήμης των υπολογιστών. Είναι ο πυρήνας της επιστήμης των υπολογιστών και μπορεί να ειπωθεί δικαιωματικά ότι αφορά σχεδόν όλη την επιστήμη, τις επιχειρήσεις και την τεχνολογία» *D. Harel "Algorithmics: The Spirit of Computing"*
- Οι αλγόριθμοι μπορούν να ειδωθούν σαν **ειδικές λύσεις σε προβλήματα**. Δεν είναι απαντήσεις αλλά ακριβείς διαδικασίες για να φτάσουμε στις απαντήσεις.
- Οι **τεχνικές σχεδιασμού των αλγορίθμων είναι στρατηγικές επίλυσης** προβλημάτων ανεξάρτητα από το αν χρησιμοποιείται υπολογιστής

- *Δυστυχώς δεν μπορούν όλα τα προβλήματα να λυθούν με τη βοήθεια αλγορίθμων. Π.χ. δεν υπάρχει αλγόριθμος που να εξηγεί πως μπορεί κανείς να ζήσει ευτυχισμένος ή πως μπορεί κανείς να γίνει πλούσιος ή διάσημος*

Κάποιοι όμως έγιναν πλούσιοι

Larry Page and Sergey Brin

Συνιδρυτές της google



Τι λέει ο Donald Knuth

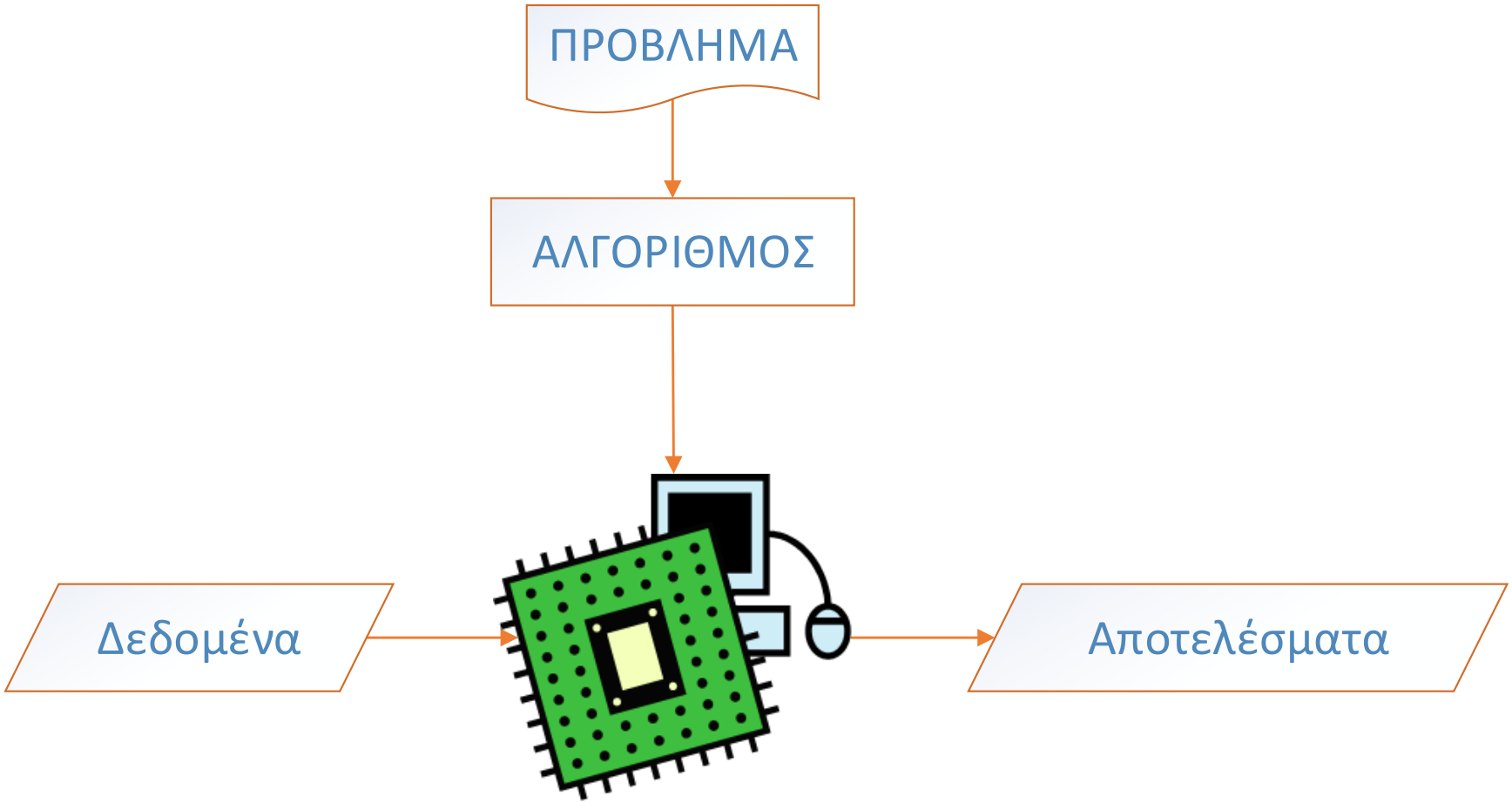
- «Ένας ανθρωπος που εκπαιδεύεται στην επιστήμη των υπολογιστών γνωρίζει να κατασκευάζει, να χρησιμοποιεί, να κατανοεί και να αναλύει αλγορίθμους.
- Αυτή η γνώση είναι προετοιμασία για πολύ περισσότερα από το να γράφει κανείς κώδικα· είναι ένα διανοητικό εργαλείο γενικής χρήσης που σίγουρα χρησιμεύει στην κατανόηση άλλων αντικειμένων είτε πρόκειται για χημεία είτε για γλωσσολογία ή μουσική.
- Έχει ειπωθεί ότι κάποιος δεν κατανοεί πραγματικά κάτι παρά μόνο αφού το διδάξει σε κάποιον άλλο. Στην πραγματικότητα κάποιος δεν κατανοεί πραγματικά κάτι παρα μόνο αφού το διδάξει στον υπολογιστή»

Κύριο έργο: “*The art of Computer Programming*”

Τι είναι ο αλγόριθμος

- **Επίλυση**: (ενός προβλήματος) συγκεκριμένη λύση (αποτέλεσμα), σε πεπερασμένο χρόνο, για οποιοδήποτε στιγμιότυπο του προβλήματος
- **Στιγμιότυπο**: σύνολο δεδομένων εισόδου τα οποία περιλαμβάνονται στο πεδίο ορισμού του προβλήματος
- **Αλγόριθμος**: διαδικασία αποτελούμενη από σαφείς (μη αμφίσημες) εντολές που καθοδηγούν στην επίλυση ενός προβλήματος

Τι είναι ο αλγόριθμος



Ο αλγόριθμος μπορεί να αναπαρασταθεί με διάφορους τρόπους 1

Αλγόριθμος του Ευκλείδη για τον υπολογισμό του $\text{ΜΚΔ}(m, n)$

ΒΗΜΑ 1: Αν το $n = 0$, το αποτέσμα είναι $\text{ΜΚΔ} = m$ STOP
Αλλιώς πήγαινε στο ΒΗΜΑ 2

ΒΗΜΑ 2: Διαίρεσε m/n και δώσε την τιμή του υπολοίπου στο r

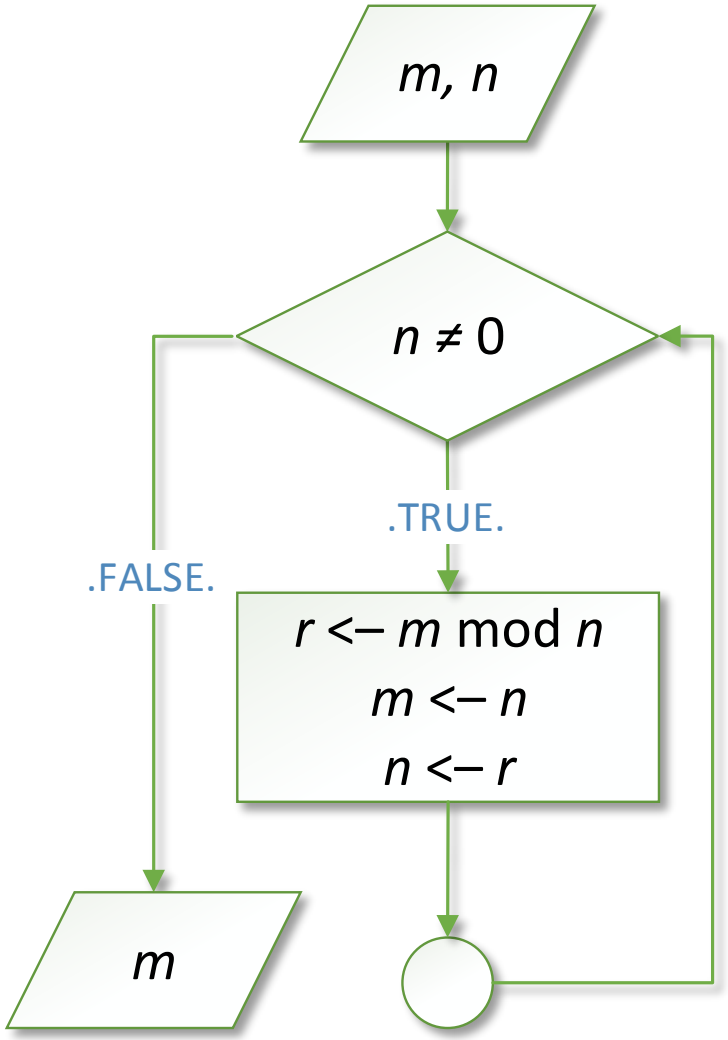
ΒΗΜΑ 3: Δώσε την τιμή του n στο m και του r στο n. Πήγαινε στο ΒΗΜΑ 1

ALGORITHM Euclid(m, n)

```
while n ≠ 0 do
    r ← m mod n
    m ← n
    n ← r
return m
```

```
ΠΡΟΓΡΑΜΜΑ Ευκλείδης
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: m, n, r
ΑΡΧΗ
  ΔΙΑΒΑΣΕ m, n
  ΟΣΟ n <> 0 ΕΠΑΝΑΛΑΒΕ
    r <- m mod n
    m <- n
    n <- r
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ΓΡΑΨΕ m
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Ο αλγόριθμος μπορεί να αναπαρασταθεί με διάφορους τρόπους




```
ΠΡΟΓΡΑΜΜΑ Ευκλείδης
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: m, n, r
ΑΡΧΗ
  ΔΙΑΒΑΣΕ m, n
  ΟΣΟ n <> 0 ΕΠΑΝΑΛΑΒΕ
    r <- m mod n
    m <- n
    n <- r
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ΓΡΑΨΕ m
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Υπάρχουν κι' άλλοι αλγόριθμοι 1

Έλεγχος Διαδοχικών Ακεραίων για τον υπολογισμό του $\text{ΜΚΔ}(m, n)$

ΒΗΜΑ 1: θεωρούμε ότι $t = \min(m, n)$

ΒΗΜΑ 2: Διαίρεσε m/t. Αν το υπόλοιπο είναι 0 πήγαινε στο ΒΗΜΑ 3,
Αλλιώς πήγαινε στο ΒΗΜΑ 4

ΒΗΜΑ 3: Διαίρεσε n/t. Αν το υπόλοιπο είναι 0 τότε $\text{ΜΚΔ} = t$ STOP
Αλλιώς πήγαινε στο ΒΗΜΑ 4

ΒΗΜΑ 3: Μείωσε την τιμή του t κατά 1. Πήγαινε στο ΒΗΜΑ 2

- Δεν δουλεύει καλά αν ένα από τα δεδομένα είναι 0

Κάθε αλγόριθμος έχει συγκεκριμένο πεδίο ορισμού

- Χρειάζεται ασφαλώς περισσότερες πράξεις/συγκρίσεις

Υπάρχουν κι' άλλοι αλγόριθμοι 2

Γινόμενο Πρώτων Παραγόντων για τον υπολογισμό του $\text{ΜΚΔ}(m, n)$

ΒΗΜΑ 1: Βρες τους πρώτους παράγοντες του m

ΒΗΜΑ 2: Βρες τους πρώτους παράγοντες του n

ΒΗΜΑ 3: Βρες όλους τους κοινούς πρώτους παράγοντες

ΒΗΜΑ 4: Υπολόγισε το γινόμενο p όλων των κοινών πρώτων παραγόντων.

$\text{ΜΚΔ} = p$ STOP

Π.χ $60 = 2 \cdot 2 \cdot 3 \cdot 5$

$24 = 2 \cdot 2 \cdot 2 \cdot 3$

$\text{ΜΚΔ}(60,24) = 2 \cdot 2 \cdot 3 = 12$

Ο αλγόριθμος δεν είναι σαφής: χρειαζόμαστε ένα επιπλέον αλγόριθμο για να βρίσκουμε τους πρώτους παράγοντες (και άλλον ένα για να βρίσκουμε τους κοινούς πρώτους παράγοντες)

- Χρειάζεται πολύ περισσότερες πράξεις/συγκρίσεις

Αλγόριθμος του Ερατοσθένη 1

ALGORITHM Heratosthenes(m, n)
    for $p \leftarrow 2$ **to** n **do**
        $A[p] \leftarrow p$
    for $p \leftarrow 2$ **to** $\sqrt{n}$ **do**
        if $A[p] \neq 0$
            $j \leftarrow p * p$
            while $j \leq n$ **do**
                $A[p] \leftarrow 0$
                $j \leftarrow j + p$
    $i \leftarrow 0$
    for $p \leftarrow 2$ **to** n **do**
        if $A[p] \neq 0$
            $B[i] \leftarrow A[p]$
    $i \leftarrow i + 1$
    return B

Αλγόριθμος του Ερατοσθένη 2

```
ΠΡΟΓΡΑΜΜΑ Ερατοσθένης

ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: n, i, j, p, Α[100], Β[30]

ΑΡΧΗ

  ΓΡΑΨΕ 'Δώσε ακέραιο από 2 έως 100'
  ΔΙΑΒΑΣΕ n

  ΓΙΑ p ΑΠΟ 2 ΜΕΧΡΙ n

    Α[p] <- p

  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

Γεμίζω τον πίνακα Α
με αριθμούς
από 2 μέχρι n

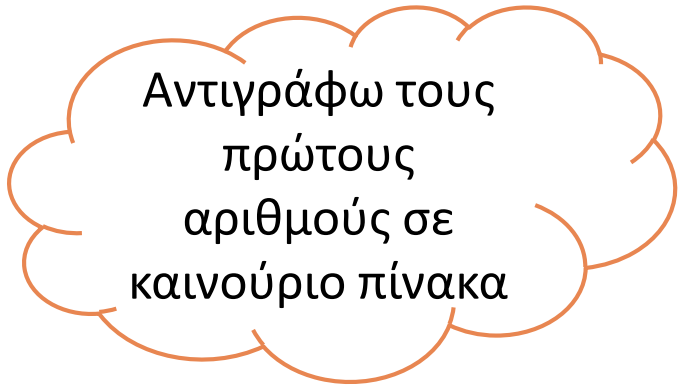
Αλγόριθμος του Ερατοσθένη 3

```
ΓΙΑ p ΑΠΟ 2 ΜΕΧΡΙ Τ_Ρ(n)
    ΑΝ Α[p] <> 0 ΤΟΤΕ
        j <- p * p
        ΟΣΟ j <= n ΕΠΑΝΑΛΑΒΕ
            Α[j] <- 0
            j <- j + p
        ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
    ΤΕΛΟΣ_ΑΝ
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

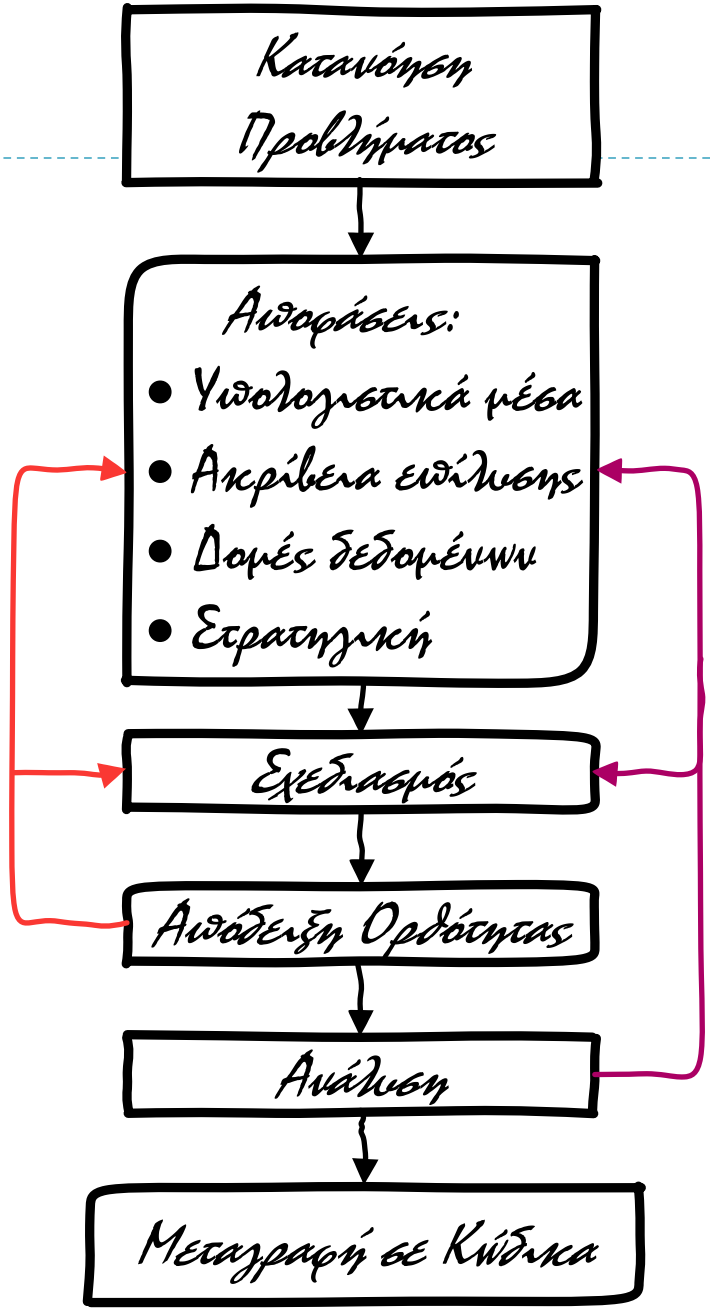
Μηδενίζω τα πολλαπλάσια του p

Αλγόριθμος του Ερατοσθένη 4

```
i <- 1
ΓΙΑ p ΑΠΟ 2 ΜΕΧΡΙ n

  ΑΝ A[p] <> 0 ΤΟΤΕ

    B[i] <- A[p]
    ΓΡΑΨΕ B[i]
    i <- i + 1

  ΤΕΛΟΣ_ΑΝ

ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```



Βασικά βήματα



Βασικά βήματα

- Κατανόηση προβλήματος
- Επιλογή υπολογιστικών μέσων
  - Σειριακοί Αλγόριθμοι
  - Παράλληλοι Αλγόριθμοι
- Ακρίβεια
  - Ακριβείς Αλγόριθμοι
  - Προσεγγιστικοί Αλγόριθμοι
- Δομές δεδομένων
- Τεχνικές Σχεδιασμού Αλγορίθμων
- Μέθοδοι Αναπαράστασης
  - Ψευδοκώδικας
  - Διάγραμμα ροής
- Απόδειξη Ορθότητας
- Ανάλυση
  - Χρονική Απόδοση
  - Χωρική Απόδοση
  - Απλότητα
  - Γενικότητα
- Κωδικοποίηση

Τυπικές Κατηγορίες Προβλημάτων

- Διάταξη (sorting)
- Αναζήτηση (searching)
- Διαχείριση Ακολουθίας Χαρακτήρων (string processing)
- Γραφήματα (graph problems)
- Συνδυαστικά προβλήματα (combinatorial problems)
- Γεωμετρικά προβλήματα (geometric problems)
- Αριθμητική ανάλυση (numerical alalysis)

Δομές Δεδομένων

- Γραμμικές Δομές δεδομένων
  - Πίνακες (arrays)
    - Στατικοί, Δυναμικοί
    - Ακολουθίες Χαρακτήρων (strings)
  - Συνδεδεμένες Λίστες (linked lists)
    - Απλά συνδεδεμένες, Διπλά συνδεδεμένες
  - Λίστες (Πίνακες ή συνδεδεμένες Λίστες)
    - Σωροί (stacks, heaps), Ουρές (queues)
- Γραφήματα
  - Graphs, Trees, Sets

Μερικές Τεχνικές Σχεδιασμού Αλγορίθμων

- Brute Force
  - Συστηματική εξέταση όλων των πιθανοτήτων (Σειριακή αναζήτηση)
- Divide and Conquer
  - Υποδιαιρούμε το πεδίο τιμών επαναληπτικά (Διχοτόμηση)
- Decrease and Conquer
  - Ανάγουμε το πρόβλημα επαναληπτικά σε ένα μικρότερο (Ευκλείδης)
- Transform and Conquer
  - Μετατρέπουμε τη μορφή των δεδομένων (π.χ. διάταξη και μετά διχοτομική αναζήτηση)
- Dynamic Programming
  - Ανάγουμε σε απλούστερα προβλήματα (π.χ. Fibonacci)
- Greedy Techniques
  - Σε κάθε βήμα παίρνουμε τη βέλτιστη απόφαση

Ανάλυση Απόδοσης Αλγορίθμων

- Μέγεθος δεδομένων
  - Η απόδοση είναι συνάρτηση του πλήθους των δεδομένων n.
- Μονάδες μέτρησης του χρόνου εκτέλεσης
  - $T(n) \approx c_{op}C(n)$
    - $T(n)$ ο χρόνος εκτέλεσης
    - c_{op} ο χρόνος εκτέλεσης μιας βασικής πράξης
    - $C(n)$ το πλήθος επαναλήψεων της βασικής πράξης

Ανάλυση Απόδοσης Αλγορίθμων

- Κλίμακες Ανάπτυξης (Orders of Growth)

n	$\log_2 n$	n	$n \log_2 n$	n^2	n^3	2^n	$n!$
10^1	3.3	10^1	$3.3 \cdot 10^1$	10^2	10^3	10^3	$3.6 \cdot 10^6$
10^2	6.6	10^2	$6.6 \cdot 10^2$	10^4	10^6	$1.3 \cdot 10^{30}$	$9.3 \cdot 10^{157}$
10^3	10	10^3	$1.0 \cdot 10^4$	10^6	10^9		
10^4	13	10^4	$1.3 \cdot 10^5$	10^8	10^{12}		
10^5	17	10^5	$1.7 \cdot 10^6$	10^{10}	10^{15}		
10^6	20	10^6	$2.0 \cdot 10^7$	10^{12}	10^{18}		

- Χειρότερη, Καλύτερη και Μέση περίπτωση
  - Υπάρχουν αλγόριθμοι για τους οποίους ο χρόνος εκτέλεσης δεν εξαρτάται μόνο από το πλήθος των δεδομένων αλλά και από την κατάσταση τους (π.χ. σειριακή αναζήτηση)

Ανάλυση Απόδοσης Αλγορίθμων

- Ασυμπτωτικός Συμβολισμός: Αν $t(n)$ ο χρόνος εκτέλεσης ενός αλγορίθμου με πλήθος βασικής πράξης $C(n)$, και $g(n)$ μια απλή συνάρτηση σύγκρισης
  - Συμβολισμός O: το σύνολο όλων των συναρτήσεων που έχουν κλίμακα ανάπτυξης μικρότερη ή ίση με $g(n)$, του n τείνοντος στο άπειρο
    - Π.χ $n \in O(n^2), 100n + 5 \in O(n^2), \frac{1}{2}n(n-1) \in O(n^2)$
    - Ενώ $n^3 \notin O(n^2), 0.0001n^3 \notin O(n^2), n64 + n + 1 \notin O(n^2)$
  - Συμβολισμός Ω: το σύνολο όλων των συναρτήσεων που έχουν κλίμακα ανάπτυξης μεγαλύτερη ή ίση με $g(n)$, του n τείνοντος στο άπειρο
    - Π.χ $n^3 \in \Omega(n^2), \frac{1}{2}n(n-1) \in \Omega(n^2)$ αλλά $100n + 5 \notin \Omega(n^2)$
  - Συμβολισμός Θ: το σύνολο όλων των συναρτήσεων που έχουν την ίδια κλίμακα ανάπτυξης με τη $g(n)$, του n τείνοντος στο άπειρο
    - Π.χ $n^2 + \sin n \in \Omega(n^2), \frac{1}{2}n(n-1) \in \Omega(n^2)$ αλλά $100n + 5 \notin \Omega(n^2)$

Βασικές Τάξεις Απόδοσης

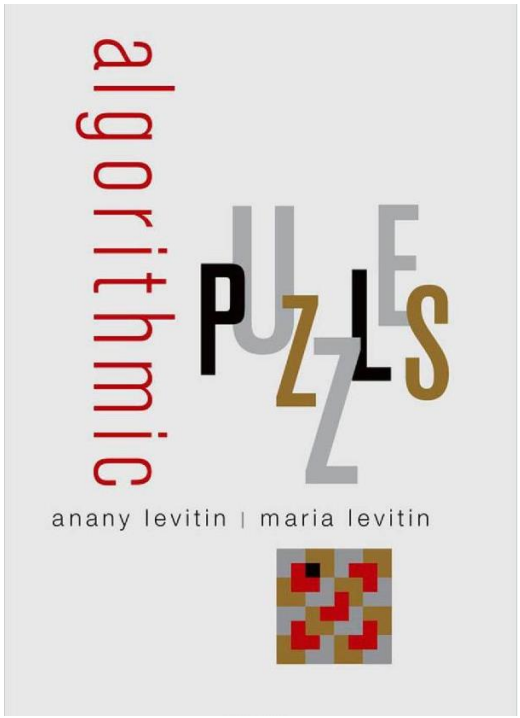
Τάξη	Όνομα	Σχόλια
1	Σταθερά	Αποδόσεις βέλτιστης περίπτωσης
$\log n$	Λογαριθμική	Το πρόβλημα τεμαχίζεται με μια σταθερά σε κάθε επανάληψη του αλγορίθμου
n	Γραμμική	Ο αλγόριθμος σαρώνει όλα τα στοιχεία λίστας μεγέθους n
$n \log n$	'n-log-n'	Αλγόριθμοι 'διαίρει και βασίλευε'
n^2	Τετραγωνική	Δυο εμφωλευμένες επαναλήψεις
n^3	Κυβική	Τρεις εμφωλευμένες επαναλήψεις
2^n	Εκθετική	Αλγόριθμοι που δημιουργούν όλα τα υποσύνολα ενός συνόλου με n στοιχεία
$n!$	Παραγοντική	Αλγόριθμοι που δημιουργούν όλους τους συνδυασμούς ενός συνόλου με n στοιχεία

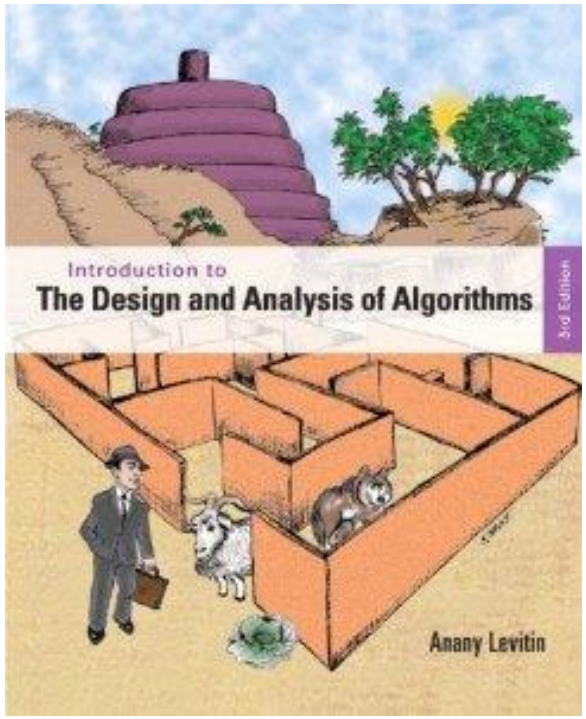
Οπτικοποίηση Αλγορίθμων

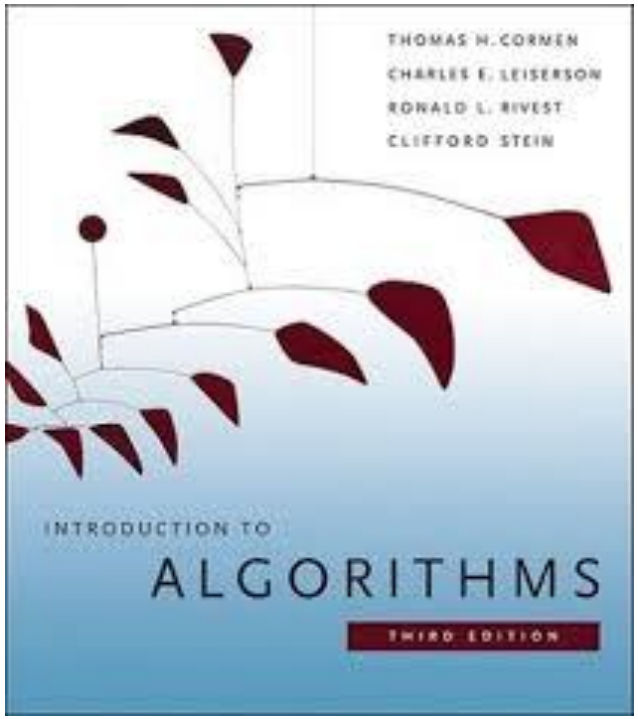
- http://www.sorting-algorithms.com/
- http://www.cs.oswego.edu/~mohammad/classes/csc241/samples/sort/Sort2-E.html
- http://people.cs.pitt.edu/~kirk/cs1501/animations/Sort2.html

Προτεινόμενη Βιβλιογραφία

- A. Levitin, M. Levitin: 'Algorithmic Puzzles'
- Anany Levitin: 'Introduction to The Design & Analysis of Algorithms'
- T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein: 'Introduction to Algorithms'



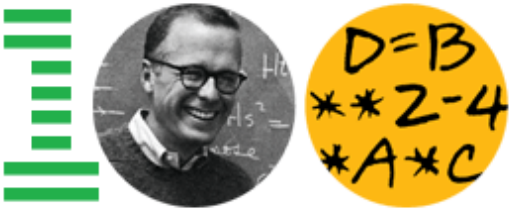




ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

at 100

http://www-03.ibm.com/ibm/history/ibm100/us/en/



FORTRAN

The Pioneering Programming Language

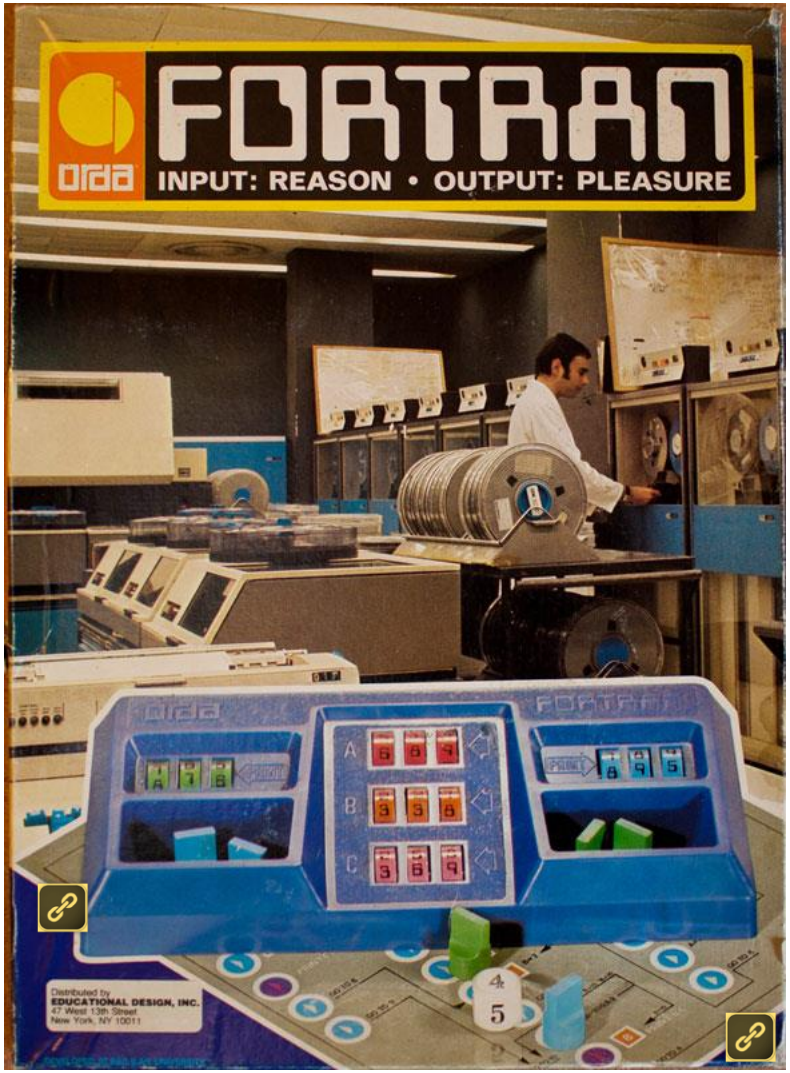
Programming early computers meant using an arcane "machine code" specific to each computer. IBM programmer John Backus found a better solution. In 1957, he and his team produced the first high-level language, FORTRAN (for FORmula TRANslator). A FORTRAN program could run on any system with a FORTRAN compiler, which translated Backus's code to machine code almost as efficiently as a good programmer. For the first time, code was comprehensible to people other than programmers, giving mathematicians and scientists the ability to write programs they could share on different systems. FORTRAN was a significant step toward freeing software from the constraints of its hardware.

http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/fortran/

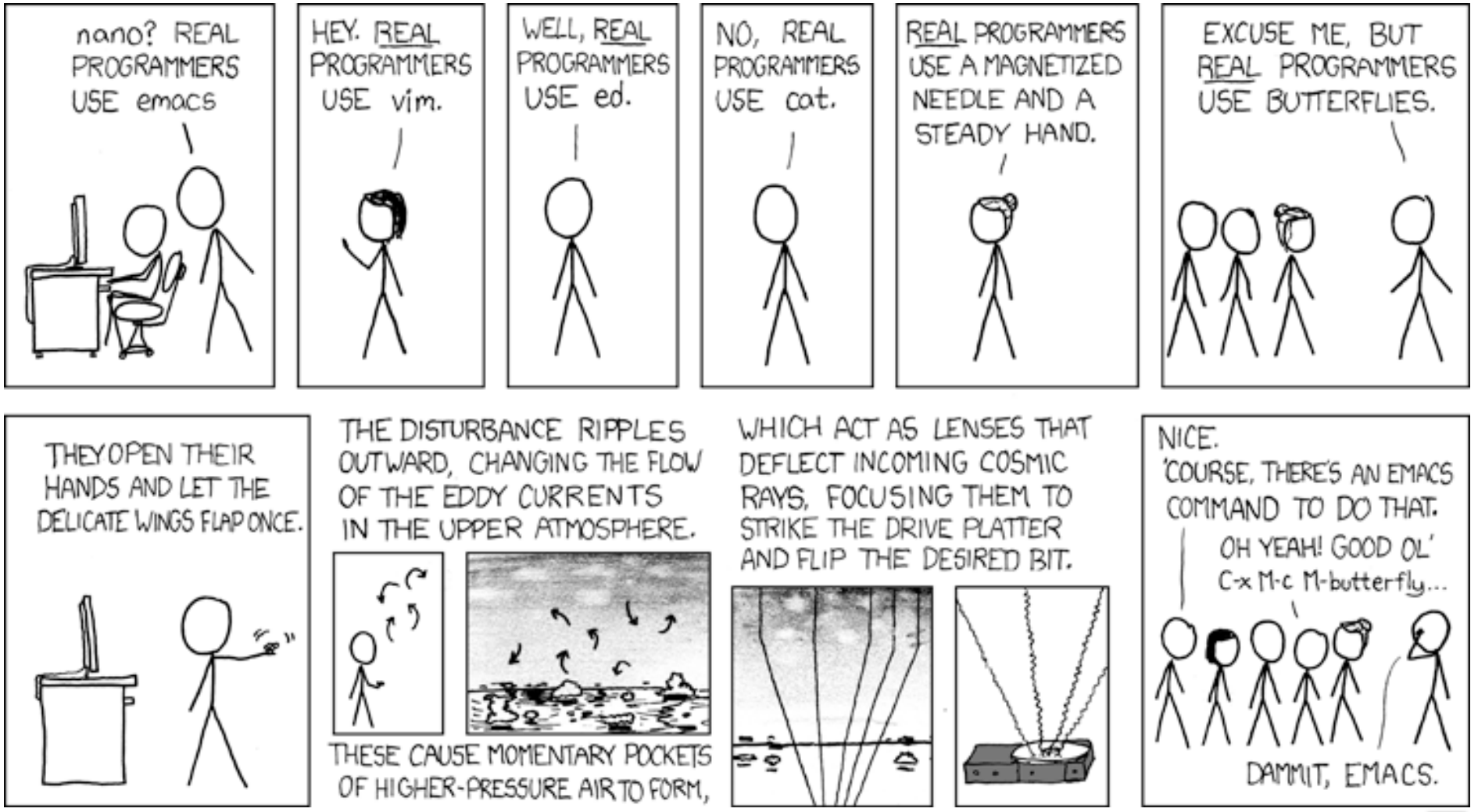
Το μαρτύριο της κάρτας

Βάζεις Λογική, Παίρνεις…Ευχαρίστηση!

- Από τη Λογική στην Ιδεολογία
- Προγραμματιστικός και Τεχνολογικός Πατριωτισμός (Οπαδισμός)
- Η έννοια του Μάγκα στον προγραμματισμό



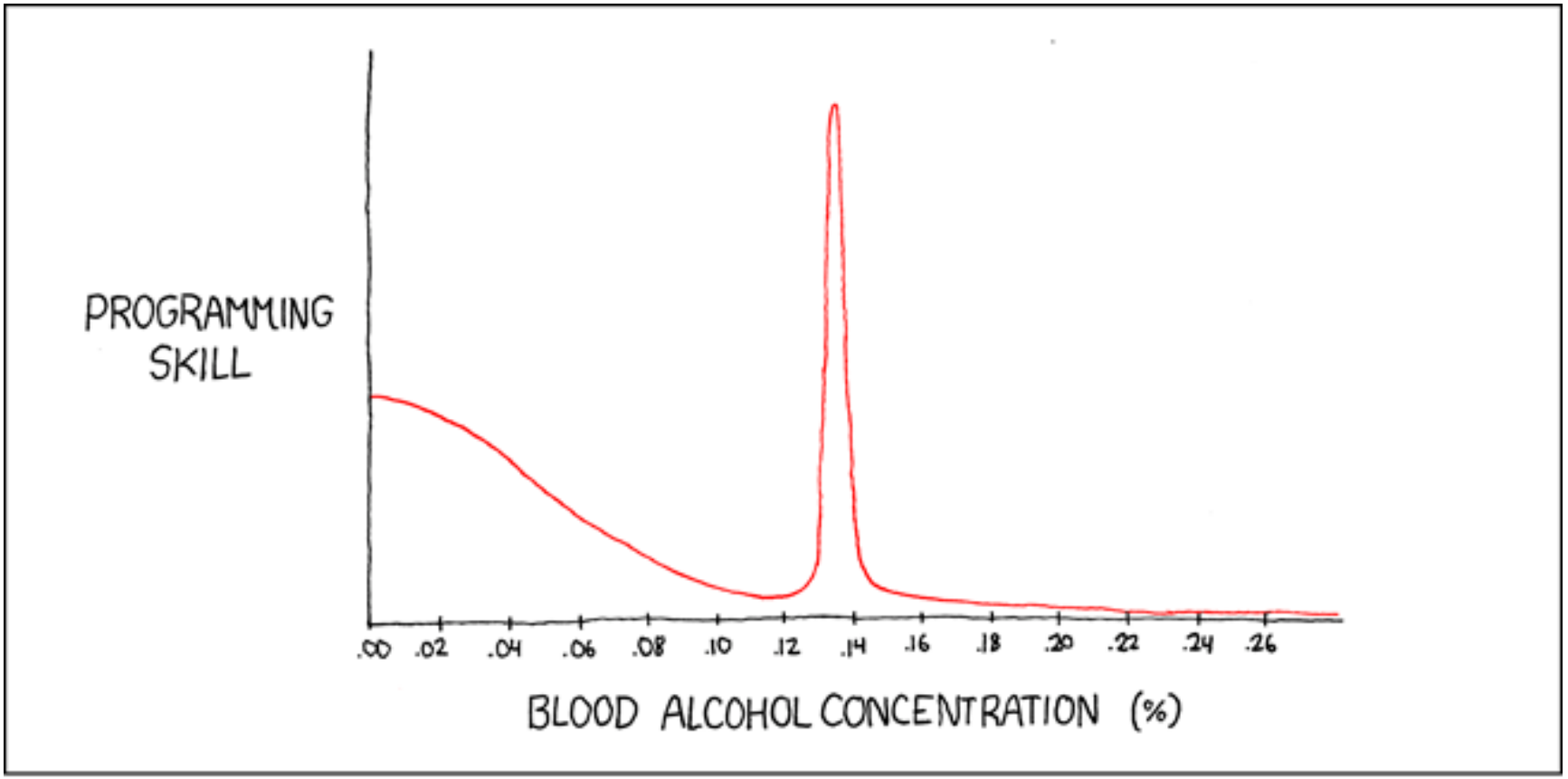
Τι γλώσσα χρησιμοποιεί ο Chuck Norris?



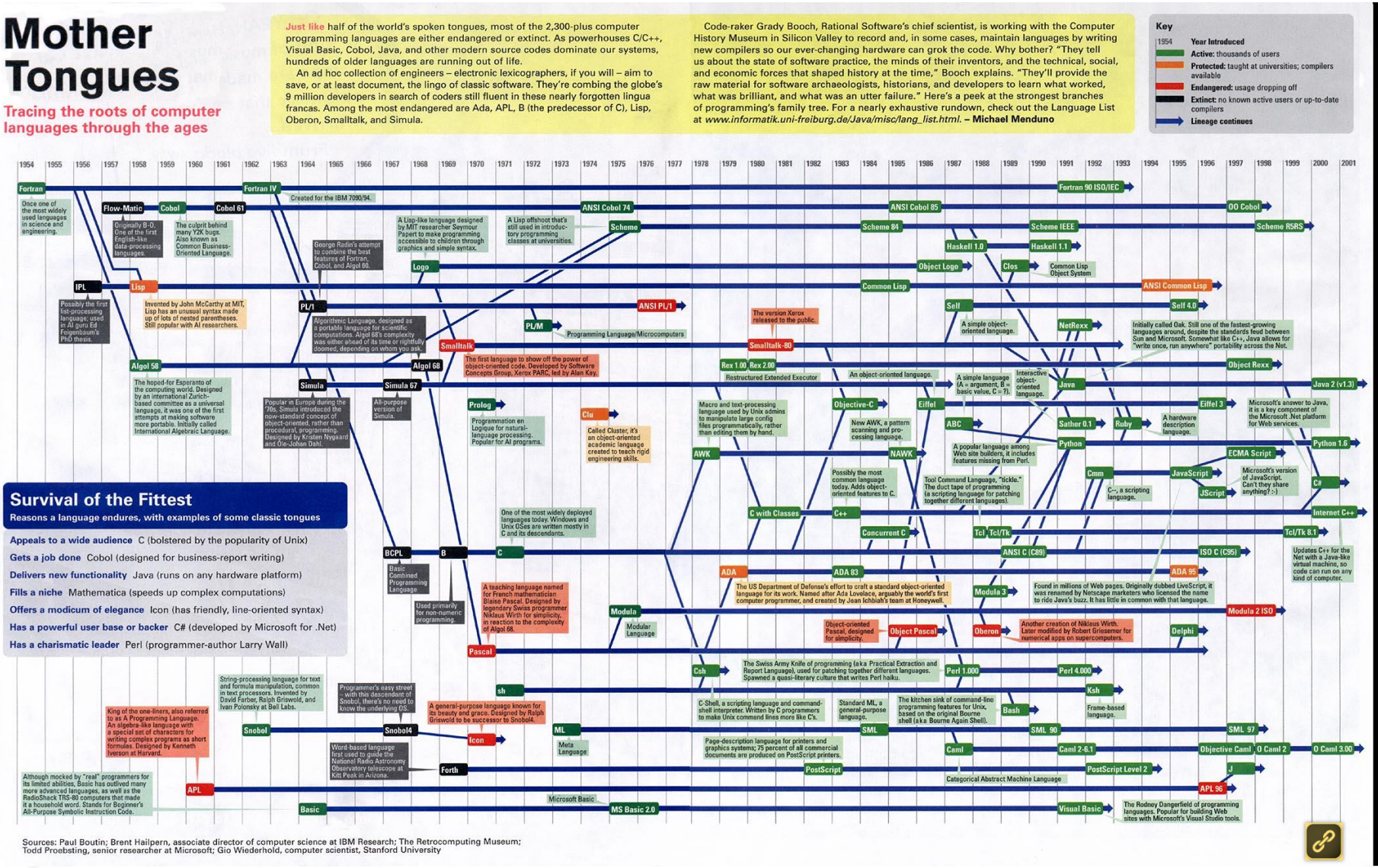
Πριν πέσουμε στα σκληρά…

- Μόλις κατακτήσει κανείς μερικές βασικές γνώσεις:
- Ο προγραμματισμός δίνει την αίσθηση της δημιουργικότητας και της ελευθερίας
- Ας μην ξεχνάμε ότι πρόκειται για ένα εργαλείο

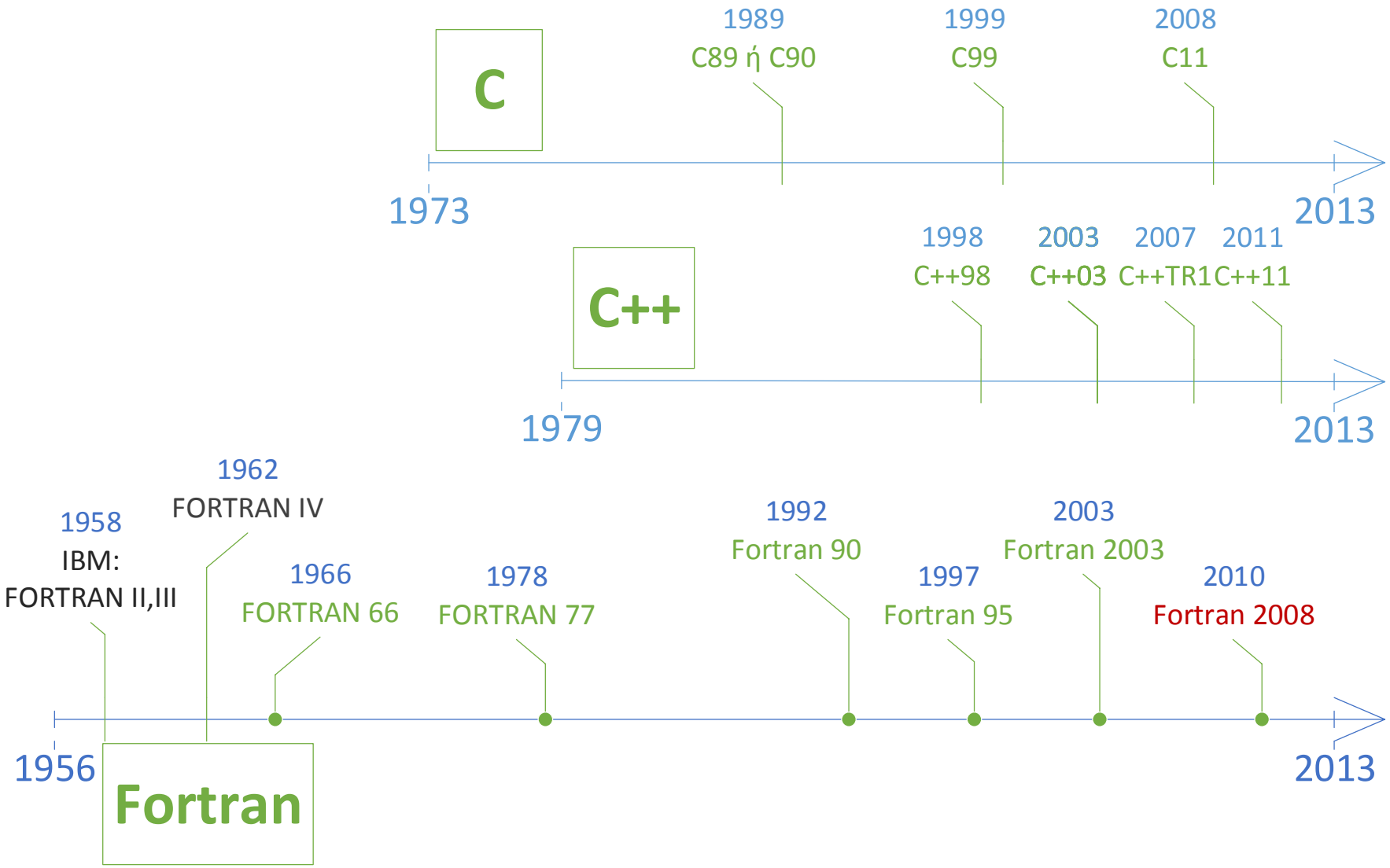
Με λίγη προσοχή όμως…



BABEL


Mother Tongues
Tracing the roots of computer languages through the ages
Just like half of the world's spoken tongues, most of the 2,300-plus computer programming languages are either endangered or extinct. As powerhouses C/C++, Visual Basic, Cobol, Java, and other modern source codes dominate our systems, hundreds of older languages are running out of life.
An ad hoc collection of engineers – electronic lexicographers, if you will – aim to save, or at least document, the lingo of classic software. They're combining the globe's 9 million developers in search of coders still fluent in these nearly forgotten lingua francas. Among the most endangered are Ada, APL, B (the predecessor of C), Lisp, Oberon, Smalltalk, and Simula.
Code-raker Grady Booch, Rational Software's chief scientist, is working with the Computer History Museum in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can grok the code. Why bother? "They tell us about the state of software practice, the minds of their inventors, and the technical, social, and economic forces that shaped history at the time," Booch explains. "They'll provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an utter failure." Here's a peek at the strongest branches of programming's family tree. For a nearly exhaustive rundown, check out the Language List at www.informatik.uni-freiburg.de/Java/misc/lang_list.html. – Michael Menduno
Key
1954 Year Introduced
Active: thousands of users
Protected: taught at universities; compilers available
Endangered: usage dropping off
Extinct: no known active users or up-to-date compilers
Lineage continues
1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001
Fortran
Once one of the most widely used languages in science and engineering.
Fortran IV
Created for the IBM 7090/94.
Fortran 90 ISO/IEC
Flow-Matic
Originally B-0. One of the first English-like data-processing languages.
Cobol
Cobol 61
The culprit behind many Y2K bugs. Also known as Common Business-Oriented Language.
ANSI Cobol 74
ANSI Cobol 85
OO Cobol
A Lisp-like language designed by MIT researcher Seymour Papert to make programming accessible to children through graphics and simple syntax.
A Lisp offshoot that's still used in introductory programming classes at universities.
Scheme
Scheme 84
Scheme IEEE
Scheme R5RS
George Radin's attempt to combine the best features of Fortran, Cobol, and Algol 60.
Haskell 1.0
Haskell 1.1
Logo
Object Logo
Clos
Common Lisp Object System
IPL
Possibly the first list-processing language; used in AI guru Ed Feigenbaum's PhD thesis.
Lisp
Invented by John McCarthy at MIT, Lisp has an unusual syntax made up of lots of nested parentheses. Still popular with AI researchers.
Common Lisp
ANSI Common Lisp
PL/1
Algorithmic Language, designed as a portable language for scientific computations. Algol 68's complexity was either ahead of its time or rightfully doomed, depending on whom you ask.
ANSI PL/1
Self
Self 4.0
PL/M
Programming Language/Microcomputers
The version Xerox released to the public.
A simple object-oriented language.
NetRexx
Initially called Oak. Still one of the fastest-growing languages around, despite the standards feud between Sun and Microsoft. Somewhat like C++, Java allows for "write once, run anywhere" portability across the Net.
Smalltalk
Smalltalk-80
Algol 58
The hoped-for Esperanto of the computing world. Designed by an international Zurich-based committee as a universal language, it was one of the first attempts at making software more portable. Initially called International Algebraic Language.
Algol 68
The first language to show off the power of object-oriented code. Developed by Software Concepts Group, Xerox PARC, led by Alan Key.
Rex 1.00
Rex 2.00
Restructured Extended Executor
Object Rexx
Simula
Popular in Europe during the '70s, Simula introduced the now-standard concept of object-oriented, rather than procedural, programming. Designed by Kristen Nygaard and Ole-Johan Dahl.
Simula 67
All-purpose version of Simula.
An object-oriented language.
A simple language (A = argument, B = basic value, C = ?).
Interactive object-oriented language.
Java
Java 2 (v1.3)
Prolog
Programmation en Logique for natural-language processing. Popular for AI programs.
Clu
Called Cluster, it's an object-oriented academic language created to teach rigid engineering skills.
Macro and text-processing language used by Unix admins to manipulate large config files programmatically, rather than editing them by hand.
Objective-C
Eiffel
Eiffel 3
Microsoft's answer to Java, it is a key component of the Microsoft .Net platform for Web services.
New AWK, a pattern scanning and processing language.
ABC
Sather 0.1
Ruby
A hardware description language.
AWK
NAWK
A popular language among Web site builders, it includes features missing from Perl.
Python
Python 1.6
ECMA Script
Possibly the most common language today. Adds object-oriented features to C.
Tool Command Language, "tickle." The duct tape of programming (a scripting language for patching together different languages).
Cmm
JavaScript
Microsoft's version of JavaScript. Can't they share anything? :-)
C--, a scripting language.
JScript
C#
One of the most widely deployed languages today. Windows and Unix OSes are written mostly in C and its descendants.
C with Classes
C++
Internet C++
Concurrent C
Tcl
Tcl/Tk
Tcl/Tk 8.1
BCPL
B
C
ANSI C (C89)
ISO C (C95)
Updates C++ for the Net with a Java-like virtual machine, so code can run on any kind of computer.
Basic Combined Programming Language
ADA
ADA 83
ADA 95
The US Department of Defense's effort to craft a standard object-oriented language for its work. Named after Ada Lovelace, arguably the world's first computer programmer, and created by Joan Ichbiah's team at Honeywell.
Modula 3
Found in millions of Web pages. Originally dubbed LiveScript, it was renamed by Netscape marketers who licensed the name to ride Java's buzz. It has little in common with that language.
A teaching language named for French mathematician Blaise Pascal. Designed by legendary Swiss programmer Niklaus Wirth for simplicity, in reaction to the complexity of Algol 68.
Used primarily for non-numeric programming.
Modula
Modula 2 ISO
Modular Language
Object-oriented Pascal, designed for simplicity.
Object Pascal
Oberon
Another creation of Niklaus Wirth. Later modified by Robert Griesemer for numerical apps on supercomputers.
Delphi
Pascal
Survival of the Fittest
Reasons a language endures, with examples of some classic tongues
Appeals to a wide audience C (bolstered by the popularity of Unix)
Gets a job done Cobol (designed for business-report writing)
Delivers new functionality Java (runs on any hardware platform)
Fills a niche Mathematica (speeds up complex computations)
Offers a modicum of elegance Icon (has friendly, line-oriented syntax)
Has a powerful user base or backer C# (developed by Microsoft for .Net)
Has a charismatic leader Perl (programmer-author Larry Wall)
Csh
The Swiss Army Knife of programming (a.k.a Practical Extraction and Report Language), used for patching together different languages. Spawned a quasi-literary culture that writes Perl haiku.
Perl 1.000
Perl 4.000
String-processing language for text and formula manipulation, common in text processors. Invented by David Farber, Ralph Griswold, and Ivan Polonsky at Bell Labs.
Programmer's easy street - with this descendant of Snobol, there's no need to know the underlying OS.
sh
Ksh
King of the one-liners, also referred to as A Programming Language. An algebra-like language with a special set of characters for writing complex programs as short formulas. Designed by Kenneth Iverson at Harvard.
A general-purpose language known for its beauty and grace. Designed by Ralph Griswold to be successor to Snobol4.
C-Shell, a scripting language and command-shell interpreter. Written by C programmers to make Unix command lines more like C's.
Standard ML, a general-purpose language.
The kitchen sink of command-line programming features for Unix, based on the original Bourne shell (a.k.a Bourne Again Shell).
Bash
Frame-based language.
Snobol
Snobol4
Icon
ML
Meta Language
SML
SML 90
SML 97
Word-based language first used to guide the National Radio Astronomy Observatory telescope at Kitt Peak in Arizona.
Page-description language for printers and graphics systems; 75 percent of all commercial documents are produced on PostScript printers.
Caml
Caml 2-6.1
Objective Caml
O Caml 2
O Caml 3.00
Forth
PostScript
PostScript Level 2
J
Although mocked by "real" programmers for its limited abilities, Basic has outlived many more advanced languages, as well as the RadioShack TRS-80 computers that made it a household word. Stands for Beginner's All-Purpose Symbolic Instruction Code.
APL
Categorical Abstract Machine Language
APL 96
Microsoft Basic
Basic
MS Basic 2.0
Visual Basic
The Rodney Dangerfield of programming languages. Popular for building Web sites with Microsoft's Visual Studio tools.
Sources: Paul Boutin; Brent Hailpern, associate director of computer science at IBM Research; The Retrocomputing Museum; Todd Proebsting, senior researcher at Microsoft; Gio Wiederhold, computer scientist, Stanford University

Η χρονογραμμή της Fortran



Γιατί Fortran;

- Οι μύθοι προκύπτουν όταν γενικεύουμε αυθαίρετα μια επιμέρους προσωπική εμπειρία
- Υπάρχει τεράστιος όγκος δουλειάς και ανθρώπινο κεφάλαιο (εκπαίδευση, εμπειρία) παγκόσμια
- Πολύ σημαντικές 'κρίσιμες εφαρμογές' γράφονται σε Fortran (πρόγνωση καιρού, αεροπλάνα)
- Είναι η ευκολότερη γλώσσα προγραμματισμού για να μάθει κανείς, ιδιαίτερα σαν πρώτη γλώσσα
- Είναι πολύ ευκολότερο για ένα προγραμματιστή Fortran να μάθει C++ παρά το αντίθετο
- Η Fortran είναι σήμερα μια δομημένη αντικειμενοστρεφής γλώσσα
- Χρειαζόμαστε μια γλώσσα για να κάνουμε επιστημονικούς υπολογισμούς (δεν θα γράψουμε εμπορικά προγράμματα ή λειτουργικά συστήματα)
- Όταν ένα εργαλείο συνεχίζει να χρησιμοποιείται μετά από τόσα πολλά χρόνια μάλλον είναι καλό εργαλείο

Ποια γλώσσα για ποια χρήση;

- Ευκολία Χρήσης: Python και Matlab, μετά Fortran. C++ η χειρότερη
- Prototyping: Matlab, μετά Python
- Αποσφαλμάτωση: Python, Matlab, μετά Fortran
- Μεταφερσιμότητα: Fortran η καλύτερη με απόσταση. C++ η χειρότερη
- Επιδόσεις: Fortran ή C++ (χωρίς μεγάλες διαφορές). Python και Matlab πολύ πιο αργές
- Παραλληλισμός: Fortran η καλύτερη για shared memory. Πρόβλημα με distributed memory
- Χειρισμός Πινάκων: Fortran η καλύτερη, μετά Matlab και Python.
- Χειρισμός Χαρακτήρων: Python η καλύτερη, μετά C++, μετά Fortran
- Computer Science, Software Engineering, System interfaces: C++, μετά Python, μετά Fortran

Τελικά Fortran γιατί…

Παλιές Κινέζικες Παροιμίες:

- Είναι καλύτερα να μιλάς κινέζικα στους κινέζους
- Άσπρη γάτα, μαύρη γάτα αρκεί να πιάνει ποντίκια
- Όταν φυσάει ο άνεμος της αλλαγής κάποιοι χτίζουν τείχη και κάποιοι άλλοι ανεμόμυλους

風向轉變時有人築牆有人造風車

Εργαλεία…

- Μεταγλωττιστές Fortran
  - G95
  - Gfortran
  - Intel Fortran Compiler
  - Numerical Algorithms Group
  - Open64
  - Oracle Solaris Studio
  - PathScale
  - The Portland Group
  - Silverfrost FTN95
  - IBM VisualAge

- Integrated Development Environment
  - MS Visual Studio
  - Code::Blocks
  - …

Ελεύθερα… Εργαλεία…

Code::Blocks
The open source, cross-platform IDE
http://www.codeblocks.org

- GNU: πρόδρομος (1984!) και το μεγαλύτερο κομμάτι αυτού που λέμε σήμερα linux
- GCC: GNU Compiler Collection. Front Ends για τις γλώσσες: C, C++, Objective-C, Fortran, Java, Ada, και Go
- Code::Blocks IDE: Integrated Development Environment Ολοκληρωμένο Περιβάλλον Ανάπτυξης Εφαρμογών για τις γλώσσες C, C++, Fortran
- Δείτε τον οδηγό εγκατάστασης για τα windows στο eclass

Ellen Ullman: “The Bug”

Αναφέρεται σε ένα προγραμματιστή με προσωπική ζωή υπό διάλυση, ο οποίος έχει πάθει εμμονή με ένα μυστηριώδες σφάλμα (Bug) σε ένα πρόγραμμα CAD. Το ερώτημα είναι: Η ζωή του διαλύεται λόγω της εμμονής του με το “bug” ή η εμμονή του λειτουργεί ως καταφύγιο για να αποφύγει να ασχοληθεί με τα πραγματικά του προβλήματα. Η Ullman είναι η ίδια προγραμματίστρια και η ιστορία είναι πολύ κοντά στην προσωπική της ζωή.

Σημείωμα Αναφοράς

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα Αδειοδότησης